BELIEF REVISION, NON-MONOTONIC REASONING AND SECRECY FOR EPISTEMIC AGENTS

DISSERTATION

ZUR ERLANGUNG DES GRADES EINES
DOKTORS DER NATURWISSENSCHAFTEN

der Technischen Universität Dortmund
an der Fakultät für Informatik

PATRICK KRÜMPELMANN

Technische Universität Dortmund — Fakultät für Informatik — Dortmund, Germany — 2015

Tag der mündlichen Prüfung:	20.10.2015
Dekan:	Prof. Dr. Gernot A. Fink
Gutachter/Gutachterinnen:	Prof. Dr. Gabriele Kern-Isberner
	Prof. Dr. Christoph Beierle

Patrick Krümpelmann: *Belief Revision, Non-monotonic Reasoning and Secrecy for Epistemic Agents*, Dissertation, Department of Computer Science, Technische Universität Dortmund, Germany. © 2015

ABSTRACT

Software agents are increasingly used to handle the information of individuals and companies. They also exchange this information with other software agents and humans. This raises the need for sophisticated methods of such agents to represent information, to change it, reason with it, and to protect it. We consider these needs for communicating autonomous agents with incomplete information in a partially observable, dynamic environment. The protection of secret information requires the agents to consider the information of agents, and the possible inferences of these. Further, they have to keep track of this information, and they have to anticipate the effects of their actions. In our considered setting the preservation of secrecy is not always possible. Consequently, an agent has to be able to evaluate and minimize the degree of violation of secrecy. Incomplete information calls for non-monotonic logics, which allow to draw tentative conclusions. A dynamic environment calls for operators that change the information of the agent when new information is received.

We develop a general framework of agents that represent their information by logical knowledge representation formalisms with the aim to integrate and combine methods for non-monotonic reasoning, for belief change, and methods to protect secret information. For the integration of belief change theory, we develop new change operators that make use of non-monotonic logic in the change process, and new operators for non-monotonic formalisms. We formally prove their adherence to the quality standards taken and adapted from belief revision theory. Based on the resulting framework we develop a formal framework for secrecy aware agents that meet the requirements described above. We consider different settings for secrecy and analyze requirements to preserve secrecy. For the protection of secrecy we elaborate on change operations and the evaluation of actions with respect to secrecy, both declaratively and by providing constructive approaches. We formally prove the adherence of the constructions to the declarative specifications. Further, we develop concrete agent instances of our framework building on and extending the well known BDI agent model. We build complete secrecy aware agents that use our extended BDI model and answer set programming for knowledge representation and reasoning. For the implementation of our agents we developed Angerona, a Java multiagent development framework. It provides a general framework for developing epistemic agents and implements most of the approaches presented in this thesis.

ACKNOWLEDGMENTS

First of all, I want to thank my advisor Gabriele Kern-Isberner. She always greatly supported me. I learned countless valuable lessons in diverse technical as well as organizational areas from her. Our discussions and her advises were of great value. I want to thank Christoph Beierle for being my second reviewer, for the interesting discussions and helpful comments. I want to thank Heinrich Müller and Stefan Dißmann for participating in my dissertation committee.

I am very grateful to the Collaborative Research Center SFB 876 and in particular to Katharina Morik, Wolfgang Rhode, Stefan Michaelis and Jens Teubner. Being a part of the interdisciplinary Collaborative Research Center with the joint meetings and workshops, working groups, and its graduate school was highly inspiring and supportive. My work in Project A5 of the SFB 876 with Gabriele Kern-Isberner, Joachim Biskup, and Cornelia Tadros formed the basis of a big part of my thesis. I want to thank the project leaders Gabriele Kern-Isberner and Joachim Biskup for creating and leading the project. I am very thankful to Joachim Biskup for his support, the intensive discussions and the deep insights into the topic of secrecy. The joint work with Cornelia Tadros was very enriching, productive, and pleasant. Our perspectives and approaches often nicely complemented each other. Organizing and supervising an internship as part of the Summer Research Program of the German Academic Exchange Service (DAAD) together was also a valuable experience. Further, we collaborated on the implementation of our framework and approaches. With respect to this, Tim Janus was of exceptional help working with us on the implementation of the Angerona framework.

I always felt very comfortable in my working environment at the TU Dortmund and I am very thankful to all my wonderful colleagues for contributing to this. In particular, I want to thank Matthias Thimm. He was my colleague from the first hour, we collaborated in several works and projects, and we shared many research trips. I learned a lot from him. I am also especially grateful to Daniela Huvermann and Daan Apeldoorn with whom I shared my office, and to Thomas Zeume. I want to thank the people from the LIDIA lab in Bahía Blanca, Argentina for the enriching exchange in our research project, which was funded by DAAD.

Finally, I am deeply grateful to Lisa Kruppa for her support.

If the human brain were so simple that we could understand it,
we would be so simple that we couldn't.

— Emerson M. Pugh

CONTENTS

1 INTRODUCTION

In the following we motivate and introduce the topic of this thesis, and the fields of research it is related to. Afterwards, we go into more detail of the addressed challenges and the contributions of the author. Then, we give an overview of the structure of this document, and describe the publications that contributed to this thesis, and the contributions to these.

1.1 CONTEXT AND MOTIVATION

In the context of communicating agents the need to consider the preservation of secret information arises naturally. In almost all realistic scenarios agents are interested to withhold some information while communicating with other agents. While communication is necessary to achieve goals, private information should not be revealed. For example, the political or sexual orientation should not matter in job interviews or negotiations; medical information should only be revealed if needed for the treatment of a patient, companies do not want exact specifications of their products to be revealed to competitors. Intelligent software systems are increasingly used to manage information and information flows in every day life on the personal and the professional level. Individuals and companies interact with software agents and let them act on their behalf. Interaction and publication of personal information via social networks is rapidly growing and generates abundant *social inference opportunities* [175]. This gives rise to the necessity to be able to model and deal with real world secrecy scenarios in multiagent systems.

We use the following example to illustrate typical aspects of such scenarios: The employee *Emma* is working in a company for her boss *Beatriz*. *Emma* wants to attend a strike committee meeting (*scm*) the next day and has to ask *Beatriz* for a day off in order to attend. *Emma* knows that *Beatriz* puts everyone who attends the *scm* on her list of employees to be fired next. Thus, *Emma* wants to keep her attendance to the *scm* secret from *Beatriz*, but has to communicate with her in order to achieve her goal of getting the day off. *Emma* does not even want *Beatriz* to be suspicious of her attending the *scm*. She also does not want other employees that are in opposition to the strike to believe that she attends the *scm*. However, she considers it sufficient that these other employees do not know for sure that she attends – she does not care if they are suspicious. From other employees who also want to attend the *scm* she does not want to keep her attendance secret. If *Emma* communicates with other agents, she has to be aware and keep track of the information she discloses to them. Moreover, she also has to be aware of the reasoning capabilities and behavior of the other agents. This includes the consideration of *meta-inferences*, i. e., that *Beatriz* might draw conclusions from *Emma*'s behavior. *Beatriz* might, for instance, conclude from *Emma* avoiding to answer the question whether she intends to attend the *scm*, that *Emma* does intend to attend. *Emma* has to choose her utterances taking these aspects into consideration. And, if *Emma* realizes that *Beatriz* overheard her phone call with the strike committee she should reconsider her secret.

This small scenario already illustrates well that the aspect of secrecy poses a plethora of challenges to the specification and implementation of secrecy preserving agents. Consider for example, that *Emma* and *Beatriz* use calendar agents and let these automatically negotiate appointments and vacations. Secrets have to be specified with respect to other agents and towards a, more or less credulous, reasoning behavior. Incomplete, subjective views on other agents have to be represented and changed adequately upon reception or emission of a speech act. The agent has to consider its secrets in its deliberation and means-ends reasoning processes. It also has to be able to reconsider and possibly modify its secrets if new information makes it necessary.

The existing work on secrecy in multiagent systems provides formal methods for the formalization and the analysis of strong secrecy guarantees with respect to an attacker that is a perfect reasoner, for overviews see, e. g., [121, 232]. These works consider secrecy on the theoretical level under idealized assumptions. Sophisticated constructive approaches for the preservation of secrecy have been developed for various specific scenarios, such as query answering for databases; for an overview see, e. g., [32]. These constructive approaches are mostly based on strong and strict notions of secrecy with respect to attackers that are perfect reasoners. As observed by Joseph Y. Halpern in [121] a major task for future work on secrecy is the "careful consideration of how secrecy definitions can be weakened to make them more useful in practice".

In this thesis, we consider secrecy for practical multiagent systems in which agents have to achieve their goals under incomplete information in a dynamic environment. We define and consider this secrecy setting on different levels of abstraction, from an abstract agent model down to concrete instances and the implementation of these. Our work shows that the consideration of secrecy is a complex task and has to be thoroughly incorporated into the agent model.

While in the area of secrecy preservation it is demanded that secrecy is always preserved, in the area of research on practical multiagent systems it is accepted with respect to such *maintenance conditions* f that "in many cases, it is impossible for an agent to exhibit such a high degree of control over the environment to guarantee *always* f" [82]. Moreover, even if it is theoretically possible, the agent under consideration might not have the necessary information or computational resources for it. It is argued that an agent has to have proactive and reactive methods available to minimize violations of the maintenance condition [82]. We take on this perspective with respect to secrecy and develop *secrecy aware agents* that do their best to avoid violations of secrecy, if that is not possible they do their best to violate secrecy as little as viable and to restore secrecy as fast and well as feasible. From this perspective, the strict preservation of secrecy is a special case and obtained whenever it is possible.

Motivated by the goal to consider secrecy in practical multiagent systems we start with the design of an agent model, and instantiations of it that are apt for this task. The consideration of secrecy requires an agent to be able to adequately represent information available to it, and the information available to the potential attackers of its secrecy interests. It has to reason on the basis this information, and it has to simulate the possible inferences of other agents. Further, it has to adapt this representation adequately while interacting with other agents. Hence, our approach of secrecy aware agents calls for sophisticated, and yet effective, methods for reasoning under incomplete information, and for well founded operators of belief change. The agent model has to be general to be able to define secrecy and methods for secrecy aware agents independently from a concrete agent architecture and knowledge representation formalism. Further, the same model shall be used to define concrete agent architectures that can be used to implement secrecy aware agents.

We define a notion of agents that are based on logical knowledge representation that we call *epistemic agents*[1]. These have an *epistemic state* which contains a logic-based representation of their entire knowledge, e. g., their knowledge about the world, other agents, their procedural knowledge. A functional component realizes the adaption of the epistemic state when the agent obtains new information from its environment, and the execution of the agent's actions, which are determined by the epistemic state, in the environment. The epistemic agent model shall be the basis for the combination of (non-monotonic) logical knowledge representation and reasoning, belief change theory, abstract agent models and compound concrete agent architectures.

We integrate non-monotonic reasoning [60] and belief change theory [94] in our agent model, and we combine both. We define families of (non-monotonic) belief operators and define desirable properties for these. We define an order on belief operators based on the credulity of their inferences behavior. Different operators can be used by different agents, and for the protection of information to be kept

1 The term epistemic agent is also used in philosophy, see, e. g., [2], and epistemic modal logics, see, e. g., [231] that describe reasoning agents.

secret. We instantiate these operators for propositional logic and for non-monotonic extension based formalisms [173], and in particular for answer set programming [112]. For *answer set programming* (ASP) many very effective solvers and several extensions are available, e. g., [63, 108, 18, 77, 84, 88, 184], and ASP is used in several real world applications, e. g., [218, 203, 185, 110, 44, 109]. We consider belief change operations for agents by means of, and for, non-monotonic logics. We develop functions to decide which part of new information shall be accepted by the agent by use of argumentation theory [28]. Moreover, we develop new revision operations for non-monotonic formalisms by example of answer set programming. Based on the general epistemic agent model we define a concrete compound agent model that is based on the well known BDI agent model [57, 247]. It extends the BDI model by a *motivation* and a *know-how* component that we developed.

Having defined the epistemic agent model in detail, we develop an agent based, subjective, notion on secrecy by use of it. We define a formal notion of secrecy and of the preservation thereof. We consider secrecy in different settings and elaborate on the secrecy relevant properties of all main tasks and components of an agent with respect to our notion of secrecy. Hereby, we distinguish between *secrecy aware* agents that take secrecy into consideration in their decision processes, and *secrecy preserving* agents, that preserve secrecy with respect to the information available to them. We develop a concrete instantiation of our secrecy agent model on the basis of our BDI$^+$ agent model and by use of answer set programming.

Furthermore, we present a multiagent programming framework that is based on our epistemic agent model. It is called ANGERONA and provides a plug-in based toolbox to develop different agent models in combination with different knowledge representation formalisms. We used it to implement the approaches presented in this thesis, and others, e. g., [75, 76, 152, 131, 244, 4, 229, 157, 159].

In the following we go into more detail on the motivating challenges behind the individual contributions of this thesis and afterwards detail our contributions towards these challenges.

1.2 CHALLENGES

Within the scenario and our goals we described above, we are facing numerous challenges. In the following we describe the most important of these and their relation to the state of the art.

1.2.1 *Gap between theoretical and practical agent models*

Theoretical formalizations of agency are mostly based on modal logics [42] and used to model and analyze multiagent systems [199, 64, 25, 116, 208, 89]. Some logical languages can be used to model what is called a *deductive agent* in which the behavior of the agent is determined entirely by deduction based on a set of logical formulae [115, 149, 166]. Some simple variants of the logical languages with better computational properties have been developed to specify agents and directly execute this specification, e. g., in [195, 167, 66]. This approach is very appealing but comes with drawbacks such as computational cost or limitations of the expressivity of the used languages. Moreover, they only model some aspects of an agent system while "an all embracing theory is some time off" [248].

There is a gap between these abstract models of agency and the agent models used for concrete agent architectures and multiagent systems [248, 182]. The mostly used approach to design concrete agents is based on a decomposition approach, determining the subcomponents of an agent and their interactions [51, 248]. It allows for a clear and modular design of agents, and facilitates more effective decision making. The *Beliefs, Desires, Intentions*, or BDI, model [57, 194] is the most widely used composition step in this process. Practical multiagent system based on such compositional approaches use very simple forms of knowledge representation and reasoning, such as production rules [51]. Further, each framework and system is fixed to one logical language or to one composition of the internal state of an agent.

We need an agent model that is abstract and can be used for the general formulation of properties, such as the preservation of secrecy, and that can be used to define concrete agent models. Further, this agent model should facilitate the use of diverse (non-monotonic) knowledge representation formalisms and the use of sophisticated belief change operators.

1.2.2 *Using non-monotonic reasoning in multiagent systems*

A variety of logical formalisms with different expressivity and computational properties have been developed for knowledge representation with the agent paradigm in mind, see, e. g., [55, 234]. Especially non-monotonic formalisms are designed to deal with incomplete information and to enable an agent to act in uncertain environments. Yet, very little of the developed approaches are considered in practical multiagent frameworks [51]. This is mostly due to the computational cost of these.

For non-monotonic logics some attempts have been made to come up with effective solvers for the formalisms. The most prominent of these is answer set programming (ASP), where several very active groups have worked on building effective solvers which lead to the availability of very effective solvers today, most notably smodels [184], DLV [165] and Potassco [108]. Implementations of formal argumentation systems [24] are also under development and the first systems are already available, such as DeLP [105] and ASPIC [192].

Several proposals exists how to implement agents based on answer set programming [182, 150, 16, 164]. However, the proposals for the implementation have not found their way in current agent programming platforms [51]. A general integration of different non-monotonic reasoning formalisms for practical agents is lacking.

1.2.3 *Belief revision and non-monotonic logics in multiagent systems*

In the field of research on belief change, the problem of how to change an agent's beliefs in the light of new information has been considered for over 25 years already [94]. While there is a profound theoretical basis, very little of the approaches developed in this field of research are considered in agent models or are available in practical multiagent frameworks. First approaches have been made in the field of belief change to develop approaches that might be applicable in practical multiagent frameworks. One important step is the consideration of operations on finite belief bases instead of infinite, deductively closed belief sets [142, 183, 122, 127, 128, 119]. These operations are guided by the principle of *cognitive realism* [128], in contrast to the assumption of a perfect reasoner. Further work to find belief change operations for less idealized agents studies operators that focus on relevant parts of a belief state [240]. Recently, belief change for less expressive languages than propositional logic, mostly for Horn theory, has been studied [50, 73, 70, 6]. None of these approaches has, to our knowledge, been used in a multiagent system.

Many approaches to the revision of non-monotonic logic programs have been proposed, but most of these are not based or related to belief change theory. Recently, some attempts have been made to relate these approaches to belief change theory [87]. The results showed that the considered postulates from belief change theory are not satisfied by the considered approaches. Also, attempts have been made to apply model-based revision operators from belief change theory to logic programs, e. g., [68, 72, 217]. These showed that the model-based techniques can be applied to logic programs, but it is not clear that the results are desirable in the logic programming setting [217]. Further, also these approaches have not been used in agent systems. Hence there is a lack of change operators that are based on belief change theory and that are adequate for the use in practical multiagent systems.

1.2.4 *Secrecy under incomplete information and bounded rationality*

Besides non-monotonic reasoning and belief change, a third field in research related to multiagent systems that has mostly been considered in theory under highly idealized assumptions is the one of secrecy in multiagent systems. Theoretical models of secrecy in multiagent systems have been formalized and analyzed [121]. Modal logics have been developed to model secrecy [232], but more realistic models or agent architectures are missing. More practical, and well studied, approaches from the database community, e. g., [213, 32, 46, 38, 34] have only been started to be studied in the context of concrete agent architectures in [39, 222]. Especially for secrecy, rational belief change and non-monotonic reasoning methods are important to achieve realistic models of potential attacking agents and their reasoning capabilities. Secrecy from the point of view of an agent with incomplete information, applying non-monotonic reasoning and belief change operators, has not been considered.

1.2.5 *A multiagent framework for epistemic agents*

A variety of frameworks for the implementation of multiagent systems exist, for overviews see, e. g., [51, 1]. Most of these have fixed agent architectures and fixed knowledge representation formalisms. Only few feature some modularity of the knowledge representation formalism, feature ASP based modules and allow for the development of new ones, e. g., [186, 130, 148]. None of them features belief change operations based on belief change theory, or the implementation of secrecy aware agents. The challenge is to design and implement a framework that facilitates the easy design and implementation of diverse types of agents with respect to their agent architecture and knowledge representation formalisms, including belief change operators based on belief change theory. This framework should allow to set up systems with heterogenous agents that interact with each other. Such a framework could be used to implement secrecy preserving agents, and to evaluate different knowledge representation formalisms and operators.

1.3 CONTRIBUTIONS

In the following we describe our contributions with respect to the challenges described in Section 1.2.

1.3.1 *General model of epistemic agents*

We present a general model of epistemic agents. Epistemic agents comprise an epistemic state and a functional component. The epistemic state contains a logic-based representation of the agent's knowledge. The functional component realizes the change of the epistemic state and the execution of actions. The epistemic state and the functional component might consist of epistemic components and compositions of sub-functions. The novelty of our model it is that it unifies diverse agent architectures, that it integrates the specification and the realization of agents, and that it integrates non-monotonic knowledge representation and belief change operators. This way it encompasses other agent architectures from purely deductive agents with a monolithic epistemic state to structured ones such as the popular Beliefs, Desires, Intentions (BDI) model. We focus on epistemic agents that communicate by means of speech acts. We motivate the ability of the agents to reflect the anticipated changes of their (communication) actions in their epistemic states. To this end, we introduce a change operator for the agent's actions.

We develop a sub-framework of our general epistemic agent model that extends the well known BDI model [56] by a *motivation* and a *know-how* component. We call it BDI$^+$ agent model. The motivation component determines the desires of the agent on the basis of a set of motives. Few other approaches to the integration of motivation in agents exist [14, 170, 180] and none combines motivation with BDI agents and know-how in a general framework. The know-how component represents the procedural knowledge of the agent that it uses to determine its course of action. The know-how component can be seen as a form of a hierarchical plan library that is often used in BDI agents [248, 51]. The difference of the know-how approach [215, 214] is that it aims to represent the structural knowledge on a logical basis, on the same level as the *know-that* of the agent, to facilitate reasoning and change operations on both in combination. We define simplified instances of our approaches to motivation [159] and know-how [157, 229]. We instantiate the BDI$^+$ model with answer set programming (ASP) [112] as an exemplary non-monotonic knowledge representation formalism and use it to develop secrecy aware agents.

1.3.2 *Non-monotonic reasoning formalisms for epistemic agents*

Based on our epistemic agent framework we introduce epistemic components as components of an epistemic state. Epistemic components are used by a belief operator to form a belief set. We define a general notion of belief operator families whose operators can be ordered, e. g., based on how credulous the inference behavior is. The assumptions we make are general enough to capture a big variety of knowledge representation formalisms and in particular those for which effective solvers are available. We define desirable properties for belief operators and ordered families of these. Further, we consider more concrete belief operators on the basis of the theory of formalisms based on extension families from [173], and extend it towards families of operators with credulity order. Finally, we instantiate and illustrate the concepts for answer set programming.

1.3.3 *Belief change operators and non-monotonic reasoning*

We integrate approaches from belief change theory in our epistemic agent model. Thereby, we also contribute to the field of belief change theory. We build on the theory of belief base revision [122, 127, 128] and instantiate it with and for non-monotonic reasoning formalisms in two main aspects. Firstly, we consider selective revision operators [96]. These decide in which form the new information shall be accepted by means of a selection function first, and then revise the belief base with the result by means of a prioritized change operator, i. e., an operator that always accepts the new information. We extend this approach for multiple revision operations and show formally that and how the transformation function can be instantiated using the framework of deductive argumentation [27].

Secondly, we apply the belief base change theory to non-monotonic logics by example of ASP. We analyze base revision postulates for non-monotonic belief bases and adapt them adequately. We consider ASP specific change postulates from the literature, generalize them for different notions of equivalence, analyze, and adapt these to augment our set of desirable postulates of a base revision operator for ASP. We formally show the relation of base revision postulates to the ASP specific postulates. Furthermore, we develop a generalized construction for base revision operators based on a new screened consolidation operation. We prove a representation theorem for this construction. Further we show that the resulting multiple base revision operator satisfies all desirable properties we proposed.

1.3.4 *Agent-based secrecy and secrecy aware agents*

We approach the topic of secrecy in multiagent systems from the subjective perspective of communicating epistemic agents. We make use of our general epistemic agent framework to define a new, subjective notion of secrecy. For agents with incomplete information and bounded rationality in a dynamic environment the strict preservation of secrecy is not always possible. In case the agent violates secrecy, it should still do its best to violate secrecy as little as possible. We call agents *secrecy aware* if they take secrecy into consideration as good as possible, and those agents of these that can be guaranteed to not violate secrecy *secrecy preserving*.

We define a secrecy agent model by augmenting the epistemic state of an agent by views on the information that is available to other agents, and by a representation of its secrets. Hereby we consider a symmetric system, which means that all agents have views on their fellow agents and all agents have their individual secrets. Further, we consider the uncertainty of such views in two dimension, the uncertainty of the information available to another agent and the uncertainty of the view on this information.

We define a notion of secrets with different levels of strength based on the belief operator families we develop. A secret formula is protected against inference by use of a particular belief operator from a belief operator family. The operators of a belief operator family are ordered, e. g., with respect to their credulity. A secret formula is protected the more strongly, the more credulous the belief operator is with respect to which it shall be protected.

Moreover, we consider a change operator for epistemic states and we define desirable properties with relation to secrecy. This includes the changes of the beliefs of the agent itself, its views on other agents, and its secrets. We develop a change operator for secrets that is capable to minimally weaken secrets if they cannot be protect with the declared strength.

We formally prove for several settings which properties of the behavior of an agent are necessary and sufficient to guarantee that it preserves secrecy. We develop a game theory based notion of epistemic states in which an attacking agent does not have a winning strategy and show that the maintenance of such a state is necessary and sufficient to preserve secrecy. We consider the evaluation of the agent's options for action with respect to secrecy, and with respect to the informativity of its actions. We define five principles on how an agent should use its available information to classify a set of actions with respect to secrecy as fine grained as possible, while still justified. Moreover, we develop an algorithm to classify actions and prove that it satisfies all principles.

On the basis of the abstract model and joining the developed aspects of secrecy described above, we define secrecy aware agents as an extension of our BDI$^+$ agent model. We instantiate this model with ASP for knowledge representation and develop ASP representations of speech acts on the information and meta-information level. Further, we formalize ASP based patterns of meta-inferences of potential attacking agents. We show that and how we can use this *secrecy ASP BDI$^+$ agents* to construct a two-agent system that simulates our *strike committee meeting* example.

1.3.5 *Epistemic agent programming framework* Angerona

We present the Angerona framework for the implementation of epistemic agents with a strong focus on flexibility, extensibility and compatibility with diverse formalisms for non-monotonic reasoning and with diverse agent models. It implements the general epistemic agent framework developed in this thesis. In contrast to other multiagent system frameworks it has no fixed agent model, but follows a flexible plug-in architecture and facilitates the use of diverse (non-monotonic) knowledge representation formalisms and belief change operators. Different kinds of knowledge representation formalisms can be used within one agent. Different agents can be based on different agent architectures and can each use different knowledge representation formalisms as long as they share a common communication language. The knowledge representation plug-ins of ANGERONA use the interfaces of the TWEETY *library* for knowledge representation [227]. The belief operators and belief change operators developed in this thesis are implemented in the TWEETY library. The BDI^+ agent model developed in this thesis is implemented as a sub-framework that can be used and extended based on operator plug-ins. ANGERONA also contains default and extended operators to instantiate the BDI^+ model. The *secrecy ASP BDI*$^+$ model and several simulations based on it are also implemented. Furthermore, several other approaches are available in ANGERONA, e. g., [75, 76, 152, 131, 244, 4, 229, 157, 159]. ANGERONA features an environment plug-in for communicating agents and a versatile GUI to monitor the simulation and the agents, including the dynamics of their epistemic states. The ANGERONA framework as well as TWEETY are open source[2].

1.4 OUTLINE

The remainder of this thesis is organized as follows. In Chapter 2 we present the necessary background for the following chapters. In Chapter 3 we develop the general framework of epistemic agents. We start by laying the foundations for the consideration of communicating agents and agents based on logical knowledge representation formalisms. By use of these foundations we then define our notion of epistemic agents. On the basis of the general model we define the concept of compound agent models and formalize our BDI^+ agent model as an instance of the general compound model. Based on this model we define basic instances of the operators of the BDI^+ model. Then we go into more detail on the belief operators of agents and define belief operator families, discuss properties of these and give exemplary instantiations.

In Chapter 4 we first develop the general structure of belief change operations for epistemic agents. Then we elaborate on selective revision operators, extend the selective revision framework to multiple revision operations and develop selection functions that are based on deductive argumentation. Afterwards we elaborate on the base revision approach for non-monotonic belief bases in general and ASP in particular.

In Chapter 5 we develop our subjective, agent-based notion of secrecy on the basis of our epistemic agent framework. We first introduce a secrecy agent model and elaborate various aspects of agent

2 `https://github.com/Angerona, http://sourceforge.net/projects/tweety/`

models with respect to secrecy. Then we show formally for different settings which behavior of an agent guarantees the preservation of secrecy. Next we develop properties for the classification of actions with respect to secrecy and devise an algorithm that adheres to these principles. Afterwards we build on the BDI^+ agent model developed in Chapter 3 to construct complete agents for secrecy preservation and elaborate the instantiation of these by use of ASP. We develop ASP representations for meta-information and meta-inferences and model the *strike committee meeting* example from the introduction.

In Chapter 6 we present the multiagent programming framework Angerona, which is based on the general agent model introduced in Chapter 3. We describe the main ideas of the framework and how the instances based on the results of Chapters 4 and 5 are realized. In Chapter 7 we summarize and discuss the contributions of this thesis.

1.5 PUBLICATIONS AND CONTRIBUTIONS

Some ideas and parts of this thesis are based on previously published work. In the following, I first specify which parts of this thesis are based on previously published work. Then, I specify the contributions of my co-authors in the respective works.

1.5.1 *Publications that contributed to this thesis*

First versions of the epistemic agent model for secrecy, belief operator families, and the definition of agent-based secrecy have been published in [155]. All of these concepts have been substantially revised and extended. They contributed in particular to parts of Section 3.3 and Sections 5.1 to 5.3. A first version of the consideration of secrecy aware BDI agents based on answer set programming has been published in [156]. It has been substantially revised and extended, and contributed in particular to Section 5.7.

Our work on extensions of the BDI model by motivation and know-how have been published before. In this thesis we develop a new extended BDI model on the basis of simplified versions of the published approaches to motivation and know-how. The work on motivation for BDI agents has been published in [159]. In this thesis we define a simplified version of the full approach developed in [159] in Section 3.6. The work on know-how has been published in [229] and [157]. As for motivation we define and use a basic version of the full know-how approach in this thesis, we do this in Section 3.6. The consideration of selective revision by means of argumentation in Section 4.2 is based on the Publication [160]. The belief base revision approach to ASP change operations presented in Section 4.3 is based on a revised version of the Publication [154]. Sections 5.5.2 and 5.5.3

are based on revised parts of the work published in [41]. Chapter 6 is based on extensions of parts of the Publication [161].

1.5.2 *Contributions to the publications*

The publications [154, 156, 155] are the result of my own work. They have been produced under the supervision of my supervisor Gabriele Kern-Isberner. As such she proofread drafts of the articles, discussed them with me and suggested further research questions.

The works [160, 161, 41] are the result of collaborations with further research partners within two research projects in which I participated. In the following I clarify the contributions of the respective authors to these publications.

The work [41] was carried out as part of the Project A5 "Exchange and Fusion of Information under Availability and Confidentiality Requirements in MultiAgent Systems" of the Collaborative Research Center SFB 876. The project leaders are Gabriele Kern-Isberner and Joachim Biskup. As such, the general topic, the preparatory work and project proposal are based on the work of them. This particular work originates from the elaboration of the uncertainty of the defending agent about the state of the attacking agent, which was mentioned briefly by myself in [155]. Starting from there, Cornelia Tadros and myself elaborated the two resulting dimensions of uncertainty, the uncertainty of the attacking agent in its view on the world and the defending agent's uncertainty in its view on the uncertain world view of the attacking agent. We worked on formalizing belief operators that deal with both dimensions of uncertainty, and on families of such operators that differ in their credulity in both dimensions. Then, we searched for a principled representation of the relation of the credulity of a belief operator in both dimensions and the degree of protection of a secret that is formulated by use of this operator. We had a set of basic principles in mind we wanted to formulate. We discussed our preliminary findings several times with Gabriele Kern-Isberner and Joachim Biskup. These discussions helped us identifying issues in our considerations and pointed us to further directions of research. As one result we tightened our definitions of belief operators.

We continued with a more general research question. Given the uncertainty of the defending agent, it cannot be excluded that there are situations in which the agent cannot prevent the violation of a secret. Our new goal was to find a principled approach to classify actions in our considered setting. Cornelia Tadros looked into concepts from works on normative reasoning that deal with violations of norms. In our setting, secrecy can be seen as a norm that is induced by the secrecy constraints and a belief operator family with credulity order. We connected our consideration of secrecy and action selection to the notion of betterness relations on actions in normative reasoning. This led us to the specification of a classification of actions with respect to secrecy.

I found out that the basic principles we were heading for from the beginning cannot hold in general in our considered setting. To find a solution, I considered the problem from a different perspective, considering the information that is available to the defending agent to classify possible actions and considering how this information can be used for the classification task. As a result, I developed the notion of violation sets and two new principles for the classification of actions (Principle I.1 and I.2) with respect to secrecy. Moreover, Cornelia Tadros came up with the idea of a minimality principle inspired by the rational closure for conditional knowledge bases, which resulted in Principle II. With these three principles we found a complete set of principles for the characterization of the classification task.

We further refined Principle I.2 as a result of discussions of Cornelia Tadros, Joachim Biskup and myself on the treatment of cyclic dependencies with respect to Principle II. Reconsidering our initial idea of the basic principles, I looked for weaker versions of these that are satisfiable in our setting. As a result, I came up with the Principles III and IV. Further, I sketched a proof that shows that the satisfaction of Principles III and IV follows from the satisfaction of Principles I.1, I.2 and II (Proposition 5.5.12). This sketch was then worked out by Cornelia Tadros and myself. It was Joachim Biskup who pointed out that we should show that the classification we characterized by use of the principles can be operationalized. He proposed the focused scenario we consider in the article, that is, the consideration of inform actions for a fixed point in time by a secrecy reasoner. He also proposed the use of the concrete and simple change operator.

The search for an operationalization resulted in the definition of a secrecy reasoner and the design of an algorithm to implement it. The first version of the algorithm of the secrecy reasoner was developed by myself. This algorithm was then further refined in discussions of Cornelia Tadros, Joachim Biskup and myself. Together we also devised the desired propositions with respect to the algorithm: that it always terminates, and returns a complete classification (Proposition 5.5.15); and that it satisfies all principles (Proposition 5.5.16). The proof of Proposition 5.5.15 and the part of the satisfaction of Principles I.1 and I.2 in the proof of Proposition 5.5.16 is my work. For the satisfaction of Principle II of Proposition 5.5.16 the first draft of the proof was my work, which was then worked out in detail by Cornelia Tadros and Joachim Biskup. Preliminary versions of the article [41] were proofread by Gabriele Kern-Isberner and Joachim Biskup. Joachim Biskup also wrote parts of the introduction and conclusion, and helped Cornelia Tadros and myself with the clear formulation of our ideas.

In this thesis the approach and results of the Sections 3 and 4 of the article [41] are integrated in the framework developed in the thesis to support the agent's action selection. To this end, the sections have been revised and adapted to the setting and notation of the thesis.

The Angerona framework that is described in the Publication [161] was developed within the Project A5 "Exchange and Fusion of Information under Availability and Confidentiality Requirements in MultiAgent Systems" of the Collaborative Research Center SFB 876. It was implemented mainly by Tim Janus under my supervision as part of his work in the A5 Project. He also participated in discussions about the conceptualization of the implementation. Several additional approaches that are implemented in the Angerona framework are the result of final theses under my supervision, these are [3, 137, 19, 52, 131, 45]. Moreover, Angerona was used and extended by Daniel Dilger under the supervision of Cornelia Tadros and myself in an internship financed by the German Academic Exchange Service (DAAD); a report on this work can be found in [78]. Angerona was further extended by Pia Wierzoch, Stefan Rötner and Sebastian Homann as part of their work in the A5 Project under the supervision of Cornelia Tadros and myself.

The work on [160] was carried out within the project "Combining belief revision and argumentation for enhancing the reasoning capabilities of agents in multiagent systems", supported by the German Academic Exchange Service DAAD. The main idea of the Publication [160] emerged from discussions of the authors of the article. In these discussions various ways to use argumentation in the belief revision process were considered. The idea to use argumentation to decide if and which parts of the input shall be accepted emerged. Looking at the belief revision literature I found out, that the selective revision framework constitutes a very good basis for our work. Matthias Thimm looked at the deductive argumentation framework. In close collaboration Matthias Thimm and myself then extended the selective revision framework and determined the appropriate properties for the use in combination with an argumentation approach. We devised new postulates for the selective revision framework and properties of the categorizer and accumulator. Then we showed the formal results of the publication. The examples result from the collaboration of Alejandro Garcia and myself. All authors proofread the publication.

The work on know-how in [229] and [157] is the result of a close collaboration of Matthias Thimm and myself. The work on motivation is based on the Diploma thesis of Regina Fritsch under the supervision of Gabriele Kern-Isberner and myself. The original approach has been thoroughly revised in collaboration of Matthias Thimm, Gabriele Kern-Isberner and myself which resulted in the Publication [159].

2 BACKGROUND

In this chapter we present an overview of important background for the following chapters. In Section 2.1 we start with an introduction to the theory of agents and in particular to abstract agent models and to the BDI agent architecture. Then, we introduce some knowledge representation formalisms that might be used by the agents we develop in this thesis. Propositional logic is introduced in Section 2.2, deductive argumentation in Section 2.3 and answer set programming in Section 2.4. In Section 2.5 we give a summary of the basics of belief revision theory.

2.1 AGENT MODELS

The research on intelligent agents started in the 1980s and gained momentum in the 1990s. Today, the field is well developed on diverse lines of research from the purely theoretical works to actually implemented systems and applications in a variety of domains [241, 248].

An agent is an entity which is capable to act autonomously towards its goals in its environment. The environment of an agent is usually given by a multiagent system, which consists of a set of agents that interact with each other and some representation of a physical environment that the agent can observe and manipulate partially. Hence the agent continuously receives new information from its dynamic environment. The agent has to be able to adapt and react to these changes while it pursues the realization of its goals. In this scenario a plethora of challenges for the design of such an agent arises.

For the conceptualization of intelligent agents two fundamental approaches can be distinguished: the formal modeling of the behavior of agents and multiagent systems; and the constructive, decompositional, approach leading to concrete agent architectures and systems There is a big gap between both, the formal models can often not be implemented in any direct way, and the concrete architectures often lack a theoretical basis. In the next two sections we describe two typical representatives of both approaches, first an abstract notion of agents and then a compositional approach.

2.1.1 *Abstract notion of agents*

We introduce common concepts of abstract agents based on the presentation in [248]. On the abstract level *systems* $(\mathcal{X}, \mathsf{Env})$ are formalized that consist of an agent $\mathcal{X}$ and an environment Env. Formally, the environment is modeled by a set of discrete environment states $E = \{e, e', \dots\}$. The environment state is transformed by the actions of an agent from the set of possible actions $Ac = \{\alpha, \alpha', \dots\}$. A *run* is a sequence of environment states, interleaved by actions, as follows:

$$\mathsf{run} : e_0 \xrightarrow{\alpha_0} e_1 \xrightarrow{\alpha_1} e_2 \xrightarrow{\alpha_2} e_3 \xrightarrow{\alpha_3} \dots \xrightarrow{\alpha_{n-1}} e_n$$

The set of finite runs given the sets E and Ac is denoted by $\mathcal{R}$, the subset of runs that end with an action is denoted by $\mathcal{R}^{Ac}$ and the subset of runs that end with an environment state with $\mathcal{R}^E$. An agent is a function that maps runs ending with an environment state to an action:

$$\mathcal{X} : \mathcal{R}^E \to Ac$$

The effects of actions on the environment are modeled by a *state transformer function*

$$\mathsf{tr} : \mathcal{R}^{Ac} \to \mathcal{P}(E).$$

Hereby $\mathcal{P}(S)$ denotes the *power set* of a set S, i. e. the set of all subsets of S. This formalization implies that agents as well as environments are history dependent since the next state/action is dependent on the entire run and not only on the last action/state. The transformer function returns a set of environment states, which models that the effect of the agent is non-deterministic. The agent on the other hand is assumed to be deterministic, i e. the corresponding function returns a single action.

An environment Env is a triple $\mathsf{Env} = \langle E, e_0, \mathsf{tr} \rangle$ with a set of environment states E, an initial environment state $e_0 \in E$ and a state transformer function tr. For a system $(\mathcal{X}, \mathsf{Env})$ its set of possible runs $\mathcal{R}(\mathcal{X}, \mathsf{Env})$ is the set of runs of agent $\mathcal{X}$ in the environment Env. A *run* of agent $\mathcal{X}$ in the environment $\mathsf{Env} = \langle E, e_0, \mathsf{tr} \rangle$ is a sequence

$$(e_0, \alpha_0, e_1, \alpha_1, e_2, \alpha_2, \dots)$$

such that $\mathcal{X}(e_0) = \alpha_0$ and for all $i > 0$

$$e_i \in \mathsf{tr}((e_0, \alpha_0, e_1, \alpha_1, \dots, e_{i-1}, \alpha_{i-1})) \text{ and } \alpha_i = \mathcal{X}((e_0, \alpha_0, \dots, e_{i-1})).$$

Hence in this formalization of abstract agents the behavior of an agent is represented with respect to an environment by its set of possible runs. This allows to formalize and prove abstract properties of agents in environments. However, it does not help with the conceptualization and realization of agents. In the next section we present the most popular conceptualization of rational agents.

2.1.2 *The BDI Model*

There are several general approaches for realizing intelligent agents, for an overview we refer to [248]. The most popular model for the realization of intelligent agents is the *BDI model*. It distinguishes between *Beliefs*, *Desires*, and *Intentions* as the main components of an agent's state, the interactions of which determine its behavior.

This model originates from the work of the philosopher Michael E. Bratman [56] on practical reasoning and the role of intentions. This led to the BDI model first described by Bratman, Israel and Pollack in [57]. The BDI-model has been adopted by many others. Besides different versions for the general BDI architecture several frameworks for agent programming are based on the BDI paradigm. The first implemented and used system was the procedural reasoning system (PRS) [117]. It was used for fault detection in the NASA space shuttle Discovery and was later further developed and applied in various scenarios [117, 118]. It also forms the basis for current multiagent programming platforms [51]. On the analytical side, there has also been substantial work on developing logics to analyze BDI based systems [200, 196, 201, 194, 198, 197, 199].

In the following we present an overview of the original BDI architecture, on the basis of [57] and [247]. In this model the *Beliefs* represent the (plausible) beliefs the agent has concerning the current situation as well as its background information. The *Desires* represent what the agent desires to achieve in general but to which it has not committed, i. e., possible goals. *Intentions* are goals the agent has committed to, they are consecutively refined by the agent and lead to the next action the agent performs to achieve its current goals. Formally, sets of all possible sets of beliefs $\mathcal{L}_{\mathfrak{B}}$, desires $\mathcal{L}_{\mathfrak{D}}$ and intentions $\mathcal{L}_{\mathfrak{I}}$ are defined, which are often formulae of a logical language. As already indicated by the symbols we also refer to,e. g., the set of possible beliefs, as a language. The current state of a BDI agent is then characterized by the triple $(\mathfrak{B}, \mathfrak{D}, \mathfrak{I})$ of its currently held *beliefs* $\mathfrak{B} \in \mathcal{L}_{\mathfrak{B}}$, *desires* $\mathfrak{D} \in \mathcal{L}_{\mathfrak{D}}$ and *intentions* $\mathfrak{I} \in \mathcal{L}_{\mathfrak{I}}$. The behavior of the agent is determined by its process of practical reasoning which is based on its mental state (its beliefs, desires, and intentions) and the percepts it receives. This process is realized by an agent cycle as depicted in Figure 2.1.1. The current beliefs of the agent are revised on the basis of the current percept $p \in \mathsf{Per}$ by the *belief revision function*

$$\mathsf{brf} : \mathcal{L}_{\mathfrak{B}} \times \mathsf{Per} \to \mathcal{L}_{\mathfrak{B}}.$$

On the basis of the revised belief and the current set of intentions the current options, i. e., the desires of the agent are determined by the function

$$\mathsf{options} : \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{I}} \to \mathcal{L}_{\mathfrak{D}}.$$

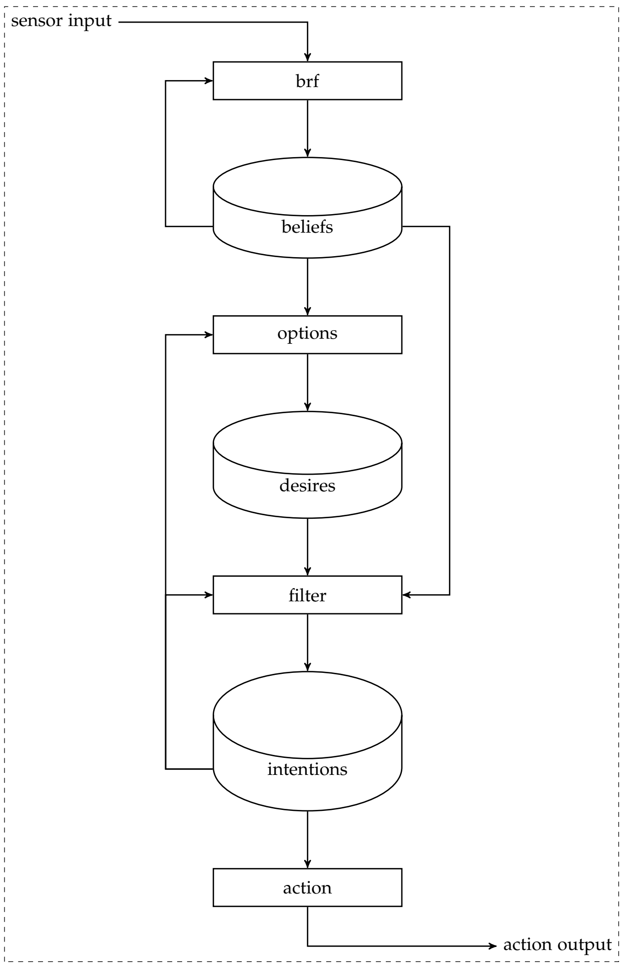


Figure 2.1.1: Classic BDI agent cycle (adopted from [247])

The *options function* generates the options to refine the currently held intentions of the agent in the light of its current beliefs. Recursively applied, the options function determines sub-goals for realizing some intention and then sub-goals for the newly adopted sub-goals and so on. This way the means-ends-reasoning of the agent is realized. The agent's deliberation process, the process of choosing what to do, is realized by the *filter function*

$$\mathsf{filter} : \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{D}} \times \mathcal{L}_{\mathfrak{I}} \to \mathcal{L}_{\mathfrak{I}}.$$

This function selects some of the current desires as new intentions and removes some of the former intentions of the agent to form the current set of intentions. Intentions are to be removed if they are successfully achieved or if they do not seem to be achievable at all or not at a reasonable cost. Note that therefore the new set of intentions is a subset of the union of the old intentions and the current desires. At last, the *action function*

$$\mathsf{act} : \mathcal{L}_{\mathfrak{I}} \to \mathsf{Act}$$

returns an action on the basis of a set of intentions. An intention is called an *atomic intention* if it can be directly satisfied by a single action.

The general BDI architecture defines an intuitive decomposition of the mental state and the functionality of an intelligent agent. However, it just describes the general architecture of an agent and leaves the further instantiation and solution of problems involved open. The belief revision function of a BDI agent or desirable properties of it are not specified or discussed in the original BDI literature. The means-ends reasoning process needs some form of a plan library to determine the sub-intentions for a given intention. It is also necessary to structure the set of intentions by some ordering, for example a stack, or a list of stacks. Further, the strength of commitment to an intention is an issue. The agent has to be committed to achieve its goals but it also has to reconsider its intentions and realize when it is worth dropping its current intention for the sake of other intentions or when they become unachievable. In most domains the options in the deliberation process of an agent can, and should, be compared based on some notion of utility. For more details on these problems see e. g.[248, 241].

We want to consider agents that use logical languages for knowledge representation and reasoning. We introduce propositional logic, which forms the basis for many other knowledge representation formalisms as well, in the next section.

2.2 PROPOSITIONAL LOGIC

Let At be a propositional signature, i. e. a set of propositional atoms. Let $\mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ be the corresponding propositional language generated by the atoms in At and the connectives $\wedge$ (*and*), $\vee$ (*or*), $\Rightarrow$ (*implication*), and $\neg$ (*negation*). To simplify the presentation, we assume some arbitrary total order $\succ$ on the elements of $\mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ which is used to enumerate elements of each finite $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ in a unique way, cf. [26]. For a finite subset $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ the *canonical enumeration* of Φ is the vector $\langle \phi_1, \ldots, \phi_n \rangle$ such that $\{\phi_1, \ldots, \phi_n\} = \Phi$ and $\phi_i \succ \phi_j$ for every $i < j$ with $i, j = 1, \ldots, n$. As $\succ$ is total the canonical enumeration of every finite subset $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ is uniquely defined.

We use the operator $\vdash$ to denote classical entailment, i. e., for sets of propositional sentences $\Phi_1, \Phi_2 \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ we say that Φ_2 *follows* from Φ_1, denoted by $\Phi_1 \vdash \Phi_2$, if and only if Φ_2 is entailed by Φ_1 in the classical logical sense. For sentences $\phi, \phi' \in \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ we write $\phi \vdash \phi'$ instead of $\{\phi\} \vdash \{\phi'\}$. We define the *deductive closure* $Cn^{\mathsf{prop}}(\cdot)$ of a set of sentences Φ as

$$Cn^{\mathsf{prop}}(\Phi) = \{\phi \in \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}} \mid \Phi \vdash \phi\}.$$

If $\Phi \vdash \perp$ we say that Φ is *inconsistent*. Two sets of sentences $\Phi, \Phi' \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ are *equivalent*, denoted by $\Phi \equiv^{\mathsf{p}} \Phi'$, if and only if $\Phi \vdash \Phi'$ and $\Phi' \vdash \Phi$.

We also use the equivalence relation $\cong^{\mathsf{p}}$ which is defined as $\Phi \cong^{\mathsf{p}} \Phi'$ if and only if there is a bijection $\sigma : \Phi \to \Phi'$ such that for every $\phi \in \Phi$ it holds that $\phi \equiv^{\mathsf{p}} \sigma(\phi)$. This means that $\Phi \cong^{\mathsf{p}} \Phi'$ if Φ and Φ' are *element-wise* equivalent. Note that $\Phi \cong^{\mathsf{p}} \Phi'$ implies $\Phi \equiv^{\mathsf{p}} \Phi'$ but not vice versa. In particular, it holds that e. g. $\{a \wedge b\} \equiv^{\mathsf{p}} \{a, b\}$ but $\{a \wedge b\} \not\cong^{\mathsf{p}} \{a, b\}$. For sentences $\phi, \phi' \in \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ we write $\phi \equiv \phi'$ instead of $\{\phi\} \equiv \{\phi'\}$ if $\equiv \in \{\equiv^{\mathsf{p}}, \cong^{\mathsf{p}}\}$.

For a set S let $\mathfrak{P}(S)$ denote the *set of multi-sets* of S, i. e. the set of all subsets of S where an element may occur more than once. To distinguish sets from multi-sets we use brackets "$\langle$" and "$\rangle$" for the latter.

Propositional logic forms the basis for the deductive argumentation approach that we present in the next section.

2.3 DEDUCTIVE ARGUMENTATION

Argumentation frameworks [24] allow for reasoning with inconsistent information based on the notions of arguments, counterarguments and their relationships. Since the seminal paper of Phan Minh Dung [83] interest has grown in research in computational models for argumentation. In this thesis we use the framework of *deductive argumentation* as proposed by Besnard and Hunter [26] to construct selection functions that decide which part of the input of a belief change operator should be accepted. The central notion of the framework of deductive argumentation is the notion of an *argument*.

Definition 2.3.1 (Argument). Let $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ be a set of sentences. An *argument* A for a sentence $\alpha \in \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ in Φ is a tuple $A = \langle \Psi, \alpha \rangle$ with $\Psi \subseteq \Phi$ that satisfies

1. $\Psi \nvdash \perp$,
2. $\Psi \vdash \alpha$, and
3. there is no $\Psi' \subsetneq \Psi$ with $\Psi' \vdash \alpha$.

For an argument $A = \langle \Psi, \alpha \rangle$ we say that α is the *claim* of A and Ψ is the *support* of A.

Thus, in an argument $A = \langle \Psi, \alpha \rangle$ for α, the set Ψ is a minimal set entailing α. Given a set $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ of sentences there may be multiple arguments for α. As in [26] we are interested in arguments that are most conservative.

Definition 2.3.2 (Conservativeness). An argument $A = \langle \Psi, \alpha \rangle$ is more *conservative* than an argument $B = \langle \Phi, \beta \rangle$ if and only if $\Psi \subseteq \Phi$ and $\beta \vdash \alpha$.

In other words, an argument A is more conservative than an argument B if B has an at least as large support (with respect to set inclusion) and a more general conclusion than A. An argument A is *strictly more conservative* than an argument B if and only if A is more conservative than B but B is not more conservative than A. If $\Phi \subseteq \mathcal{L}_{\mathsf{At}}^{\mathsf{prop}}$ is inconsistent there are arguments with contradictory claims.

Definition 2.3.3 (Undercut). An argument $A = \langle \Psi, \alpha \rangle$ is an *undercut* for an argument $B = \langle \Phi, \beta \rangle$ if and only if $\alpha = \neg(\phi_1 \wedge \ldots \wedge \phi_n)$ for some $\phi_1, \ldots, \phi_n \subseteq \Phi$.

If A is an undercut for B then we also say that A *attacks* B. In order to consider only those undercuts for an argument that are most general we restrain the notion of undercut as follows.

Definition 2.3.4 (Maximally conservative undercut). An argument $A = \langle \Psi, \alpha \rangle$ is a *maximally conservative undercut* for an argument $B = \langle \Phi, \beta \rangle$ if and only if A is an undercut of B and there is no undercut A' for B that is strictly more conservative than A.

Definition 2.3.5 (Canonical undercut). An argument $A = \langle \Psi, \neg(\phi_1 \wedge \ldots \wedge \phi_n) \rangle$ is a *canonical undercut* for an argument $B = \langle \Phi, \beta \rangle$ if and only if A is a maximally conservative undercut for B and $\langle \phi_1, \ldots, \phi_n \rangle$ is the canonical enumeration of Φ.

It can be shown that it suffices to consider only the canonical undercuts for an argument in order to come up with a reasonable argumentative evaluation of some claim α [26]. Having an undercut B for an argument A there may also be an undercut C for B which *defends* A. In order to give a proper evaluation of some argument A we have to consider all undercuts for its undercuts as well, and so on. This leads to the notion of an *argument tree*.

Definition 2.3.6 (Argument tree). Let $\alpha \in \mathcal{L}_{\mathsf{At}}^{\mathsf{prop}}$ be some sentence and let $\Phi \subseteq \mathcal{L}_{\mathsf{At}}^{\mathsf{prop}}$ be a set of sentences. An *argument tree* $\tau_\Phi(\alpha)$ for α in Φ is a tree where the nodes are arguments and that satisfies

1. the root is an argument for α in Φ,
2. for every path $[\langle \Phi_1, \alpha_1 \rangle, \ldots, \langle \Phi_n, \alpha_n \rangle]$ in $\tau_\Phi(\alpha)$ it holds that $\Phi_n \not\subseteq \Phi_1 \cup \ldots \cup \Phi_{n-1}$, and
3. the children $B_1, \ldots, B_m$ of a node A consist of all canonical undercuts for A such that condition (2) above is not violated when these canonical undercuts are added as children.

Let $\mathcal{F}(\mathsf{At})$ be the set of all argument trees.

An argument tree is a concise representation of the relationships between different arguments that favor or reject some argument $\mathcal{A}$. In order to evaluate whether a claim α can be justified we have to consider all argument trees for α and all argument trees for $\neg\alpha$. For an argument tree τ let $\mathsf{root}(\tau)$ denote the root node of τ. Furthermore, for a node $\mathcal{A} \in \tau$ let $\mathsf{ch}_\tau(\mathcal{A})$ denote the children of $\mathcal{A}$ in τ and $\mathsf{ch}^{\mathcal{T}}_\tau(\mathcal{A})$ denote the set of sub-trees rooted at a child of $\mathcal{A}$.

Definition 2.3.7 (Argument structure). Let $\alpha \in \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ be some sentence and let $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ be a set of sentences. The *argument structure* $\Gamma_\Phi(\alpha)$ for α with respect to Φ is the tuple $\Gamma_\Phi(\alpha) = (\mathcal{T}^+, \mathcal{T}^-)$ such that $\mathcal{T}^+$ is the set of argument trees for α in Φ and $\mathcal{T}^-$ is the set of arguments trees for $\neg\alpha$ in Φ.

The argument structure $\Gamma_\Phi(\alpha)$ of a $\alpha \in \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ gives a complete picture of the reasons for and against α. The argument structure has to be evaluated in order to determine the status of sentences. We introduce the powerful evaluation mechanisms from [26] and give examples of how adequate and simple instantiations can be realized.

Definition 2.3.8 (Categorizer). A *categorizer* γ is a function $\gamma : \mathcal{F}(\mathsf{At}) \to \mathbb{R}$.

A categorizer is meant to assign a value to an argument tree τ depending on how strongly this argument tree favors the root argument. In particular, the larger the value of $\gamma(\tau)$ the better the justification of believing in the claim of the root argument. For an argument structure $\Gamma_\Phi(\alpha) = (\{\tau^+_1, \ldots, \tau^+_n\}, \{\tau^-_1, \ldots, \tau^-_m\})$ and a categorizer γ we abbreviate

$$\gamma(\Gamma_\Phi(\alpha)) = (\langle\gamma(\tau^+_1), \ldots, \gamma(\tau^+_n)\rangle, \langle\gamma(\tau^-_1), \ldots, \gamma(\tau^-_m)\rangle)\,.$$

Definition 2.3.9 (Accumulator). An *accumulator* κ is a function $\kappa : \mathfrak{P}(\mathbb{R}) \times \mathfrak{P}(\mathbb{R}) \to \mathbb{R}$.

An accumulator is meant to evaluate the categorization of argument trees for or against some sentence α.

Definition 2.3.10 (Acceptance). We say that a set of sentences $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ *accepts* a sentence α with respect to a categorizer γ and an accumulator κ, denoted by

$$\Phi \mathrel{|\!\!\sim}_{\kappa,\gamma} \alpha \text{ if and and only if } \kappa(\gamma(\Gamma_\Phi(\alpha))) > 0$$

If Φ does not accept α with respect to γ and κ ($\Phi \mathrel{|\!\!\not\sim}_{\kappa,\gamma} \alpha$) we say that Φ *rejects* α with respect to γ and κ.

Simple instances of categorizers and accumulators are defined in the following example.

Example 2.3.11. Let τ be some argument tree. The classical evaluation of an argument tree, as ,e. g. employed in *Defeasible Logic Programming* [105], is that each leaf of the tree is considered "undefeated". An inner node is "undefeated" if all its children are "defeated". And an inner node is "defeated" if there is at least one child that is "undefeated". This intuition can be formalized by defining the *classical categorizer* γ_0 recursively via

$$\gamma_0(\tau) = \begin{cases} 1 & \mathsf{ch}_\tau(\mathsf{root}(\tau)) = \emptyset \\ 1 - \max\{\gamma_0(\tau') \mid \tau' \in \mathsf{ch}_\tau^{\mathcal{T}}(\mathsf{root}(\tau))\} & \text{otherwise} \end{cases}$$

Furthermore, a simple accumulator κ_0 can be defined via

$$\kappa_0(\langle N_1, \ldots, N_n \rangle, \langle M_1, \ldots, M_m \rangle) = \\ N_1 + \ldots + N_n - M_1 - \ldots - M_m\,.$$

For example, a set of sentences $\Phi \subseteq \mathcal{L}_{At}^{\mathsf{prop}}$ accepts a sentence α with respect to γ_0 and κ_0 if and only if there are more argument trees for α where the root argument is undefeated than argument trees for $\neg\alpha$ where the root argument is undefeated. ♢

More examples of categorizers and accumulators can be found in [26]. Using those notions we are able to state for every sentence $\phi \in \Phi$ whether ϕ is accepted in Φ or not, depending on the arguments that favor α and those that reject α.

In the next section we introduce a logic programming formalism, which is another non-monotonic reasoning formalism. It has close connections to argumentation theory, as was already shown in [83].

2.4 ANSWER SET PROGRAMMING

Answer set programming is based on the stable model semantics for logic programs. Different language versions are used, the basis are normal logic programs that feature default negation [113], extended logic programs feature strict and default negation[114], and disjunctive logic programs [112] feature disjunctions in the head of a rule. In this work we focus on extended logic programs under the answer set semantics based on [114].

Extended logic programs consist of rules over a set of atoms At using strong negation $\neg$ and default negation not. An atom is of the form $p(t_1, \ldots, t_n)$ with p being a predicate of arity n and $t_1, \ldots, t_n$ are terms. A *term* is a *variable* or a *constant*, no functions are allowed in standard answer set programming. We denote constants by lower case strings, and variables by strings that start with an upper case letter. A literal L can be an atom A or a negated atom $\neg A$. The complement of a literal L is denoted by $\neg L$ and is

– A if and only if $L = \neg A$ and

– $\neg A$ if and only if $L = A$.

Let At be the set of all atoms and Lit the set of all literals

$$\mathsf{Lit} = At \cup \{\neg A \mid A \in At\}.$$

The set $\mathsf{Lit}_{\mathsf{not}} = \{\mathsf{not}\ L \mid L \in \mathsf{Lit}\}$ denotes the set of all default negated literals. The set of all literals and default negated literals is denoted by $\mathfrak{L} = \mathsf{Lit} \cup \mathsf{Lit}_{\mathsf{not}}$. A rule r is written as

$$L \leftarrow L_0, \ldots, L_m, \mathsf{not}\ L_{m+1}, \ldots, \mathsf{not}\ L_n.$$

where the head of the rule $H(r) = \{L\}$ is either empty or consists of a single literal and the body

$$\mathsf{body}(r) = \{L_0, \ldots, L_m, \mathsf{not}\ L_{m+1}, \ldots, \mathsf{not}\ L_n\}$$

is a subset of $\mathfrak{L}$. The body consists of a set of literals

$$\mathsf{body}(r)^+ = \{L_0, \ldots, L_m\}$$

and a set of default negated literals denoted by

$$\mathsf{body}(r)^- = \{L_{m+1}, \ldots, L_n\}.$$

If $\mathsf{body}(r) = \emptyset$, r is a fact and if $H(r) = \emptyset$, r is a constraint. A program without default negation is a *strict program*. For a given program P we define the *strict sub-program* for it as

$$P^{\mathsf{strict}}(P) =_{def} \{r \in P \mid \mathsf{body}(r)^- = \emptyset\}.$$

The language of rules constructed over the set of atoms At is referred to as $\mathcal{L}^{\mathsf{asp}}_{At}$. A finite set of sentences from $\mathcal{L}^{\mathsf{asp}}_{At}$ is called an *extended logic program* $P \subseteq \mathcal{L}^{\mathsf{asp}}_{At}$.

Rules with variables are treated as rule schemata that represent the set of grounded rules resulting from grounding the variables with all possible combinations of constant symbols. Formally, given a program P the *Herbrand universe* U_P comprises all constants occurring in P. For a given set of literals Lit we denote the set of grounded literals by $U(\mathsf{Lit})$. The set of ground rules for the *rule schema* r is given by replacing all variables in r with all possible combinations of constants from U_P and denoted by $\mathsf{ground}(r)$. The grounded version of a program P is given by $\mathsf{grnd}(P) = \bigcup_{r \in P} \mathsf{ground}(r)$.

The semantics of extended logic programs is defined as follows. A state S is a set of literals that is *consistent*, i. e., it does not contain any complementary literals L and $\neg L$. A state S is a *model* of a program P if for all $r \in \mathsf{grnd}(P)$

$$\text{if } \mathsf{body}(r)^+ \subseteq S \text{ and } \mathsf{body}(r)^- \cap S = \emptyset \text{ then } H(r) \cap S \neq \emptyset.$$

That is, a state is a model of a program if for all grounded rules, if the positive body of the rule is contained in the state and none of the negative body literals is contained in it, then the head has a non-empty intersection with the state. The formulation of the last condition as the requirement that the set consisting of the head literal intersected with the state has to be non-empty is important for the semantics of constraints. A constraint has an empty head such that this condition is unsatisfiable. This implies that the body of a constraint cannot be satisfied by any model. The *reduct* P^S of a ground program P relative to a set S of literals is defined as

$$P^S = \{H(r) \leftarrow \mathcal{B}^+(r) \mid r \in P, \mathcal{B}^-(r) \cap S = \emptyset\}.$$

An answer set of a program P is a state S that is the minimal model of $\mathsf{grnd}(P)^S$. The set of all answer sets of P is denoted by $\mathit{AS}(P)$.

A program P is *consistent* if and only if $AS(P) \neq \emptyset$. Note, that we defined answer sets of a program based on a state, which does not contain complementary literals. There are other notions of answer sets in the literature according to which answer sets can contain complementary literals. In this case the answer set is usually defined to equal the set of all literals. It is be easy to adapt the notion of consistency to such definitions.

There are different notions of equivalence for logic programs [246]. Here we introduce the most important ones. For some fixed set of atoms At two programs $P \subseteq \mathcal{L}^{\mathsf{asp}}_{At}$ and $P' \subseteq \mathcal{L}^{\mathsf{asp}}_{At}$ are:

strongly equivalent, $P \equiv_{SE} P'$, iff for all programs F,

$$AS(P \cup F) = AS(P' \cup F),$$

uniformly equivalent, $P \equiv_{UE} P'$, iff for all sets of facts F,

$$AS(P \cup F) = AS(P' \cup F),$$

answer set equivalent, $P \equiv_{SE} P'$, iff $AS(P) = AS(P')$, i. e., $F = \emptyset$.

If two programs are strongly equivalent, then they are also uniformly and ordinarily equivalent.

Up to this point we introduced different knowledge representation formalisms, in the following we introduce the theory of change of beliefs represented by some knowledge representation formalism.

2.5 BELIEF REVISION THEORY

The research area of belief revision is concerned with the question of how to change currently held beliefs in the light of new information. Research on this topic is fundamentally influenced by the seminal paper of Alchourrón, Gärdenfors and Makinson [5]. They formalized rationality postulates for change operations, called the AGM postulates, and showed relations to constructions for change operations for propositional logic. In the time since then, the original postulates were discussed, modified, extended and adapted to various other settings, see [94] for an overview.

In the following we introduce the original formulation [5] of the basic set of postulates for the revision of a *belief set*, a deductively closed set of formulae, by a formula. Then we introduce an alternative formulation on the basis of *epistemic states*, which contain more information than *belief sets*. After that we describe postulates for *belief bases*, which are finite sets of propositional formulae. Belief bases contain more information than *belief sets* and can be seen as a special case of an *epistemic state*. Further, we introduce some common constructions of change operators, exemplarily for *belief bases*. Finally we give a brief overview of the work on the principle based consideration of change operations for logic programs.

2.5.1 *Belief Sets*

The AGM approach and many consequent work in the area studies the change of a *belief set*, a set of propositional formulae Φ that is deductively closed, i. e., $Cn^{prop}(\Phi) = \Phi$, by a propositional formula ϕ. Considering deductively closed belief sets allows to define rationality postulates and change operations on the *knowledge level.*

If new information is acquired it has to be incorporated into the current set of beliefs. Given a belief set Φ and the new information being represented as a sentence ϕ, the operation of adding ϕ to the current beliefs is called *expansion* and denoted by $\Phi + \phi$. An expansion can be performed without problems if the new information ϕ is consistent with Φ. The expansion is uniquely determined as $\Phi + \phi = Cn(\Phi \cup \phi)$ given the six AGM postulates for expansion [5]. In the case of an inconsistency of Φ and ϕ, conflicts arising from the addition of ϕ to the current set of beliefs have to be resolved, which amounts to a revision of the beliefs. This means that some of the current beliefs have to be given up in order to obtain a consistent belief set. The AGM model defines six basic postulates a revision operator $*$ should satisfy:

(κ*1) $\Phi * \phi$ is a belief set.

(κ*2) $\phi \in \Phi * \phi$.

(κ*3) $\Phi * \phi \subseteq \Phi + \phi$.

(κ*4) If $\neg\phi \notin \Phi * \phi$, then $\Phi + \phi \subseteq \Phi * \phi$.

(κ*5) $\Phi * \phi$ is inconsistent *iff* ϕ is inconsistent.

(κ*6) If $\phi \equiv^p \psi$, then $\Phi * \phi = \Phi * \psi$.

Postulate (K*1) ensures, that the result the revision of a belief set by a formula is a belief set. Postulate (K*2) is also called the *success postulate*. It demands, that the new information is prioritized over all current beliefs such that the new information is accepted in any case. Postulate (K*3) formalizes that the new information added by an expansion is an upper bound for the new information added by the revision operation. Postulate (K*4) states that if the new information is consistent with the belief set, then the expansion is a lower bound for the revision operation. Together Postulates (K*3) and (K*4) imply that in case of the new information being consistent with the current beliefs the revision should yield the same result as the expansion operation. The objective of a belief revision is to remove conflicts and to generate a consistent belief set. Postulate (K*5) demands consistency of the result, unless the new information is inconsistent in itself, in which case Postulate (K*2) does not allow for a consistent belief set. Postulate (K*6) formulates, that the result of a revision operation should be independent of the actual syntactic formulation of new information and should lead to the same result for semantically equivalent new information.

2.5.2 *Epistemic States*

A belief set represents the set of beliefs an agent is committed to at some point in time. *Belief sets* lack any information on the structure of the beliefs. A representation of this structure is necessary to perform change operations that follow a foundational approach that consider the dependencies of beliefs [106, 127, 65]. A general representation is based on *epistemic state* that represents all information that is necessary for reasoning and most importantly a representation of *conditional beliefs*, i. e., beliefs that the agent adopts, conditioned on other beliefs or evidence. An epistemic state is often represented by a total pre-order on the set of interpretations of the considered propositional language [65], or as an *ordinal conditional function* that assigns natural numbers to interpretations [220, 143, 144]. In other works epistemic states consist of sequences of propositional formulae [49], or can be composed of different elements [43]. A common view is that a belief set can be generated from an epistemic state $\mathcal{K}$ by means of a belief operator Bel, such that the belief set is given by $\mathsf{Bel}(\mathcal{K})$. In [65] the AGM postulates have been reformulated for epistemic states and and a change operator $\circ$ for epistemic states. There it is assumed that the belief set is represented as a single propositional formula. The basic postulates are as follows:

(R*1) $\mathsf{Bel}(\mathcal{K} \circ \mu) \vdash \mu$.

(R*2) If $\mathsf{Bel}(\mathcal{K}) \wedge \mu$ is satisfiable, then $\mathsf{Bel}(\mathcal{K} \circ \mu) \equiv^{p} \mathsf{Bel}(\mathcal{K}) \wedge \mu$.

(R*3) If μ is satisfiable, then $\mathsf{Bel}(\mathcal{K} \circ \mu)$ is satisfiable.

[(R*4′) If $\mathsf{Bel}(\mathcal{K}) \equiv^p \mathsf{Bel}(\mathcal{K}')$ and $\mu \equiv^p \mu'$, then

$$\mathsf{Bel}(\mathcal{K} \circ \mu) \equiv^p \mathsf{Bel}(\mathcal{K}' \circ \mu')]$$

(R*4) If $\mathcal{K} = \mathcal{K}'$ and $\mu \equiv^p \mu'$, then $\mathsf{Bel}(\mathcal{K} \circ \mu) \equiv^p \mathsf{Bel}(\mathcal{K}' \circ \mu')$

Postulate (R*1) corresponds to Postulate (K*2), Postulate (R*2) to Postulates (K*3) and (K*4), Postulate (R*3) to Postulate (K*5), and Postulate (R*4) and to Postulate (K*6). The Postulate (R*4) is a weakening of Postulate (R*4′) that only demands the revision results for equivalent inputs to be equivalent if the epistemic states are identical; instead of equivalent as demanded in the stricter formulation in Postulate (R*4′). This difference is characteristic for the use of epistemic states: The belief set alone does not contain sufficient information to determine the revision result, it depends on the exact epistemic state. The Postulates (R*1) to (R*4), without (R*4′), form the basis for the revision of epistemic states and are extended by additional postulates for iterated revision in [65].

The postulates for epistemic states are also defined on the belief set level, but it is taken into consideration that the belief set is generated by an epistemic state. In the next section we give an overview of the research on belief base revision.

2.5.3 *Belief Bases*

In another strain of research the change of belief bases has been considered. Belief bases are finite sets of formulas. They represent more information than belief sets since they allow to differentiate between explicit and inferred beliefs [127]. That is, they can be seen as a special form of epistemic states. Postulates for belief bases have been formulated on the level of the belief base, and not on the belief sets that are generated by belief bases. Belief base change theory has been mostly developed by Sven Ove Hansson [122, 123, 124, 127], and others, e. g., [183, 206, 119]. In the following we summarize the main postulates and ideas for the construction of belief base change operators on the basis of [127].

A base revision operator $*$ changes a belief base $\mathcal{B} \subseteq \mathcal{L}_{\mathrm{prop}}$ by a sentence $\phi \in \mathcal{L}_{\mathrm{prop}}$ and returns the revised belief base $\mathcal{B} * \phi$. Moreover, an expansion operator $+$ is defined for belief bases as the *non-closing expansion operator* defined as $\mathcal{B} + \phi = \mathcal{B} \cup \{\phi\}$. The basic set of postulates demanded for a belief base revision operator is the following:

SUCCESS: $\phi \in \mathcal{B} * \phi$

INCLUSION: $\mathcal{B} * \phi \subseteq \mathcal{B} + \phi$

VACUITY: If $\mathcal{B} \cup \{\phi\}$ is consistent, then $\mathcal{B} + \phi \subseteq \mathcal{B} * \phi$

CONSISTENCY: If ϕ is consistent, then $\mathcal{B} * \phi$ is consistent

RELEVANCE: If $\phi' \in (\mathcal{B} \cup \phi) \setminus (\mathcal{B} * \phi)$, then there is a set H such that $\mathcal{B} * \phi \subseteq H \subseteq \mathcal{B} \cup \phi$ and H is consistent but $H \cup \{\phi'\}$ is inconsistent

UNIFORMITY: If for all $\mathcal{B}' \subseteq \mathcal{B}, \mathcal{B}' \cup \{\phi\}$ is inconsistent if and only if $\mathcal{B}' \cup \{\phi'\}$ is inconsistent, then $\mathcal{B} \cap (\mathcal{B} * \phi) = \mathcal{B} \cap (\mathcal{B} * \phi')$

The *Success* postulate states that the new information should be part of the revision result, i. e., it should be prioritized over the information in the belief base. *Inclusion* demands that revision by some information should not introduce more information than expansion. *Vacuity* demands that if the new information is consistent with the belief base then no information should be discarded. *Consistency* postulates that if the new information is consistent in itself then the result of the revision is consistent as well. *Relevance* states that any discarded piece of information would have lead to inconsistency in a super set of the revision result. Finally, *Uniformity* says that the effect of revision on $\mathcal{B}$ is basically determined by subsets which are inconsistent with the new information.

The construction of belief base revision operators that satisfy the postulates specified above can be realized in the same way as belief set revision operators, with only slight differences as we describe in the following. Besides *expansion* and *revision*, *contraction*, denoted by $-$, which removes information from a belief base $\mathcal{B}$, is the third change operation considered in classical scenarios. It is linked to believe base revision via the *Levi-Identity* by use of a *contraction operator* $-$ and an *expansion operator* $+$:

$$\mathcal{B} * \phi = (\mathcal{B} - \neg\phi) + \phi.$$

The Levi-Identity helps establishing connections between rationality properties of the contraction and the revision operator, respectively. A standard construction is to define a contraction $\mathcal{B} - \phi$ as the intersection of a selection of maximal sets not containing ϕ. Formally, for a given belief base $\mathcal{B}$ and a sentence ϕ the set of *remainder sets* $\mathcal{B} \perp \phi$ is such that $X \in \mathcal{B} \perp \phi$ if and only if:

1. $X \subseteq \mathcal{B}$,
2. $\phi \notin Cn^{\mathsf{prop}}(X)$ and
3. there is no X' such that $X \subset X' \subseteq \mathcal{B}$ and $\phi \notin Cn^{\mathsf{prop}}(X')$.

Let $\mathcal{B}$ be a set of sentences. A function γ is a *selection function* for $\mathcal{B}$ if and only if for all sentences ϕ

1. if $\mathcal{B} \perp \phi \neq \emptyset$ then $\emptyset \neq \gamma(\mathcal{B} \perp \phi) \subseteq \mathcal{B} \perp \phi$, and
2. if $\mathcal{B} \perp \phi = \emptyset$ then $\gamma(\mathcal{B} \perp \phi) = \{\mathcal{B}\}$.

The *partial meet contraction* operator is then defined as

$$\mathcal{B} -_{\gamma} \phi = \bigcap \gamma(\mathcal{B} \perp \phi).$$

It has been shown that a revision operator satisfies all of the above postulates if and only if it is constructible via the Levi-identity and a partial meet contraction operator.

Partial meet contraction can be applied to believe sets in the same way as for belief bases. The difference between the revision of belief sets and belief bases results from the difference of the expansion operator. For belief bases the *non-closing expansion operator* defined as $\mathcal{B} + \phi = \mathcal{B} \cup \{\phi\}$ is used and for belief sets the *closing expansion operator* $\mathcal{B} \dot{+} \phi = Cn^{\mathsf{prop}}(\mathcal{B} \cup \{\phi\})$. This slight difference has implications for the resulting revision operator. These implications are reflected by the corresponding postulates for the revision of belief sets [5]. The postulates of Success and Consistency are identical and the postulates Inclusion and Vacuity only differ in the expansion operator. In addition to these four postulates the basic six AGM postulates contain the following two postulates:

(K*1) $\Phi * \phi$ is a belief set.

(K*6) If $\phi \equiv^{p} \psi$, then $\Phi * \phi = \Phi * \psi$.

Postulate (K*1) states that the revision result shall be a belief set, i. e., it shall be deductively closed. Postulate (K*6) demands for irrelevance of syntax, that is, all logically equivalent formulae shall lead to the same change. The (K*1) postulate is an inherent feature of the knowledge level approach and is definitely not appropriate in a base change framework. The idea of (K*6) is that the result of a revision operation should only depend on the semantic content of the information and not its syntactical representation. This is also desirable for belief base operations in general as well, but is not satisfied by these.

This far we described different approaches of change operations primarily developed for propositional logic. In the next section we give an overview of the works that considered the comparison with or the application of these approaches to answer set programs.

2.5.4 *Answer set programming and belief change*

Belief dynamics within classical logics have been studied for over 25 years now [94]. Well defined sets of rationality postulates for different change operations have been defined. For logic programs, however, most existing approaches to dynamics are very pragmatic and lack a formal representation of requirements of the change process. Few works examined the formal properties of approaches to change logic programs. Most of these use the classic AGM postulates for the revision of belief sets for logic programs. The adequate definition of a belief set, a consequence relation and a notion of equality is crucial but not at all trivial for the applicability of the classic postulates.

The most thorough account to study change operations to extended logic programs in the light of established postulates from belief revision theory can be found in [87]. In this work an epistemic state is represented by a sequence of logic programs $\mathbf{P} = (P_1, \ldots, P_n)$. The revision of a sequence $\mathbf{P} = (P_1, \ldots, P_n)$ by a program P results in the sequence $(\mathbf{P}, P) = (P_1, \ldots, P_n, P)$. The belief set is determined by constructing a single program $P_\triangleleft$ over an extended alphabet from the sequence. This program is called update program and resolves possible inconsistencies between the single programs giving priority to programs with higher indices. It generalizes several approaches from the literature that are based on the causal rejection principle [87].

The answer sets of this program, $AS(\mathbf{P}_\triangleleft)$ are then projected to the alphabet of the programs At. For the projected answer sets the set of rules satisfied by all answer sets is determined and resembles the belief set for the epistemic state, i. e.:

$$\mathsf{Bel}(\mathbf{P}) = \bigcap_{S \in AS(\mathbf{P}_\triangleleft)} \{r \in \mathcal{L}^{\mathsf{asp}}_{At} \mid (S \cap At) \text{ is a model of } \{r\}\}$$

Based on this definition of an epistemic state and belief operator the authors considered adopted versions of the AGM postulates, postulates for belief update [141] and for iterated change [65]. The majority of all important postulates are violated, many also for the case in which the sequence consists of only a single program. Due to this, ASP specific postulates for change operators have been proposed in [87] and adopted by several authors for the evaluation of their approaches afterwards [75, 68, 74, 188].

A more successful approach of a relation to the AGM theory was achieved by the use of monotonic SE-Models first presented in [68]; and later on extended to merge operations [71] and to update operations in the style of Katsuno and Mendelson by Slota and Leite [216, 217]. Said work makes use of the semantic characterization of programs via SE-models and applies an adapted version of distance based revision operators from classic belief revision. This approach was shown to satisfy the majority of the adopted AGM postulates. It is, however, still an open question if the semantic approach based on SE-models is adequate for ASP. In [216] it has been pointed out that the SE-models for the programs $P_1 = \{p., \ q.\}$ and $P_2 = \{p \leftarrow q., \ q.\}$ are the same and therefore also the results of the revision by the program $Q = \{\neg q.\}$. It holds that the sets of answer sets for both revision results are the same and have $\{p\}$ as the only answer set, i. e., $AS(P_1 *^{SE} Q) = AS(P_2 *^{SE} Q) = \{\{p\}\}$. While for P_1 this is a desired result, for P_2 it is not since p is not justified if q is not in the answer set.

2.5.5 *Summary*

In this chapter we introduced concepts and formalisms that are important for our considerations in the following chapters. We introduced the theory of agents, an abstract agent model and the BDI agent architecture in Section 2.1. Then, we introduced three different knowledge representation formalisms that are used in this thesis: propositional logic in Section 2.2, deductive argumentation in Section 2.3, and answer set programming in Section 2.4. Afterwards, in Section 2.5, we summarized important concepts of belief change theory for belief sets, epistemic states and belief bases.

3 EPISTEMIC AGENT MODEL

In this chapter we develop our epistemic agent model. We argued in Section 1.2.1 that there is a need for an agent model that embraces abstract notions of agents as well as concrete ones such as the BDI architecture on the one hand; and one that integrates approaches to belief change and non-monotonic reasoning on the other hand.

We presented the typical notion of abstract agents in Section 2.1.1. It considers agents as functions that map an environment state to an action. A state transformer function produces the set of possible successor environment states. That is, these abstract agents do not possess an explicit internal state and the environment is assumed to be completely modeled and available to the agent. In contrast, the BDI model, as introduced in Section 2.1.2, defines an agent to have a mental state that contains representations of the agent's current beliefs, desires, and intentions. The agent's behavior is implemented by an agent cycle comprising four functions. These resort to the agent's current percept and the components of the mental state, modify these, and determine the next action. The environment is not explicitly modeled and only accessible to the agent by means of percepts and actions.

The goal in this chapter is to define a general model of agents that can be used for the abstract analysis of agents, and that captures composed agent architectures such as the BDI model. Moreover, these modeled agents shall be based on logical knowledge representation, (non-monotonic) logical inference and belief change theory. We call them *epistemic agents* and their internal state an *epistemic state*. An epistemic state can consist of one or several epistemic components that are each based on a logical formalism and have a belief operator that determines a belief set for a given epistemic component. An epistemic component might be instantiated, for instance, by an extended logic program or a set of programs, both potentially in combination with a preference relation and a belief operator based on answer set semantics [112, 59, 87]. It might also be instantiated by an ordinal conditional function [220, 143], default logic [60], an argumentation formalism [24], a propositional set of sentences or a stratified propositional epistemic component [92], to name just a few.

Epistemic agents have an incomplete and uncertain view on the world and other agents. In particular, the state of the environment is not accessible to the agents, i. e., they have to rely on their subjective view on it. For each agent there is a set of *percepts* Per that it can process and a set of *actions* Act that it can perform. For our considerations we focus our model on communication, realized by the interchange of speech acts. We follow a top-down approach to develop the structure and properties of an epistemic agent. We start with an abstract model of an agent with a state and then gradually concretize the components of an agent. These are then studied in more detail by defining sub-components, and their dependencies and interactions. This way we define an extended BDI agent model, which we call the BDI$^+$ model. The BDI$^+$ model extends the BDI model by a motivation component [159] and a know-how component [229].

We define basic instantiations of the functional component and some of the components of the epistemic state of this model. The resulting agent model leaves only an evaluation function for actions and the actual knowledge representation to be instantiated. It is thus a good basis to easily specify concrete agents on the basis of different knowledge representation formalisms. This includes different belief operators and change operators, or different evaluation functions. We can model for instance ASP agents or agents that evaluate actions with respect to the preservation of secrets. Along with the development of the formal model, we continuously formalize an instance of our model that is based on answer set programming.

Furthermore, we formalize families of belief operators that can be ordered, for instance, by their credulity. We formalize properties of these and consider operators for extension based formalisms. We also consider instantiations of belief operator families for propositional logic and answer set programming, of which we make use later on in this thesis. The remainder of this chapter is structured as follows. In Section 3.1 we start defining the basic notions of the agents and multiagent systems we consider. In particular we introduce communication as a special form of percept and action. Then we define the fundamentals of epistemic components and belief operators for our epistemic agents in Section 3.2. In Section 3.3 we define our general epistemic agent model. We start defining our ASP instance, which also serves as a running example. Based on this abstract model, in Section 3.5, we define the structure of the epistemic state and the basic functional component for a BDI$^+$-based model. We also define a more concrete basic BDI$^+$ agents in the same section. In Section 3.6 we elaborate on the concept of belief operators, which we introduced in Section 3.2. We develop general families of operators for forming beliefs in the light of incomplete information. Further, we develop general desirable properties of such operators and families thereof. We use the general perspective of extension-based, non-monotonic formalisms to formulate more concrete properties. A variety of approaches to non-monotonic reasoning are directly or indirectly based on a notion of extensions [173]. We give two concrete instantiations of belief operator families, a propositional one, and an ASP instance as a representative of an extension-based instance. In Section 3.7 we discuss related work to the approaches presented in this chapter and in Section 3.8 we summarize and conclude.

3.1 COMMUNICATING AGENTS

In general actions of an agent in an environment have direct and indirect effects on the state of the environment [243]. The altered state of the environment is then, partially, perceived by the agent. Hereby an agent can perceive the effects of its own actions and evaluate if they were successful or if they failed. Actions that change the environment are called *ontic actions*. In contrast, *epistemic actions* only have an effect on the epistemic states of agents. For a more detailed discussion of the differences between ontic and epistemic actions see, e. g., [233]. In this thesis, we focus on *communicating agents* that only perform particular *epistemic actions*, which are called *speech acts* [212].

We abstract from the physical mediation of speech acts and assume that communication is direct and that sender, receiver, type and content are explicit. This also means, that we assume that every speech act is always received by the intended receivers. While communication is simple from a technical point of view, from the epistemic point of view it can be highly complex. Agents reveal information of different levels by transmitting a speech act. Besides the explicit informational content of a speech act, represented by a logical formula, they reveal meta-information such as the type of speech act, the sender, and receivers. Moreover, the context of a speech act is important for its interpretation. All of these aspects are relevant for the adequate processing of the speech act. A speech act of an agent generally has the effect of a change of the epistemic state of the receivers of the speech act. These changes are, in contrast to changes induced by ontic actions in a simulated or physical environment, not directly perceptible by agents through observation of changes in the environment. In particular, the agent performing a speech act cannot verify the success of its action directly, by observation of the anticipated changes in the physical environment. The agents have to be capable to reflect the anticipated changes implied by their actions in their own epistemic state.

Example 3.1.1. If *Beatriz* explains a complex and important task to *Emma*, she cannot see directly if *Emma* really understood what she has to do, and that she now intends to do it. Thus she cannot immediately verify if her speech act was successful. ♢

We consider purely communicating agents such that each agent's possible actions are given by a set of speech acts. The possible percepts of each agent result from the union of all possible actions of other agents that are directed to the former agent.

We do not require that all agents use the same knowledge representation formalism. For the communication between agents, which potentially use different knowledge representation formalisms, we presuppose a *common logical language* $\mathcal{L}_{\mathsf{Base}}$, that all agents can process. In this thesis we consider this language to consist of the set of propositional literals Lit.

We consider *multiagent systems* that contain a set of interacting agents that are identified by an *agent identifier*. The set of agent identifiers of a system is denoted by Ag. We use a notion of speech acts that is a simplification of the FIPA specification [100]. The general form of the speech acts we consider is

$$\langle \mathcal{X}_s, \mathcal{Y}_r, \mathsf{type}, \phi \rangle,$$

specifying the source $\mathcal{X}_s \in \mathsf{Ag}$, the receivers $\{\mathcal{Y}_{r_1}, \dots, \mathcal{Y}_{r_n}\} \subseteq \mathsf{Ag}$, the type of speech act $\mathsf{type} \in \mathsf{types}$ and the informational content $\phi \in \mathcal{L}_{\mathsf{Base}}$. We distinguish between requesting and informative types of speech acts. We define the type of a speech act $\mathsf{type} \in \mathsf{types}_I \cup \mathsf{types}_R$ whereby $\mathsf{types}_I = \{\mathsf{inform}, \mathsf{answer}\}$ is the set of informative speech acts and $\mathsf{types}_R = \{\mathsf{query}\}$ the set of requesting speech acts. The possible percepts and actions of communicating agents are given by sets of speech acts.

Definition 3.1.2 (Perceptions and Actions for Communication). The set of possible actions of an agent $\mathcal{X} \in \mathsf{Ag}$ is given by the set of speech acts

$$\mathsf{Act}_{\mathcal{X}} = \{\langle \mathcal{X}, \mathcal{Y}_r, \mathsf{type}, \phi \rangle \mid \mathcal{Y}_r \in \mathsf{Ag} \setminus \{\mathcal{X}\}, \mathsf{type} \in \mathsf{types}, \phi \in \mathcal{L}_{\mathsf{Base}}\} \cup \epsilon.$$

That is, the possible actions of an agent are given by all speech acts in which the respective agent is the sender. The symbol ϵ represents the empty action, it formally gives the agent the possibility to do nothing. The set of possible percepts of agent $\mathcal{X}$ is given by the speech acts it might perceive from other agents

$$\mathsf{Per}_{\mathcal{X}} = \{\langle \mathcal{X}_s, \mathcal{X}, \mathsf{type}, \phi \rangle \mid \mathcal{X}_s \in \mathsf{Ag}, \mathsf{type} \in \mathsf{types}, \phi \in \mathcal{L}_{\mathsf{Base}}\} \cup p_\epsilon.$$

This is the set of speech acts in which the respective agent is one of the receivers. The empty percept p_ϵ formally gives the agent the possibility to not receive any speech act.

Example 3.1.3. We consider our strike committee meeting example and the set of literals as the base language, i. e., $\mathcal{L}_{\mathsf{Base}} = \mathsf{Lit}$. The agent *Beatriz* sends a speech act

$$\langle \mathit{Beatriz}, \mathit{Emma}, \mathsf{query}, \mathsf{attend\_scm} \rangle$$

to agent *Emma*. The answer *Emma* should send to preserve her secret, that she intends to attend the *strike committee meeting*, is

$$\langle \mathit{Emma}, \mathit{Beatriz}, \mathsf{answer}, \neg\mathsf{attend\_scm} \rangle. \qquad \diamondsuit$$

Let Ag be a set of agents and Lit a set of literals. The *set of all speech acts* Σ is defined as

$$\begin{aligned}\Sigma = \{\langle \mathcal{X}_s, \mathcal{X}_r, \mathsf{type}, \mathsf{L}\rangle \mid \\ \mathcal{X}_s, \mathcal{X}_r \in \mathsf{Ag}, \mathsf{type} \in \mathsf{types}, \mathsf{L} \in \mathsf{Lit}\} \cup \{\epsilon, p_\epsilon\}.\end{aligned} \tag{3.1.1}$$

The symbols for the empty action ϵ and for the empty percept p_ϵ are added for the pure technical reason to formalize not-acting and not-perceiving.

The sets of speech acts we consider here can be easily extended by other types of speech acts such as, e. g., *perform, justify, propose, accept, agree*. For our means this formalization is completely sufficient. In fact, almost all types of communication can be reduced to the basic set we consider here [248].

The logical content of a speech act has a relation to an epistemic component and the resulting belief set of an agent. A sending agent decides on the speech act it sends based on its relation to its beliefs, and a receiving agent incorporates the information from the received speech act into its epistemic component. In the next section we formalize epistemic components and belief operators.

3.2 epistemic components

We use *epistemic componentsepistemic component*, i. e., finite sets of sentences of a logical language to formalize the knowledge representation and reasoning of an agent. An epistemic component can be seen as an epistemic state or a belief base as we considered for belief change in Section 2.5. An epistemic component forms the basis for logical inferences of the agent. The set of inferences from an epistemic component results in the agent's *belief set*. We aim at facilitating the use of formalisms for non-monotonic reasoning in agent systems. Examples we have in mind are default logics [202], argumentation systems [24], various forms of logic programs [47], or ordinal conditional functions [220]. To allow for such a variety of formalisms for knowledge representation, we keep our concepts and notions independent of a particular formalism and make only some general assumptions. We fix a syntactic structure that is general enough to support a wide variety of such formalisms. All formalisms are based on an underlying *base language* $\mathcal{L}_B$ representing the basic vocabulary. This is typically a fragment of propositional logic. This base language is then used to define the *epistemic component language* $\mathcal{L}_E$. These two languages might be equal or the latter is an extension of the former, that is that the sentences of the former are, potentially, connected by additional constructors to form the latter language.

Example 3.2.1. For the ASP case, cf. Section 2.4, the base language are atoms and the epistemic component language are rules, which

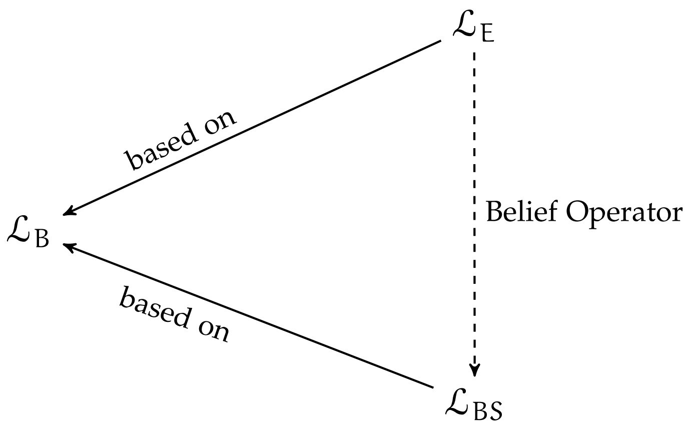


Figure 3.2.1: Relations of languages for Belief Bases

are formed by the rule constructor $\leftarrow$, the strict negation $\neg$ and the default negation not. ◇

An epistemic component is a set of sentences of the *epistemic component language* $\mathcal{L}_E$. On the epistemic component a *belief operator* is applied to produce the set of currently held beliefs, the *belief set* BS, which is a set of sentences of the *belief set language* $\mathcal{L}_{BS}$. The belief set language is also based on the base language and potentially built by use of some extra constructors, but usually not as rich as the epistemic component language. We consider agents that have incomplete information. Thus, given an epistemic component E the agent has to decide which sentences it believes to be true. The set of such sentences forms its current belief set BS. To formalize this process we introduce belief operators.

Definition 3.2.2 (Belief Operator). A *belief operator* is a function

$$\mathsf{Bel} : \mathcal{P}(\mathcal{L}_E) \to \mathcal{P}(\mathcal{L}_{BS})$$

that maps an epistemic component $E \subseteq \mathcal{L}_E$ to a belief set $BS \subseteq \mathcal{L}_{BS}$.

The general relations among the languages we introduced are illustrated in Figure 3.2.1. We assume that an agent $\mathcal{X}$ has for each epistemic component E a belief operator $\mathsf{Bel}^E_{\mathcal{X}}$ assigned to it. If an agent has just one epistemic component or if all epistemic components of an agent are of the same type and use the same operator we write $\mathsf{Bel}_{\mathcal{X}}$ to denote the *agent's belief operator*.

We illustrate this general concept by use of a basic propositional instance and an ASP instance.

Example 3.2.3. For classical propositional logic the base language are the propositional atoms, i.e., $\mathcal{L}_B = At$. The epistemic component, and belief set language are the language of propositional formulae,

i. e., $\mathcal{L}_E = \mathcal{L}_{BS} = \mathcal{L}_{At}^{\mathsf{prop}}$. The belief operator is the propositional consequence operator: $\mathsf{Bel}^{\mathsf{prop}} =_{def} \mathsf{Cn}^{\mathsf{prop}}$. $\diamondsuit$

ASP Instance 1. The languages we just introduced are instantiated as follows for our running ASP instance. The base language of answer set programming is a set of propositional atoms $\mathcal{L}_B^{\mathsf{asp}} = At$. The epistemic component language is the language of extended logic programs $\mathcal{L}_{EC}^{\mathsf{asp}} = \mathcal{L}_{At}^{\mathsf{asp}}$. It is constructed on the basis of $\mathcal{L}_B^{\mathsf{asp}}$ by means of the rule constructor $\leftarrow$ and the default negation not constructors resulting in rules of the form

$$L_0 \leftarrow L_1, \ldots L_n, \mathsf{not}\ L_{n+1}, \ldots, \mathsf{not}\ L_m.$$

with $L_i \in \mathsf{Lit}$, $0 \leqslant i \leqslant m$ with $\mathsf{Lit} = \{\neg A \mid A \in \mathcal{L}_B^{\mathsf{asp}}\} \cup \mathcal{L}_B^{\mathsf{asp}}$, as defined in Section 2.4. Hence the belief set language is the set of literals $\mathcal{L}_{BS}^{\mathsf{asp}} = \mathsf{Lit}$, which is constructed from the set of atoms At by the strict negation constructor.

An *ASP belief operator* is of the form $\mathsf{Bel} : \mathcal{P}(\mathcal{L}_{At}^{\mathsf{asp}}) \to \mathsf{Lit}$, for instance defined as the skeptical operator $\mathsf{Bel}_{\mathsf{skep}}(P) =_{def} \cap AS(P)$, also called *cautious entailment*, e. g., in [112].

In this and the previous section we defined the fundamentals for the knowledge representation and interaction of our epistemic agents. In the next section we formalize our epistemic agent model.

3.3 epistemic agents

In the following we define our model of epistemic agents. Formally, we start with a definition of an *agent with state*, as sketched in [248]. We refine this abstract concept and consider composed states and functions to define our model of epistemic agents.

An agent is represented by its agent identifier. Let a multiagent system with a set of agent identifiers Ag be given. For each point in time the agent identifier $\mathcal{X} \in \mathsf{Ag}$ points to the respective current *agent state* of $\mathcal{X}$. An agent state comprises an epistemic state $\mathcal{K}$ and a functional component ξ. The epistemic state of an agent evolves during runtime of the agent. Its functional component does not change. The epistemic state is formulated in a language of epistemic states $\mathcal{L}_{ES}$. This language can, for instance, be the power set of an epistemic component language $\mathcal{L}_E$, e. g., the set of all extended logic programs for a given set of atoms, or the set of total preorders or ordinal conditional functions for a given propositional language. An agent's epistemic state contains the agent's *know-that*, describing the agent's beliefs about the world, as well as its *know-how*, representing information about how to achieve goals, cf. [215]. Further, it contains representations of the agent's goals.

Each agent starts with an *initial epistemic state* $\mathcal{K}^0$. In each agent cycle an agent receives a percept $p \in \mathsf{Per}$. Upon reception of the speech act it changes its epistemic state to incorporate the new information represented by this percept. As we argued in Section 3.1, a communicating agent has to be capable of changing its epistemic state upon the reception of an incoming percept, which changes its view on the system and leads to the agent's decision for the next action; and on the basis of the action it intends to execute, such that its view on the system reflects the anticipated changes of the action. We model this by use of two separate change operators. The modifications to the epistemic state upon reception of a p are realized by the change operator

$$\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$$

of the agent, i.e., the new state is given by $\mathcal{K}^0 \circ p$. This new state determines the next action of the agent. This action is produced from the epistemic state by the function

$$\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}.$$

The action $\mathsf{act}(\mathcal{K}^0 \circ p)$ is executed in the environment and used as input for the *change operator for actions*

$$\circ^a : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}.$$

This change operator modifies the epistemic state to reflect the anticipated changes of the execution of the action. Hence, an agent with current epistemic state $\mathcal{K}$ has, after a complete *agent cycle* with percept p , the new epistemic state

$$\mathcal{K} \circ p \circ^a \mathsf{act}(\mathcal{K} \circ p). \tag{3.3.1}$$

The action that is executed is $\mathsf{act}(\mathcal{K} \circ p)$. Hence, the functional component of an agent comprises the operators $\circ$, $\circ^a$ and act. This agent cycle is illustrated in Figure 3.3.1.

We assume a discrete representation of *time*, represented by natural numbers, as in other formalizations of multiagent system, e.g., in [89].[1] The agents are executed one after the other in an arbitrary but fixed order. With each execution of an agent cycle the time is increased by 1. At each point in time the state of an agent is given by an *agent!state* as defined next.

Definition 3.3.1 (Agent State). Let Per be a set of percepts, Act a set of actions and $\mathcal{L}_{ES}$ a language of epistemic states. An *agent state* is a tuple $(\mathcal{K}, \xi)$ comprising of an epistemic state $\mathcal{K} \in \mathcal{L}_{ES}$ and a functional component $\xi = (\circ, \circ^a, \mathsf{act})$ with:

$$\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}, \circ^a : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES} \text{ and } \mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}.$$

The functional component is part of the agent state even though it does not change over time. This way an agent is completely defined at each point in time by reference to its agent state.

1 For a discussion on the use of a discrete formalization of time see [89].

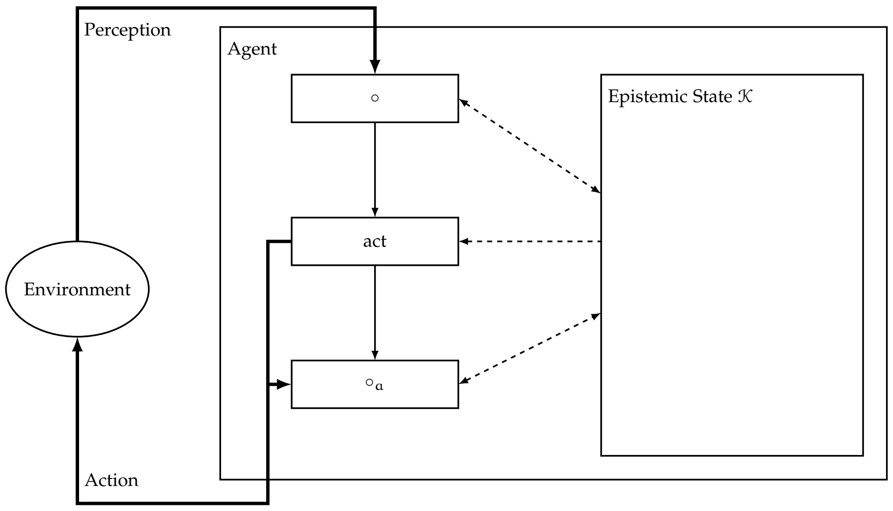


Figure 3.3.1: Epistemic Agent Cycle

At each point in time, only one agent state changes and only one action is performed. A *multiagent cycle* consist of the execution of all agents in some order. For example, in a multiagent system with 10 agents the time increases by 10 in each multiagent cycle. The agents are assumed to have access to a clock that always provides the current time.

We illustrate the just described concept of a multiagent system by use of our running example. Agents that interact with a physical environment get percepts from the environment, starting with the percept of the initial state of the world. In the communication setting we consider here the percepts of agents result from the speech acts from other agents. Since we defined an agent cycle to start with the reception of a percept also the very first agent to be executed has, for this technical reason, to receive a percept even though no other agent has yet performed a speech act. We model this by the *empty percept* p_ϵ, as introduced in Definition 3.1.2.

Example 3.3.2. We model the *strike committee meeting* example from Chapter 1 as a two agent system. The system contains the agent $\mathcal{E}$ for *Emma* and the agent $\mathcal{B}$ for *Beatriz* such that

$$\mathsf{Ag} = \{\mathcal{E}, \mathcal{B}\}.$$

The initial epistemic states are $\mathcal{K}^0_{\mathcal{E}}$ and $\mathcal{K}^0_{\mathcal{B}}$, the time is 0. The functional components are $\xi_{\mathcal{E}}$ and $\xi_{\mathcal{B}}$ such that the initial agent states are $(\mathcal{K}^0_{\mathcal{E}}, \xi_{\mathcal{E}})$ and $(\mathcal{K}^0_{\mathcal{B}}, \xi_{\mathcal{B}})$. Since the functional components are static, we identify the agent states by the respective epistemic states in the following. The complete scenario we describe here is illustrated in Figure 3.3.2.

We assume that the order of the execution of the agents is such that first $\mathcal{E}$ is executed and then $\mathcal{B}$. Hence, in the first multiagent cycle $\mathcal{E}$

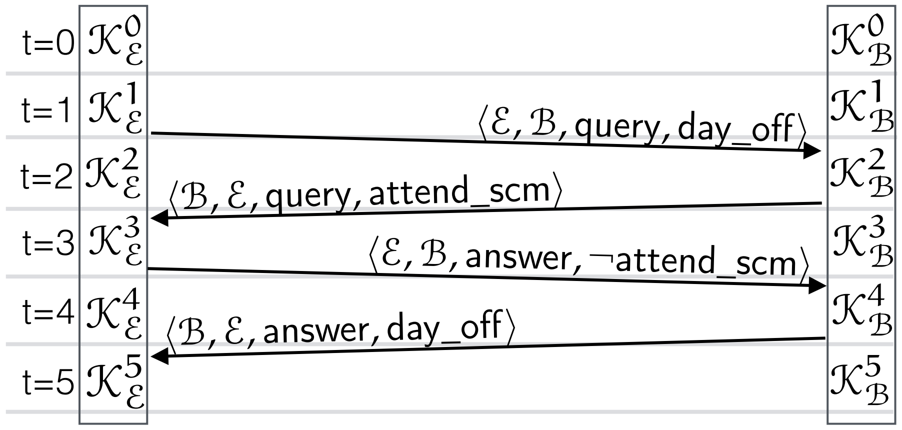

t=0 $\mathcal{K}^0_\mathcal{E}$ $\mathcal{K}^0_\mathcal{B}$
t=1 $\mathcal{K}^1_\mathcal{E}$ $\mathcal{K}^1_\mathcal{B}$
$\langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off} \rangle$
t=2 $\mathcal{K}^2_\mathcal{E}$ $\mathcal{K}^2_\mathcal{B}$
$\langle \mathcal{B}, \mathcal{E}, \mathsf{query}, \mathsf{attend\_scm} \rangle$
t=3 $\mathcal{K}^3_\mathcal{E}$ $\mathcal{K}^3_\mathcal{B}$
$\langle \mathcal{E}, \mathcal{B}, \mathsf{answer}, \neg\mathsf{attend\_scm} \rangle$
t=4 $\mathcal{K}^4_\mathcal{E}$ $\mathcal{K}^4_\mathcal{B}$
$\langle \mathcal{B}, \mathcal{E}, \mathsf{answer}, \mathsf{day\_off} \rangle$
t=5 $\mathcal{K}^5_\mathcal{E}$ $\mathcal{K}^5_\mathcal{B}$

Figure 3.3.2: Timeline for the scenario in Example 3.3.2.

is the first agent to be executed. For the technical reason explained above $\mathcal{E}$ receives the empty percept p^0_ϵ such that:

$$\mathcal{K}^1_\mathcal{E} = \mathcal{K}^0_\mathcal{E} \circ p^0_\epsilon \circ^a \mathsf{act}(\mathcal{K}^0_\mathcal{E} \circ p^0_\epsilon),$$

$$\mathcal{K}^1_\mathcal{B} = \mathcal{K}^0_\mathcal{B}.$$

Now agent $\mathcal{E}$ asks $\mathcal{B}$ for the day of the *strike committee meeting* off. Let this action be denoted by

$$a^1 = \mathsf{act}(\mathcal{K}^0_\mathcal{E} \circ p^0_\epsilon) = \langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off} \rangle.$$

We assume that this action results in the percept p^1 for $\mathcal{B}$ such that:

$$\mathcal{K}^2_\mathcal{B} = \mathcal{K}^1_\mathcal{B} \circ p^1 \circ^a \mathsf{act}(\mathcal{K}^1_\mathcal{B} \circ p^1),$$

$$\mathcal{K}^2_\mathcal{E} = \mathcal{K}^1_\mathcal{E}.$$

The action of agent $\mathcal{B}$ is then to ask $\mathcal{E}$ if she intends to attend the *strike committee meeting*. Let this action be denoted by

$$a^2 = \mathsf{act}(\mathcal{K}^1_\mathcal{B} \circ p^1) = \langle \mathcal{B}, \mathcal{E}, \mathsf{query}, \mathsf{attend\_scm} \rangle.$$

We assume that this action results in the percept p^2 for agent $\mathcal{E}$. The new epistemic states are given as:

$$\mathcal{K}^3_\mathcal{E} = \mathcal{K}^2_\mathcal{E} \circ p^2 \circ^a \mathsf{act}(\mathcal{K}^2_\mathcal{E} \circ p^2),$$

$$\mathcal{K}^3_\mathcal{B} = \mathcal{K}^2_\mathcal{B}.$$

This continues in the same way for the interaction illustrated in Figure 3.3.2. $\diamondsuit$

The agent's behavior is determined by its initial epistemic state in combination with its functional component. Hereby, the implementation of the behavior can be realized by the epistemic state and the reasoning based thereupon, or by the functional component. One extreme is a purely logic-based agent whose behavior is entirely defined by means of logical formulae represented in its epistemic state. The functional component of such an agent solely implements the interface to the environment and an inference operator for the logical formalism. Examples of languages for such agents are concurrent MetateM [97] and GOLOG [167]. As discussed in [248] such agents can also be built on the basis of various logical languages. The behavior of these agents entirely depends on the knowledge representation in the agent's epistemic state and a belief operator that performs the agent's reasoning, deliberation and means-ends reasoning. We do not consider purely logic-based agents in detail in this work. In the following example we sketch a purely ASP based agent to illustrate the idea of such an agent.

Example 3.3.3 (ASP Agent). We consider a purely logic based agent $\mathcal{X}$ based on ASP. Its epistemic state consists of a single epistemic component given by an extended logic program such that $\mathcal{L}_{ES} = \mathcal{P}(\mathcal{L}^{\mathsf{asp}}_{At})$. Its percepts and actions are represented as literals such that $\mathsf{Per} \subseteq \mathsf{Lit}$ and $\mathsf{Act} \subseteq \mathsf{Lit}$. The change operator is defined as $P \circ L = P \cup \{L.\}$. The agent's ASP belief operator $\mathsf{Bel}_{\mathcal{X}} : \mathcal{P}(\mathcal{L}^{\mathsf{asp}}_{At}) \to \mathcal{P}(\mathsf{Lit})$ is defined as $\mathsf{Bel}_{\mathcal{X}}(P) = \cap AS(P)$. The action operator is defined as $\mathsf{act}(P) = a$ for some uniquely determined $a \in \mathsf{Bel}_{\mathcal{X}}(P) \cap \mathsf{Act}$. The change operator for actions is defined to be the identity function $P \circ^{a} L = P$. ♢

The behavior of such an ASP agent is completely defined by an extended logic program and the belief operator. The change operator only adds percepts to the program, the action operator only transmits the determined action to the environment. The program has to handle conflicting information induced by the addition of the new percept. For more details on how this is done we refer to our work on this in [152] or the approaches of others, e. g., [15, 87, 7]. The act operator extracts the next action from the belief set of the agent, which is determined by the program and the application of the belief operator $\mathsf{Bel}_{\mathcal{X}}$. The change operator for actions is assumed to not change anything. For details on deliberation and planning with ASP we refer to our work on this in [157], or the work of others, e. g., [236, 111, 219].

The other extreme is an agent whose behavior is determined entirely by its functional component. In our model such an agent can be seen to only store the current percept in the epistemic state by means of the change operator and to map it to an action by the action operator. Hence the behavior of the agent is entirely determined by the action operator.

The intention of our agent model is to represent and facilitate the design of all agents in between the just described extremes.

3.4 ABSTRACT COMPOUND AGENT

This far we described an abstract agent model. As motivated in the introduction of this thesis and of this section we aim at refining this abstract model to capture agent models that are based on epistemic states with several components and a functional component with functions modifying these components, e. g., as in the BDI model.

If an epistemic state consists of a single component we call it *monolithic*, and *compound* otherwise. Example 3.3.3 is also an example of an agent with a monolithic epistemic state. Epistemic states generally have to represent different types of knowledge. As mentioned above an agent's epistemic state typically contains the agent's *know-that*, as well as its *know-how* and a representation of the agent's goals. BDI logics [200, 230] for instance use different modalities for beliefs, desires and intentions.

The most common approach for the design of agent models is to separate the different aspects to be represented in an epistemic state into different components. In the following we formalize a general concept of composed epistemic states and functional components. It generalizes the BDI model, extensions of it, and other compound models.

The BDI model, which we described in Section 2.1.2 (Page 19), is the most widely used agent model. It defines three separate components for the agent's beliefs, desires, and intentions.

Example 3.4.1. According to the BDI model epistemic states are of the form $\mathcal{K}_{\mathsf{BDI}} = \langle \mathfrak{B}, \mathfrak{D}, \mathfrak{I} \rangle$ with the agent's beliefs $\mathfrak{B} \in \mathcal{L}_{\mathfrak{B}}$, a set of desires $\mathfrak{D} \in \mathcal{L}_{\mathfrak{D}}$, a set of intentions $\mathfrak{I} \in \mathcal{L}_{\mathfrak{I}}$. The language of BDI epistemic states is then $\mathcal{L}_{\mathsf{BDI}} = \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{D}} \times \mathcal{L}_{\mathfrak{I}}$. ♢

In general a compound epistemic state is a component, which again can either be atomic or compound. An atomic component $\mathcal{C}$ is an element from the component's language $\mathcal{L}_{\mathcal{C}}$, e. g., the power set of an epistemic component language $\mathcal{P}(\mathcal{L}_E)$. A compound component is a tuple of components, $\mathcal{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_n \rangle$. The language of a compound component is a Cartesian product of languages: $\mathcal{L}_{\mathcal{C}} = \mathcal{L}_{\mathcal{C}_1} \times \cdots \times \mathcal{L}_{\mathcal{C}_n}$. In particular, each component can be represented by use of a different knowledge representation formalism.

Example 3.4.2. An example of a composed language is the language $\mathcal{L}_{\mathsf{BDI}} = \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{D}} \times \mathcal{L}_{\mathfrak{I}}$ of Example 3.4.1. These might, for example, be instantiated with the agent's belief being represented by logic programs, i. e., $\mathcal{L}_{\mathfrak{B}} = \mathcal{P}(\mathcal{L}^{\mathsf{asp}}_{\mathsf{At}})$, and its desires and intentions by literals, i. e., $\mathcal{L}_{\mathfrak{D}} \subseteq \mathcal{P}(\mathsf{Lit})$ and $\mathcal{L}_{\mathfrak{I}} \subseteq \mathcal{P}(\mathsf{Lit})$. ♢

The interaction of the components is realized by the functional component of the agent. In particular, for an epistemic state $\mathcal{K} \in \mathcal{L}_{ES}$

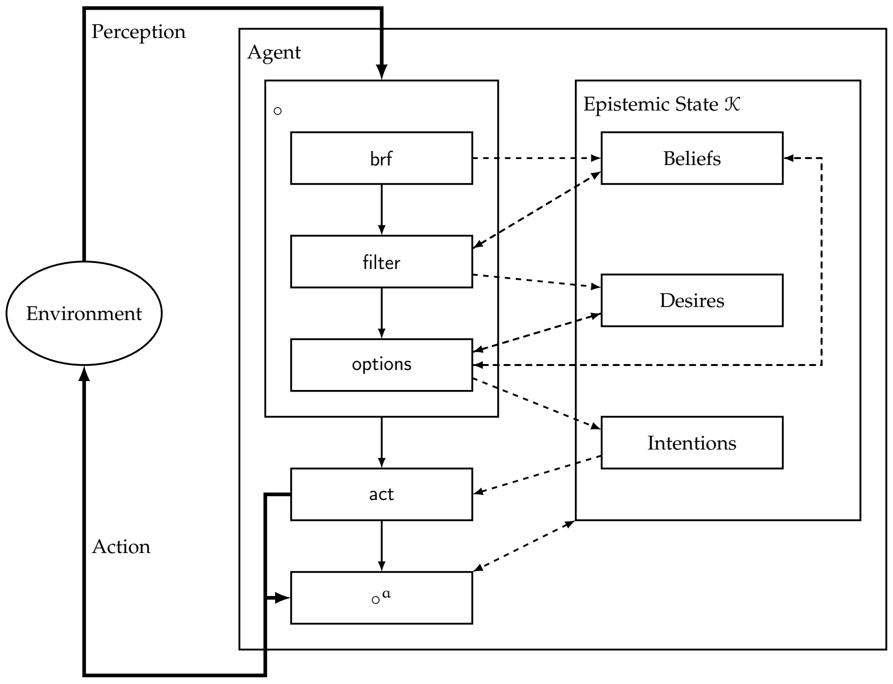


Figure 3.4.1: Simple BDI model as compound agent model

and a functional component $\xi = (\circ, \circ^a, \mathsf{act})$, the change operators $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$ and $\circ^a : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ are single functions or compositions of functions. The action function is assumed to be atomic. That is, we define the change operators to be realized by a sequence of sub-functions. Each sub-function of a change operator modifies a single component or a set of components of the epistemic state, the next function operates on the epistemic state that results from the modifications of the previous functions. This concept formalizes the idea of an agent cycle. The following example illustrates such a composed agent cycle for the BDI model.

Example 3.4.3. Figure 3.4.1 illustrates how the BDI model can be realized as a compound model. In the BDI model we introduced in Section 2.1.2 the functions are of the following type. The current beliefs of the agent are revised on the basis of the current percept $p \in \mathsf{Per}$ by the belief revision function

$$\mathsf{brf} : \mathcal{L}_{\mathfrak{B}} \times \mathsf{Per} \to \mathcal{L}_{\mathfrak{B}}.$$

Then, on the basis of the revised beliefs and the current set of intentions, the desires of the agent are determined by the function

$$\mathsf{options} : \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{I}} \to \mathcal{L}_{\mathfrak{D}}.$$

The agent's deliberation process, the process of choosing what to do, of a BDI agent is realized by the filter function

$$\mathsf{filter} : \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{D}} \times \mathcal{L}_{\mathfrak{I}} \to \mathcal{L}_{\mathfrak{I}}.$$

At last, the action function

$$\mathsf{act} : \mathcal{L}_{\mathfrak{I}} \to \mathsf{Act}$$

determines the agent's next action on the basis of a set of intentions. The change operator for actions $\circ^a$ is not needed in this model such that we set to the identity function id, i. e., $\mathcal{K} \circ^a a = \mathcal{K}$. ♢

As indicated in Figure 3.4.1 the functions brf, options and filter together realize the agent's change operator $\circ$. The change operator changes the agent's epistemic state upon the reception of a percept. The brf operator changes the beliefs of the agent given new information as its input. The functions options and filter do not directly use the incoming percept as input. The options function uses the beliefs that have already been changed by the percept as input to generate new desires. The filter function uses the changed beliefs and the changed desires as input to change the set of intentions accordingly. Hence, for the change of the epistemic state only the brf function processes the percept by adapting the beliefs component. The functions options and filter then realize the changes to the other components of the epistemic state on the basis of the already changed components.

In the following we formalize our general agent model with compound epistemic state and compound functional component. The general setup is illustrated in Figure 3.4.2. Formally, a compound functional component consists of a change operator that is a composition of operators, and an action function. The change operator is defined as

$$\circ =_{def} \circ_1 \cdot \ldots \cdot \circ_k. \qquad (3.4.1)$$

The function composition $\cdot$ is defined as usual but with reversed order, i. e., the functions are applied in the order of the sequence of concatenation, formally

$$f \cdot f'(x) = f'(f(x)). \qquad (3.4.2)$$

The composed change operator is of the type $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$. In order for this function composition to be well-defined we require of the domains and co-domains of the sub-operations $\circ_i : D_i \to coD_i, 1 \leqslant i \leqslant n'$ that:

1. $D_1 = \mathcal{L}_{ES} \times \mathsf{Per}$
2. $coD_n = \mathcal{L}_{ES}$
3. for $1 < i < n' - 1$: $coD_{i-1} = D_i$.

For the BDI model, as shown in Example 3.4.3, the domains and co-domains of the sub-functions of the BDI model only contain those components that are either needed as the input or that are modified

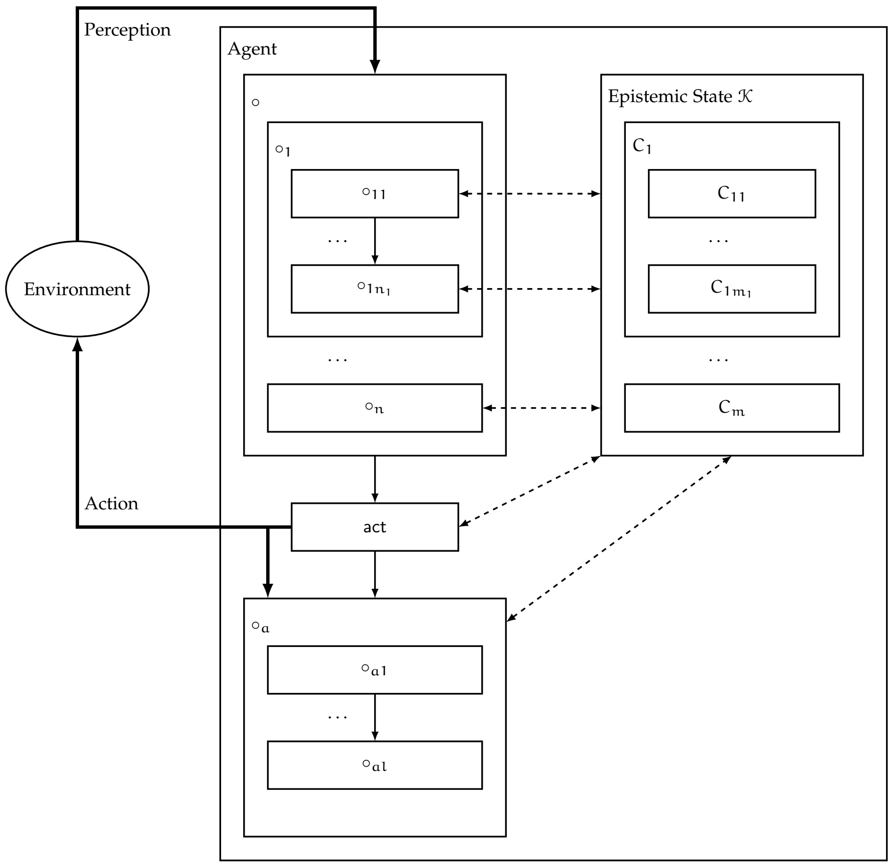


Figure 3.4.2: Compound agent model

by the function. This definition of the inputs and outputs of the functions is intuitive and clarifies the role of the functions. To be used as a sub-function in a composed functional component as described above, the domain and co-domain of the sub-functions have to be extended such that also the components of the epistemic state that are not modified are included in both. We denote the extended version of a function f by $f^\dagger$. These extensions are always possible since they only add components that are not modified to the domains and co-domains.

Example 3.4.4. The functions from Example 3.4.3 should be extended to

$$\mathsf{brf}^\dagger : \mathcal{L}_{\mathsf{BDI}} \times \mathsf{Per} \quad \to \mathcal{L}_{\mathsf{BDI}} \times \mathsf{Per},$$

$$\mathsf{options}^\dagger : \mathcal{L}_{\mathsf{BDI}} \times \mathsf{Per} \to \mathcal{L}_{\mathsf{BDI}} \times \mathsf{Per},$$

$$\mathsf{filter}^\dagger : \mathcal{L}_{\mathsf{BDI}} \times \mathsf{Per} \quad \to \mathcal{L}_{\mathsf{BDI}}.$$

With these extensions the functions satisfy the necessary conditions specified above for the function composition such that the operator $\circ_{\mathsf{BDI}} =_{def} \mathsf{brf}^\dagger \cdot \mathsf{options}^\dagger \cdot \mathsf{filter}^\dagger$ is well-defined. ♢

3.5 BDI$^+$ AGENTS

In this section we define compound agents with a composition in the style of the BDI model, which we presented in Section 2.1.2 and in examples in Section 3.3. Here, we extend this BDI model. Basically, we add two additional components to the epistemic state. One of these is a set of *motives* as we defined in [159]. Motives take the role of describing an agent's personality by causing the generation of desires. The other additional component is a *know-how base* on the basis of our work in [229, 157]. It realizes a plan library as at least assumed implicitly in most BDI architectures as in the original proposal [57] and as made explicit by most instantiations of the BDI model, e. g., [117, 118, 51]. However, in contrast to other approaches our approach follows the ideas of Singh in [215] to enable the agent to reason about its planning capabilities in the same way as it can reason about any other of its beliefs by extending a BDI-based agent architecture to allow the representation of procedural beliefs explicitly as part of the agent's logical beliefs, as we elaborated in [229, 157].

We call our BDI model with motivation and know-how a BDI$^+$ agent model. After these conceptual extensions we instantiate our model with basic realizations of all functions, which can easily be extended by use of more complex operators. The purpose of this basic instantiation is to have a concrete and versatile instantiation of the BDI$^+$ agent model that can be used to develop and analyze specialized agents on this basis, e. g., secrecy preserving agents or agents that use different forms of knowledge representation. Herein, the components and their interaction are instantiated such that only the actual knowledge representation is left to be instantiated.

The remainder of this section is structured as follows. In the next sections we introduce the concepts of motives (Section 3.5.1) and know-how (Section 3.5.2) for our BDI$^+$ agent model. Following on this we define the BDI$^+$ model in Section 3.5.3 and concretize it through our basic instantiations in Section 3.5.4.

3.5.1 *Motives*

Formal BDI models assume the desires of an agent to be given, as agents are typically situated in some constrained environment with limited capabilities and tasks. For a cleaner robot the two desires "*I want to clean the room*" and "*I want to have a high battery level*" might be completely sufficient to represent its setting. Generally, however, assuming desires to be given inhibits real autonomous behavior. Bringing agents into more complex environments demands for mechanisms that allow an agent to set its desires by itself. Looking at how humans handle their desires, motivation theory [179] explains how a being's personality leads to the desires it wishes to satisfy. A *motive* such as *"benevolence"* or *"greed"* is a basic characteristic of an agent's personality leads to the creation (or abandonment) of some desire. We illustrate this intuition with a simple example.

Example 3.5.1. Given an agent competes with other agents for food, the motive *"greed"* would generate the desire of acquiring as much food as possible, while the motive *"benevolence"* would generate the desire of acquiring just as much food as really needed and to help that other agents acquire as much food as they need. Both motives might very well be present in the agent's personality but with differing strengths. Moreover, these strength can change over time as the situation of the agent changes. $\diamondsuit$

In [159] we developed a formal account for the incorporation of motivation into the BDI approach. Instead of assuming desires to be given, we assume that an agent has some set of basic motives that drives its behavior. Each motive of the agent is equipped with some weight and each motive is coupled with a set of desires that can be positively or negatively influenced by the motive. We give a formal account on the aggregation of the weights of the motives and these couplings in order to determine the desires the agent is motivated to follow. Furthermore, using the notion of *reliability* [157] we investigate how beliefs about actions (*know-how*) and beliefs about the world (*know-that*) might influence the deliberation based on motivation. More precisely, the motivation to follow some desire is strongly influenced by the current uncertainty of the situation of the agent. This uncertainty is reflected by the agent's confidence in its ability to achieve its goals, as represented by its know-how and its view on the world. The more uncertain the situation, the more focused is the agent on satisfying its basic needs. We developed an approach to realize this shift of focus in [159]. Here, we use *belief-desire couplings* as a basic form of a motivational structure that we introduced in [159]. We also call belief-desire couplings motives in the following. By use of this basic version we define a clear, yet powerful, framework that can easily be extended to incorporate the full fledged approach.

A set of *belief-desire couplings* is denoted by $\mathfrak{M}_b$ and used to generate new desires. Formally, $\mathfrak{M}_b$ is a set of *belief-desire couplings* (ϕ, Φ, μ) that comprise a *desire* $D = (\phi, \mu) \in \mathcal{L}_{\mathfrak{D}}$ and a set of logical formulae $\Phi \subseteq \mathcal{L}_{BS}$. A desire is a tuple with a logical formula $\phi \in \mathcal{L}_{BS}$ and a *motivation value* $\mu \in [0, 1]$. We denote the power set of all possible belief-desire couplings by $\mathcal{L}_{\mathfrak{M},b}$. The semantics of a belief-desire coupling is that the desire (ϕ, μ) is only created if all formulae in Φ can be verified by the beliefs of the agent.

Example 3.5.2. The employee *Emma* might have the desire to take the day of a strike committee meeting off when she is informed that there is a strike committee meeting. This can be formalized by the belief-desire coupling

$$(\mathsf{day\_off}, \{\mathsf{scm},\ \mathsf{attend\_work}\}, 0.8). \qquad \diamondsuit$$

The other extension of the BDI model we introduce is *know-how*. We introduce it in the next section, before we define the complete BDI agents with motivation and know-how.

3.5.2 *Know-How*

An agent has to determine how it can satisfy its goals. In the know-how approach we developed in [229, 157], high-level intentions are resolved down to *atomic intentions* by means of structural knowledge represented as know-how statements. An intention I is atomic if it can be satisfied by the execution of a single action. In this process of refinement, the set of *options*, i. e., the set of possible sub-intentions for the realization of a given intention, has to be evaluated. On the basis of this evaluation, one of the best options is to be chosen.

We define a basic version of an intention operator based on a know-how base. A *know-how base* $\mathfrak{KH}$ consists of a set of *know-how statements*

$$(I, (s_1, \ldots, s_n), \{c_1, \ldots, c_m\}) \tag{3.5.1}$$

with a target goal I, a set of subgoals $(s_1, \ldots, s_n)$ and a set of conditions $\{c_1, \ldots, c_m\} \subseteq \mathcal{L}_{BS}$. The intuition of a know-how statement is that the intention I can be satisfied by achieving the sub-goals $(s_1, \ldots, s_n)$ in the given order if the conditions $\{c_1, \ldots, c_m\}$ are satisfied in the beginning.

Example 3.5.3. Imagine a cleaner robot that pursuits two goals: cleaning all rooms in a certain area, and maintaining a high battery level. It is crucial for the robot that it *knows how* to reach the charging station before beginning to clean the rooms at any time. Likewise, to effectively do its task the robot should be able to consider if it can return to the charging station in time when planning more than one step ahead.

Suppose the robot has the desire to clean all rooms in its area which is represented by cleaned_all. Suppose now that there are two rooms in its area, a hallway and a lounge, and that the robot shall verify that its battery is sufficiently charged before it starts cleaning these rooms. This knowledge can be captured by the know-how statement

$$\begin{aligned}\mathsf{kh}_1 \quad = \quad & (\mathsf{cleaned\_all},\ (\mathsf{cleaned\_hallway},\ \mathsf{cleaned\_lounge}),\\ & \{\mathsf{battery\_high}\}).\end{aligned}$$

The subgoals of this know-how statement can be further resolved by other know-how statements, until the subgoals are atomic intentions. Let $\mathfrak{KH}$ of the robot be given by $\mathfrak{KH} = \{\mathsf{kh}_1, \dots, \mathsf{kh}_5\}$:

$$\begin{aligned}
\mathsf{kh}_2 &= (\mathsf{cleaned\_hallway},(\mathsf{at\_hallway},\\
&\qquad \mathsf{vacuumed\_hallway}),\{\mathsf{bag\_empty}\})\\
\mathsf{kh}_3 &= (\mathsf{cleaned\_lounge},\\
&\qquad (\mathsf{ordered\_other\_robot\_to\_clean\_lounge}),\\
&\qquad \{\mathsf{other\_robot\_available}\})\\
\mathsf{kh}_4 &= (\mathsf{cleaned\_lounge},(\mathsf{at\_lounge},\\
&\qquad \mathsf{free\_lounge},\mathsf{vacuumed\_lounge}),\{\})\\
\mathsf{kh}_5 &= (\mathsf{free\_lounge},\\
&\qquad (\mathsf{people\_sent\_away}),\ \{\mathsf{at\_lounge}\})
\end{aligned}$$

The know-how statement kh_2 states that in order to clean the hallway the robot has first to go to it, and then it has to do the actual vacuuming. This statement can only be applied by the robot, if its vacuum cleaner bag is empty. The other statements are interpreted in the same way. Note that the robot has two alternatives for the intention clean_lounge: Given that the helper robot other_robot is present, our robot can order another robot to do the job for it. Also note, that the fulfillment of the intention at_lounge in kh_4 is a prerequisite for the fulfillment of the intention free_lounge in kh_5. The intentions

at_hallway, at_lounge, vacuumed_hallway,
vacuumed_lounge, ordered_other_robot_to_clean_lounge,
and people_sent_away

are atomic intentions that can be fulfilled by executing the corresponding atomic action, e. g., a move action or an action to send away all present persons from the place of work. ♢

For our basic instance of the agent model we define a basic version of the full know-how approach in the following. It keeps the presentation of the overall agent model clear and can be easily extended to the full approach. It still is expressive enough to model and capture many scenarios of, e. g., communicating agents.

We call a know-how statement *direct* if it has exactly one sub-target, and if this is atomic. The power set of the set of all direct know-how statements is denoted by $\mathcal{L}_{\mathfrak{KH},\mathfrak{b}}$. A know-how base that consists only of direct know-how statements is called a *basic know-how base* and denoted as $\mathfrak{KH}_\mathfrak{b}$.

Example 3.5.4. In the strike committee meeting example *Emma* knows how she can respond to the question of *Beatriz* whether she intends to participate in the *strike committee meeting*. Her basic know-how base might be the following. Note that for simplicity of presentation we slightly abuse notation by nesting predicates here.

$$\begin{aligned}\mathfrak{KH}_b = \{ &(\mathsf{answered\_query(attend\_scm)},\\ &\qquad \mathsf{answered}(\mathcal{B}, \mathsf{attend\_scm}), \mathsf{attend\_scm}),\\ &(\mathsf{answered\_query(attend\_scm)},\\ &\qquad \mathsf{answered}(\mathcal{B}, \neg\mathsf{attend\_scm}), \neg\mathsf{attend\_scm}),\\ &(\mathsf{answered\_query(attend\_scm)},\\ &\qquad \mathsf{answered}(\mathcal{B}, \mathsf{refuse}), \emptyset)\}\end{aligned}$$

This know-how base formalizes that satisfying the intention to answer the query for attend_scm can be accomplished by either answering attend_scm, if the agent believes that attend_scm holds, or by answering ¬attend_scm, if the agent believes that ¬attend_scm holds, or by refusing to answer. ♢

In the next section we present our agent model of a BDI agent with motivation and know-how.

3.5.3 *The BDI$^+$ Agent Model*

We refine the epistemic agent model to define a compound epistemic BDI$^+$ agent model. That is, we define an epistemic state that contains components for the beliefs, desires, intentions, motives and know-how of the agent, and a functional component for this epistemic state.

A *BDI$^+$ agent state* is a tuple $(\mathcal{K}_{\mathsf{BDI}^+}, \xi_{\mathsf{BDI}^+})$ with an epistemic state of the form*epistemic state!BDI$^+$*

$$\mathcal{K}_{\mathsf{BDI}^+} = \langle \mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathfrak{M}, \mathfrak{KH} \rangle$$

with the agent's beliefs $\mathfrak{B} \in \mathcal{L}_{\mathfrak{B}}$, a set of desires $\mathfrak{D} \in \mathcal{L}_{\mathfrak{D}}$, a set of intentions $\mathfrak{I} \in \mathcal{L}_{\mathfrak{I}}$, a set of motives $\mathfrak{M} \in \mathcal{L}_{\mathfrak{M}}$, and the agent's know-how base $\mathfrak{KH} \in \mathcal{L}_{\mathfrak{KH}}$. We denote the language of BDI$^+$ epistemic states as

$$\mathcal{L}_{\mathsf{BDI}^+} =_{def} \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{D}} \times \mathcal{L}_{\mathfrak{I}} \times \mathcal{L}_{\mathfrak{M}} \times \mathcal{L}_{\mathfrak{KH}}.$$

Given the $\mathcal{L}_{\mathsf{BDI}^+}$ language, the functions of a functional BDI$^+$ component $\xi_{\mathsf{BDI}^+} = (\circ_{\mathsf{BDI}^+}, \circ^a_{\mathsf{BDI}^+}, \mathsf{act}_{\mathsf{BDI}^+})$ have the following types:

$$\circ_{\mathsf{BDI}^+} : \mathcal{L}_{\mathsf{BDI}^+} \times \mathsf{Per} \to \mathcal{L}_{\mathsf{BDI}^+},$$

$$\circ^a_{\mathsf{BDI}^+} : \mathcal{L}_{\mathsf{BDI}^+} \times \mathsf{Act} \to \mathcal{L}_{\mathsf{BDI}^+} \text{ and}$$

$$\mathsf{act}_{\mathsf{BDI}^+} : \mathcal{L}_{\mathsf{BDI}^+} \to \mathsf{Act}$$

Agent components of the type of $\mathfrak{M}$ and $\mathfrak{KH}$ are usually assumed to be static. Here, we restrict our consideration of these components to the static case as well, our full consideration of these, including the dynamic treatment of them can be found in [159] and [157].

We structure the change operator into several sub-operations according to the components of the epistemic state and a set of percepts Per:

$$\circ_{\mathfrak{B}} : \mathcal{L}_{\mathfrak{B}} \times \mathsf{Per} \to \mathcal{L}_{\mathfrak{B}},$$

$$\circ_{\mathfrak{D}} : \mathcal{L}_{\mathfrak{M}} \times \mathcal{L}_{\mathfrak{B}} \to \mathcal{L}_{\mathfrak{D}} \text{ and}$$

$$\circ_{\mathfrak{I}} : \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{D}} \times \mathcal{L}_{\mathfrak{I}} \times \mathcal{L}_{\mathfrak{K}\mathfrak{H}} \to \mathcal{L}_{\mathfrak{I}}.$$

These operators can be generalized to domains and co-domains that are supersets of the original ones, as described in Section 3.3, such that they are defined for

$$\circ^{\dagger}_{\mathfrak{B}} : \mathcal{L}_{\mathsf{BDI}^{+}} \times \mathsf{Per} \to \mathcal{L}_{\mathsf{BDI}^{+}} \times \mathsf{Per},$$

$$\circ^{\dagger}_{\mathfrak{I}} : \mathcal{L}_{\mathsf{BDI}^{+}} \times \mathsf{Per} \to \mathcal{L}_{\mathsf{BDI}^{+}} \times \mathsf{Per} \text{ and}$$

$$\circ^{\dagger}_{\mathfrak{D}} : \mathcal{L}_{\mathsf{BDI}^{+}} \times \mathsf{Per} \to \mathcal{L}_{\mathsf{BDI}^{+}}.$$

The change operation can then be represented as

$$\circ_{\mathsf{BDI}^{+}} =_{def} \circ^{\dagger}_{\mathfrak{B}} \cdot \circ^{\dagger}_{\mathfrak{D}} \cdot \circ^{\dagger}_{\mathfrak{I}}.$$

That is, the state $\mathcal{K}_{\mathsf{BDI}^{+}} = \langle \mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathfrak{M}, \mathfrak{K}\mathfrak{H} \rangle$ is changed by a percept p such that $\mathcal{K}_{\mathsf{BDI}^{+}} \circ_{\mathsf{BDI}^{+}} p = \circ^{\dagger}_{\mathfrak{I}}(\circ^{\dagger}_{\mathfrak{D}}(\mathcal{K}_{\mathsf{BDI}^{+}} \circ^{\dagger}_{\mathfrak{B}} p))$.

The change operator for actions $\circ^{a}$ can be generalized in the same way.

$$\circ^{a}_{\mathfrak{B}} : \mathcal{L}_{\mathfrak{B}} \times \mathsf{Act} \to \mathcal{L}_{\mathfrak{B}},$$

$$\circ^{a}_{\mathfrak{D}} : \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{D}} \to \mathcal{L}_{\mathfrak{D}} \text{ and}$$

$$\circ^{a}_{\mathfrak{I}} : \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{I}} \to \mathcal{L}_{\mathfrak{I}}.$$

These operators can be generalized to domains and co-domains that are supersets of the original ones, as described in Section 3.3, such that they are defined for

$$\circ^{a,\dagger}_{\mathfrak{B}} : \mathcal{L}_{\mathsf{BDI}^{+}} \times \mathsf{Act} \to \mathcal{L}_{\mathsf{BDI}^{+}} \times \mathsf{Act},$$

$$\circ^{a,\dagger}_{\mathfrak{I}} : \mathcal{L}_{\mathsf{BDI}^{+}} \times \mathsf{Act} \to \mathcal{L}_{\mathsf{BDI}^{+}} \times \mathsf{Act} \text{ and}$$

$$\circ^{a,\dagger}_{\mathfrak{D}} : \mathcal{L}_{\mathsf{BDI}^{+}} \times \mathsf{Act} \to \mathcal{L}_{\mathsf{BDI}^{+}}.$$

The change operation can then be represented as

$$\circ^{a}_{\mathsf{BDI}^{+}} =_{def} \circ^{a,\dagger}_{\mathfrak{B}} \cdot \circ^{a,\dagger}_{\mathfrak{D}} \cdot \circ^{a,\dagger}_{\mathfrak{I}}.$$

That is, the state $\mathcal{K}_{\mathsf{BDI}^{+}} = \langle \mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathfrak{M}, \mathfrak{K}\mathfrak{H} \rangle$ is changed by actions Act such that $\mathcal{K}_{\mathsf{BDI}^{+}} \circ^{a}_{\mathsf{BDI}^{+}} a = \circ^{a,\dagger}_{\mathfrak{I}}(\circ^{a,\dagger}_{\mathfrak{D}}(\mathcal{K}_{\mathsf{BDI}^{+}} \circ^{a,\dagger}_{\mathfrak{B}} a))$.

The BDI^{+} agent cycle is illustrated in Figure 3.5.1. Note, that it is a concrete instance of the general agent cycle depicted in Figure 3.4.2 (Page 49). In the next section we conretise the agent model we just introduced and instantiate the functional component.

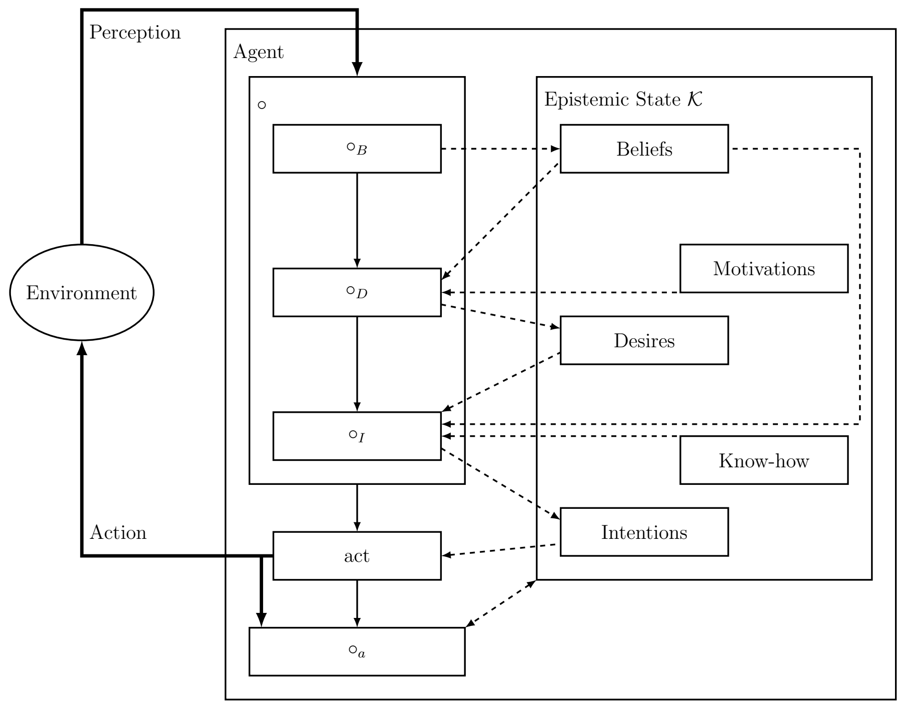


Figure 3.5.1: BDI$^+$ agent model and agent cycle - an instance of Figure 3.4.2

3.5.4 *Basic BDI$^+$ Agent Model*

In the following we define basic instantiations of the operators of the functional component. This way we define fully functional agents and only leave the knowledge representation formalism and the approach to evaluate possible actions to be specified. We call the agent model *basic* because we use basic versions of the operator specification and use basic versions of motives and know-how and specific operators for these. Extended versions of the operators and components can be found in other works of ourself [159, 157].

A *basic BDI$^+$ agent state* is a tuple $(\mathcal{K}_{\mathsf{BDI^+,b}}, \xi_{\mathsf{BDI^+,b}})$. The epistemic state is of the form

$$\mathcal{K}_{\mathsf{BDI^+,b}} = \langle \mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathfrak{M}_b, \mathfrak{KH}_b \rangle$$

with the agent's beliefs $\mathfrak{B} \in \mathcal{L}_{\mathfrak{B}}$, a set of desires $\mathfrak{D} \in \mathcal{L}_{\mathfrak{D}}$, a set of intentions $\mathfrak{I} \in \mathcal{L}_{\mathfrak{I}}$, a set of basic motives $\mathfrak{M}_b \in \mathcal{L}_{\mathfrak{M},b}$, and the agent's basic know-how base $\mathfrak{KH}_b \in \mathcal{L}_{\mathfrak{KH},b}$. We denote the language of basic BDI$^+$ epistemic states as

$$\mathcal{L}_{\mathsf{BDI^+,b}} =_{def} \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{D}} \times \mathcal{L}_{\mathfrak{I}} \times \mathcal{L}_{\mathfrak{M},b} \times \mathcal{L}_{\mathfrak{KH},b}.$$

Given the language $\mathcal{L}_{\mathsf{BDI^+,b}}$, the functions of a functional basic BDI$^+$ component $\xi_{\mathsf{BDI^+,b}} = (\circ_{\mathsf{BDI^+,b}}, \circ^a_{\mathsf{BDI^+,b}}, \mathsf{act}_{\mathsf{BDI^+,b}})$ are of the following types:

$$\circ_{\mathsf{BDI^+,b}} : \mathcal{L}_{\mathsf{BDI^+,b}} \times \mathsf{Per} \to \mathcal{L}_{\mathsf{BDI^+,b}}$$

$$\circ^{a}_{\mathsf{BDI}^+,b} : \mathcal{L}_{\mathsf{BDI}^+,b} \times \mathsf{Act} \to \mathcal{L}_{\mathsf{BDI}^+,b} \text{ and}$$

$$\mathsf{act}_{\mathsf{BDI}^+,b} : \mathcal{L}_{\mathsf{BDI}^+,b} \to \mathsf{Act}.$$

The set of all functional components of this type is denoted by $\mathfrak{F}_{\mathsf{BDI}^+,b}$.

Any model of a proactive agent needs some representation of the agent's goals, which guide its actions. In the BDI$^+$ model, as in all BDI models, the goals the agent has committed to are represented as intentions. Here, we represent these as formulae of the belief set language such that $\mathcal{L}_{\mathfrak{I}} \subseteq \mathcal{P}(\mathcal{L}_{BS})$. We denote the set of atomic intentions by $\mathsf{AtInt} \in \mathcal{L}_{\mathfrak{I}}$. We assume this action to be uniquely determined by the function

$$\alpha : \mathsf{AtInt} \to \mathsf{Act}. \tag{3.5.2}$$

The basic change operator is formed by the following composition:

$$\circ_{\mathsf{BDI}^+,b} =_{def} \circ^{\dagger}_{\mathfrak{B}} \cdot \circ^{\dagger}_{\mathfrak{D},b} \cdot \circ^{\dagger}_{\mathfrak{I},b}.$$

The belief change operator $\circ_{\mathfrak{B}}$ has to be a change operator for the knowledge representation formalism for which the model is instantiated. Since we leave the knowledge representation open here we cannot further define it at this point and come back to these in detail later.

In the following we instantiate the sub-operations $\circ_{\mathfrak{D},b}$ and $\circ_{\mathfrak{I},b}$. The functions $\circ^{\dagger}_{\mathfrak{D},b}$ and $\circ^{\dagger}_{\mathfrak{I},b}$ denote the extended versions of the functions $\circ_{\mathfrak{D},b}$ and $\circ_{\mathfrak{I},b}$, as in Section 3.4.

DESIRE OPERATOR $\circ_{\mathfrak{D},b}$ After the belief change operator has changed the agent's beliefs according to the input percept the *basic desire operator* $\circ_{\mathfrak{D},b}$ uses the belief-desire couplings to determine the current set of desires in the light of the new beliefs. It is a function of the type

$$\circ_{\mathfrak{D},b} : \mathcal{L}_{\mathfrak{M},b} \times \mathcal{L}_{\mathfrak{B}} \to \mathcal{L}_{\mathfrak{D}}.$$

It determines all belief-desire couplings whose condition is satisfied in the belief set formed from the changed beliefs of the agent. Formally it is defined as:

$$\circ_{\mathfrak{D},b}(\mathfrak{M}_b, \mathfrak{B}) =_{def} \{(D, \mu) \mid (D, \Phi, \mu) \in \mathfrak{M}_b, \Phi \subseteq \mathsf{Bel}_{\mathfrak{X}}(\mathfrak{B})\} \tag{3.5.3}$$

INTENTION OPERATOR $\circ_{\mathfrak{I},b}$ The intention operator uses the know-how of the agent to determine the options to satisfy the agent's intentions. We use a *basic options function* that uses a basic know-how base to determine the current options. It is of the form

$$\mathsf{opt}_b : \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{I}} \times \mathcal{L}_{\mathfrak{K}\mathfrak{H},b} \to \mathcal{L}_{\mathfrak{I}}.$$

It determines all subgoals that are contained in a know-how statement for the agent's current intention, if the conditions of the respective know-how statement are satisfied. Formally it is defined as follows:

$$\mathsf{opt}_b(\mathfrak{B}, \mathfrak{I}, \mathfrak{KH}_b) = \{s \mid (I, (s), \{c_1, \ldots, c_m\}) \in \mathfrak{KH}_b, \\ I \in \mathfrak{I}, \{c_1, \ldots, c_m\} \subseteq \mathsf{Bel}_{\mathcal{X}}(\mathfrak{B})\}. \tag{3.5.4}$$

We illustrate the basic options function in the following example.

Example 3.5.5. We consider the know-how base $\mathfrak{KH}_b$, which we defined in Example 3.5.4. For the set of intentions

$$\mathfrak{I} = \{\mathsf{answered\_query(attend\_scm)}\}$$

and the beliefs $\mathfrak{B} = \{\mathsf{attend\_scm}\}$ the options given by $\mathfrak{KH}_b$ are

$$\mathsf{opt}_b(\mathfrak{B}, \mathfrak{I}, \mathfrak{KH}_b) = \\ \{\mathsf{answered(attend\_scm)}, \mathsf{answered(refuse)}\}. \qquad \diamondsuit$$

For the evaluation of options the most common method is to assign utilities to the options [98]. A qualitative form of utilities [98, 248] is given by a *preference relation on options* $\preceq_{(\mathcal{K},\mathcal{E})}$. Determining the utility/preference of an action can be complex and dependent on multiple criteria. Here, we assume that the set of options is finite, and that it is evaluated with respect to the current epistemic state by the *evaluation operator*. It takes the agent's beliefs, desires and intentions, and a set of intentions as its input and outputs a partial order on the input set of intentions, i. e., it satisfies reflexivity, antisymmetry and transitivity. Formally, let $\mathfrak{B}$ be a set of beliefs, $\mathfrak{D}$ a set of desires, $\mathfrak{I}$ and $\mathfrak{I}'$ sets of intentions, then

$$\mathsf{eval} : (\mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathfrak{I}') \mapsto \mathcal{P}(\mathfrak{I}' \times \mathfrak{I}').$$

The result of the eval operator is a preference relation on the set of current options of the agent with respect to an epistemic state $\mathcal{K}_{\mathsf{BDI}^+} = \langle \mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathfrak{M}, \mathfrak{KH} \rangle$, such that:

$$\preceq_{(\mathcal{K}_{\mathsf{BDI}^+}, \mathcal{E}_{\mathsf{BDI}^+,b})} =_{def} \mathsf{eval}(\mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathsf{opt}_b(\mathfrak{B}, \mathfrak{I}, \mathfrak{KH}_b)). \tag{3.5.5}$$

If the options under consideration are atomic, then they directly correspond to actions. Hence the evaluation of actions represents a special case of the evaluation of options. In the case of the evaluation of atomic intentions we identify the atomic intentions with the corresponding actions. We define the preferences on atomic intentions as options by defining preferences on the corresponding actions.

One of the maximally preferred options according to

$$\preceq_{(\mathcal{K}_{\mathsf{BDI}^+}, \mathcal{E}_{\mathsf{BDI}^+,b})}$$

is selected and committed to. We define the max function to determine the maximal elements of a set X with respect to a preorder, i. e., a reflexive and transitive binary relation, $\preceq \subseteq \mathcal{P}(X \times X)$. We define the strict order $\prec$ from $\preceq$ as:

$$x \prec y \text{ if and only if } x \preceq y \text{ and } y \not\preceq x.$$

The max function is defined as:

$$\max_{\preceq}(X) =_{def} \{x \in X \mid \text{there is no } x' \in X \text{ such that } x' > x\} \quad (3.5.6)$$

Note that if X is finite and $X \neq \emptyset$, then $\max_{\preceq}(X) \neq \emptyset$.

If there is more than one maximally preferred option, a decision for one of the maximally preferred options has to be made. To this end we assume a *general selection function* σ that maps any nonempty set X to one of its elements such that

$$\sigma(X) \in X \text{ if } X \neq \emptyset. \quad (3.5.7)$$

Since all elements of X are maximally preferred, there is no information that could guide this selection. Hence this selection might, e. g., be at random. In the following we only consider non-empty sets of intentions.

The basic intention operator

$$\circ^b_{\mathfrak{I}} : \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{D}} \times \mathcal{L}_{\mathfrak{I}} \times \mathcal{L}_{\mathfrak{K}\mathfrak{H}} \to \mathcal{L}_{\mathfrak{I}}$$

determines the maximally motivated desires. To formalize the set of maximally motivated desires we define the following function, with $\leqslant$ being the order on real numbers:

$$\max_{\mu}(\mathfrak{D}) =_{def} \{\phi \mid (\phi, \mu) \in \mathfrak{D}, \mu \in \max_{\leqslant}(\{\mu \mid (\phi, \mu) \in \mathfrak{D}\})\}.$$

We further assume that there exists at least one belief-desire coupling that is satisfied, such that it is guaranteed that there is at least one maximally motivated desire. This assumption can, e. g., be guaranteed by defining a belief-desire coupling with an empty condition and the desire to be idle. The agent selects one of them by the selection function σ and determines the options for it. We also assume that there exists at least one option for each desire. The agent evaluates the options by means of the eval function, and determines the maximally preferred options, and selects one of these. All together this results in the definition of the intention operator as given in Procedure 3.5.1.

As stated before we assume that an agent only has one intention at a time. Then, the act function can be defined to return the action corresponding to this intention, i. e., $\mathfrak{I} \cap \mathsf{AtInt} = \{I\}$ and

$$\mathsf{act}(\mathcal{K}_{\mathsf{BDI}^+}) =_{def} \alpha(I). \quad (3.5.8)$$

Procedure 3.5.1 Basic intention operator $\circ^{b}_{\mathfrak{I}}$

```
Input: $\mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathfrak{K}\mathfrak{H}$
Output: Set of new intentions $\mathfrak{I}'$
1: $\mathsf{currentInt} := \{\sigma(\max_{\mu}(\mathfrak{D}))\}$
2: $\mathsf{options} := \mathsf{opt}_b(\mathfrak{B}, \mathsf{currentInt}, \mathfrak{K}\mathfrak{H}_b)$
3: $\preceq_{(\mathcal{K}_{\mathsf{BDI}^+}, \mathcal{E}_{\mathsf{BDI}^+})} := \mathsf{eval}(\mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathsf{options})$
4: $\mathfrak{I}' := \{\sigma(\max_{\preceq_{(\mathcal{K}_{\mathsf{BDI}^+}, \mathcal{E}_{\mathsf{BDI}^+})}}(\mathsf{options}))\}$
```

Example 3.5.6. We continue the strike committee meeting scenario of Example 3.5.5 and assume *Emma* was just asked by *Beatriz* if she attends the scm. Her current maximally motivated desire is to answer this question, i. e., $\mathsf{answered(attend\_scm)}$. The evaluation of the two options from Example 3.5.5 results in a preference relation such that

$$\mathsf{answered(dont\_know)} \preceq_{(\mathcal{K}_{\mathsf{BDI}^+}, \mathcal{E}_{\mathsf{BDI}^+,b})} \mathsf{answered(attend\_scm)}.$$

The only maximal element is thus the atomic intention

$$\mathsf{answered(attend\_scm)}$$

and the action that directly satisfies this atomic intention is the speech act

$$\alpha(\mathsf{answered(attend\_scm)}) = \langle \mathit{Emma}, \mathit{Beatriz}, \mathsf{answer}, \mathsf{attend\_scm}\rangle. \diamondsuit$$

The belief change operator $\circ^{a}_{\mathfrak{B}}$ has to be a change operator for the formalism for which the model is instantiated. Since we do not instantiate the knowledge representation here, we cannot further define it at this point and come back to these in detail later. For our basic instantiation we do not need the $\circ^{a}_{\mathfrak{D}}$ and $\circ^{a}_{\mathfrak{I}}$ operations since we focus on agents with only one goal at the same time and designed the $\circ_{\mathfrak{D}}$ and $\circ_{\mathfrak{I}}$ operators such that they determine this goal in each cycle. Hence, a basic change operator for actions comprises only a belief change operator for actions of the type $\circ^{a,\dagger}_{\mathfrak{B}} : \mathcal{L}_{\mathsf{BDI}^+,b} \times \mathsf{Act} \to \mathcal{L}_{\mathsf{BDI}^+,b}$:

$$\circ^{a}_{\mathsf{BDI}^+} =_{def} \circ^{a,\dagger}_{\mathfrak{B}}. \tag{3.5.9}$$

The functional component $\mathcal{E}_{\mathsf{BDI}^+,b}$ we defined in this section is completely defined with the exception of the knowledge representation specific operators $\circ_{\mathfrak{B}}$, $\mathsf{Bel}_{\mathcal{X}}$, and eval. These are dependent on the considered application domain and have to be instantiated together with the actual knowledge representation and the domain knowledge of the agent.

In the following section we extend on our definition of belief operators for agents and formalize belief operator families, which are sets of belief operators with an order relation.

3.6 BELIEF OPERATORS FOR EPISTEMIC AGENTS

In Section 3.2 we introduced belief operators that determine the belief set of a given epistemic component (Definition 3.2.2, Page 40). In this section we extend the concept of belief operators and define families of ordered belief operators. We present an abstract model for belief operator families and discuss general properties for them. Then, we concretize it for classes of non-monotonic reasoning formalisms on the basis of extension families [173]. The belief operator determines which plausible inferences are drawn from the incomplete information given by an epistemic component. There are different ways to deal with incomplete information and to determine the inferences, which results in different belief operators. Different belief operators can for instance be more or less credulous such as the two operators of *skeptical* and *credulous* inference which are used in several formalisms [55], or the diverse types of extensions used in argumentation theory [20]. In general we consider a set of belief operators that can be ordered, e. g., by their credulity. Different agents might use the same knowledge representation formalism but different belief operators, and each agent might use different belief operators in different situations.

Example 3.6.1. In the strike committee meeting example *Emma*'s boss *Beatriz* might use a very credulous belief operator with respect to her employees attending strike committee meetings while *Emma*'s colleagues use a skeptical one. This means that given the same information *Beatriz* might infer that *Emma* intend to attend the *strike committee meeting* while her colleagues do not infer it. $\diamondsuit$

We formalize different types of belief operators in an ordered *family of belief operators* from which an operator can be chosen.

Definition 3.6.2 (Belief Operator Family). A *belief operator family* is a pair $(\Xi, \preceq_{\mathsf{bel}})$ consisting of a *set of belief operators* Ξ of the form $\mathsf{Bel} : \mathcal{P}(\mathcal{L}_E) \to \mathcal{P}(\mathcal{L}_{BS})$ and a preorder, i. e., a reflexive and transitive binary relation, $\preceq_{\mathsf{bel}}$ on Ξ.

The definition of a family of belief operators abstracts from the underlying formalism and inference mechanism. It can be applied to a wide range of formalisms, from purely qualitative ones to quantitative ones. In the following, we define properties for belief operator families, i. e., properties on the belief operators of the belief operator family and properties of the preorder $\preceq_{\mathsf{bel}}$ of the belief operator family on the belief operators.

A rich landscape of properties of (non-monotonic) inference relations has been developed in several important works, e. g., [104, 173, 43]. Some of these inference relations and formalisms are semantically very strong in terms of the satisfaction of desirable properties. However, these formalisms are often not used in practice and the formalisms used in practice often do not satisfy the properties or do meet the assumptions need to apply the properties. Examples of such practical approaches are answer set programming [112] and structured argumentation systems [105, 192]. Hence in the following we require weaker assumptions and generalize basic properties to allow for a bigger variety of formalisms. A key difference of our setting from the typically considered setting is that we do not require a belief operator to satisfy categorical matching, i. e., that the domain and co-domain are equal. That is, we allow the languages of the epistemic component and the belief set to be different. We also do not demand that belief sets are deductively closed. We do not intent to cover the whole landscape of properties, but focus on the definition of a basic set of properties and consider these with respect to a propositional instance, and extension based formalisms in general and an ASP instance in particular.

We assume two important notions to be given by the respective formalism under consideration. We assume an adequate consequence operator for strict inferences from a given epistemic component, denoted by $\mathsf{Cn}^{\mathsf{strict}}$. For the formalism $\mathtt{prop}$ this is the deductive closure for propositional logic $\mathsf{Cn}^{\mathsf{prop}}$ as defined in Section 2.2.

ASP Instance 2. For our ASP instance we define a strict consequence operator for ASP as $\mathsf{Cn}^{\mathsf{asp}}(P) = \cap AS(P^{\mathsf{strict}}(P))$. The strict program with respect to a given program is given by $P^{\mathsf{strict}}(P) =_{def} \{r \in P \mid \mathcal{B}(r)^{-} = \emptyset\}$, as defined in Section 2.4 (Page 25). Note that strict programs have at most one answer set.

Further, we assume that a notion of consistency is defined for epistemic components and for belief sets. The following instantiation for ASP shows an example of a formalism that needs two different notions of consistency.

ASP Instance 3. An ASP epistemic component $P \subseteq \mathcal{L}_{At}^{asp}$, a program, is consistent if $AS(P) \neq \emptyset$. An ASP belief set S, a set of literals, is consistent if it does not contain any complementary literals L and $\neg L$, as defined in Section 2.4.

We define the following basic set of desirable properties of a belief operator Bel:

CONSISTENCY$_{\mathsf{Bel}}$ For all consistent epistemic components $E \subseteq \mathcal{L}_E$ the resulting belief set $\mathsf{Bel}(E) \subseteq \mathcal{L}_{BS}$ is consistent.

SUPRALITY$_{\mathsf{Bel}}$ For all $E \subseteq \mathcal{L}_E$ it holds that $\mathsf{Cn}^{\mathsf{strict}}(E) \subseteq \mathsf{Bel}(E)$.

RIGHT-WEAKENING$_{\mathsf{Bel}}$ For all $E \subseteq \mathcal{L}_E$ it holds that if $\phi \in \mathsf{Bel}(E)$ and $\psi \in \mathsf{Cn}^{\mathsf{prop}}(\phi)$, then $\psi \in \mathsf{Bel}(E)$.

RELATIVE-LEFT-ABSORPTION$_{\mathsf{Bel}}$ For all $E \subseteq \mathcal{L}_E$ it holds that $\mathsf{Bel}(E) = \mathsf{Cn}^{\mathsf{prop}}(\mathsf{Bel}(E)) \cap \mathcal{L}_{BS}$.

Note that the last two properties are only applicable for formalisms for which the $\mathsf{Cn}^{\mathsf{prop}}$ operator can be applied to belief sets.

The *Consistency*$_{\mathsf{Bel}}$ property is specific to our approach since it uses two notions of consistency, one for an epistemic component and one for a belief set. The property *Suprality*$_{\mathsf{Bel}}$ is a generalization of the property *Supraclassicality* [173] and coincides with the latter for propositional logic. The property *Right-Weakening*$_{\mathsf{Bel}}$ is the same as the property with the same name in, e. g., [173]. We just adapted it to our notation. The property of *Relative-Left-Absorption*$_{\mathsf{Bel}}$ is a variant of the *Left-Absorption* property [173], the intersection with $\mathcal{L}_{BS}$ is not present in the latter. Both properties are equivalent if $\mathcal{L}_E = \mathcal{L}_{BS}$. If $\mathcal{L}_{BS}$ is a set of literals, as is the case for ASP, then the set of propositional consequences of the belief set is equal to the original belief set. This is formalized in the following example.

Example 3.6.3. For any $BS \subseteq \mathcal{L}_{BS} = \mathsf{Lit}$ it holds that:

$$\mathsf{Cn}^{\mathsf{prop}}(BS) \cap \mathcal{L}_{BS} = BS. \qquad \diamondsuit$$

After the definition of properties of the belief operators, we introduce a property of the relation $\preceq_{\mathsf{bel}}$. In particular we formalize a set-inclusion-based credulity property. The intuition is that a belief operator Bel that is at least as credulous as another belief operator Bel' infers all beliefs that Bel' infers, and possibly more. This is formalized as:

CREDULITY$_{\preceq_{\mathsf{bel}}}$ If $\mathsf{Bel} \preceq_{\mathsf{bel}} \mathsf{Bel}'$, then for all $E \in \mathcal{L}_E$ it holds that $\mathsf{Bel}(E) \subseteq \mathsf{Bel}'(E)$.

If $\preceq_{\mathsf{bel}}$ satisfies *Credulity*$_{\preceq_{\mathsf{bel}}}$, we call $\preceq_{\mathsf{bel}}$ a *credulity order* and read $\mathsf{Bel} \preceq_{\mathsf{bel}} \mathsf{Bel}'$ as Bel' is at least as *credulous* as Bel.

We say that a belief operator family satisfies a property for belief operators, e. g., *Consistency*$_{\mathsf{Bel}}$, if all belief operators of the family satisfy the respective property. We say that a belief operator family satisfies a property for an order if its order on the belief operators satisfies it.

In the following we first define an exemplary belief operator family based on propositional logic. Then we elaborate on belief operator families based on extension families, and exemplify these by answer set programming.

Propositional Instance 1. We set $\mathcal{L}_E = \mathcal{L}_{BS} = \mathcal{L}^{\mathsf{prop}}_{At}$ as the standard propositional language over a finite alphabet At, as introduced in Section 2.2 (Page 21). We define a set of belief operators

$$\Xi^{\mathsf{prop}} = \{\mathsf{Bel}_p \mid p \in (0.5, 1]\}$$

indexed by the threshold parameter p. Each operator calculates the agent's certainty in the truth of a formula $\phi \in \mathcal{L}$ as the ratio of its models. The operator can be seen as accepting every formula as true that holds in at least $p \cdot 100$ percent of those models (given by the model operator Mod):

$$\mathsf{Bel}_p(E) = \{\phi \in \mathcal{L}^{\mathsf{prop}}_{At} \mid r(\phi, E) \geqslant p\}$$
$$\text{with } r(\phi, E) = \frac{|\mathsf{Mod}(\phi) \cap \mathsf{Mod}(E\})|}{|\mathsf{Mod}(E\})|} \qquad (3.6.1)$$

The preorder is given as

$$\mathsf{Bel}_p \preceq^{\mathsf{prop}}_{\mathsf{bel}} \mathsf{Bel}_{p'} \text{ if and only if } p' \leqslant p.$$

A propositional epistemic component $E \subseteq \mathcal{L}^{\mathsf{prop}}_{At}$ is *consistent* if it has at least one model, i. e., $\mathsf{Mod}(()E) \neq \emptyset$. Note that the belief operators presented here are only defined for consistent epistemic components. A propositional belief set $BS \subseteq \mathcal{L}^{\mathsf{prop}}_{At}$ is *consistent* if there does not exist $\phi \in BS$ such that $\neg\phi \in BS$. The condition for consistency of a belief sets is weaker than the consistency condition for epistemic components. They coincide if the belief set is deductively closed. The propositional belief operator family we consider here produces belief sets that are not deductively closed, but all operators satisfy *Consistency* as the following proposition shows.

Proposition 3.6.4. The pair $(\Xi^{\mathsf{prop}}, \preceq^{\mathsf{prop}}_{\mathsf{bel}})$ is a belief operator family that satisfies the properties:

Consistency$_{\mathsf{Bel}}$, *Suprality*$_{\mathsf{Bel}}$, *Right-Weakening*$_{\mathsf{Bel}}$, and *Credulity*$_{\preceq_{\mathsf{bel}}}$.

Proof. See Appendix A.1.1 on Page 223. □

In the following we consider the construction of belief operator families on the basis of extension families as introduced in [173]. A variety of approaches to non-monotonic reasoning are directly based on a notion of extensions, e. g., Reiter's default logic [202], Poole's default logic [191], Answer Set Programming [114], Truth Maintenance Systems [80], or argumentation formalisms [83]. Other approaches without an explicit notion of extensions can be reformulated to fit into the extension perspective, for details see [173].

Here, we combine the formalization of extension families from [173] with our concept of epistemic components and belief operators, and the corresponding languages. The original formulation of extension families is as follows:

Definition 3.6.5. [173] Suppose a 'skeptical' inference operation $C : \mathcal{P}(L) \to \mathcal{P}(L)$ by setting $C(A) = \cap\mathsf{ext}(A)$, where $\mathsf{ext}(A)$ is a family of sets of propositions such that $Cn(A) \subseteq e$ for all $e \in \mathsf{ext}(A)$. We call the $e \in \mathsf{ext}(A)$ *extensions* of A under ext, and ext itself an *extension family function*.

In our terminology an inference operation is a belief operator of the form $\mathsf{Bel} : \mathcal{P}(\mathcal{L}_E) \to \mathcal{P}(\mathcal{L}_{BS})$ that is applied to an epistemic component and produces a belief set. Further, different logical formalisms might use different epistemic component and belief set languages. An *extension* $e \subseteq \mathcal{L}_{BS}$ resembles one possible set of beliefs given an $E \subseteq \mathcal{L}_{BS}$. Further, we do not demand that an extension is deductively closed. We exemplify this definition by an instantiation for ASP.

ASP Instance 4. For our ASP instance we represent an epistemic component as an extended logic program $P \subseteq \mathcal{L}^{\mathsf{asp}}_{At}$. The extension family function is the answer set function *AS*, which produces the set of answersets $AS(P) \subseteq \mathcal{P}(\mathsf{Lit})$.

In the following we define properties of extension families based on the properties discussed in [173]. To this end, we assume an appropriate union operator $\dot\cup$ for the union of an epistemic component $E \subseteq \mathcal{L}_E$ with an extension $e \subseteq \mathcal{L}_{BS}$ such that $E \dot\cup e \subseteq \mathcal{L}_E$. For formalisms for which $\mathcal{L}_E = \mathcal{L}_{BS}$ this operator is the standard union operator for sets. For other formalisms this might be different; for example in the following instantiation of it for ASP.

ASP Instance 5. For a program P and a set of literals L we define the union operator such that it converts a set of literals into a program comprising these literals as facts, i. e., $P \dot\cup^{\mathsf{asp}} L =_{def} P \cup \{l. \mid l \in L\}$.

The properties of extension family functions we consider in this work are formalized as follows:

SUPRALITY$_{\mathsf{ext}}$: $\mathsf{Cn}^{\mathsf{strict}}(E) \subseteq e$ for all $e \in \mathsf{ext}(E)$ and all $E \subseteq \mathcal{L}_E$.

RELATIVE-LEFT-ABSORPTION$_{\mathsf{ext}}$: $e = \mathsf{Cn}^{\mathsf{prop}}(e) \cap \mathcal{L}_{BS}$ for all $e \in \mathsf{ext}(E)$ and all $E \subseteq \mathcal{L}_E$.

CLOSURE$_{\mathsf{ext}}$: $e = \mathsf{Cn}^{\mathsf{strict}}(E \cup e)$ for all $e \in \mathsf{ext}(E)$ and all $E \subseteq \mathcal{L}_E$.

CONSISTENCY$_{\mathsf{ext}}$: Each $e \in \mathsf{ext}(E)$ is consistent for all consistent $E \subseteq \mathcal{L}_E$.

SEPARATION$_{\mathsf{ext}}$: $e \not\subset e'$ for all $e, e' \in \mathsf{ext}(E)$ and all $E \subseteq \mathcal{L}_E$.

As stated in Definition 3.6.5, in [173] all extension families are assumed to satisfy *Suprality*$_{\mathsf{ext}}$ with for propositional logic, i. e., *Supraclassicality*. As noted before, we do not require this in general since this would exclude formalisms such as ASP, which do not satisfy it. Therefore we define a version of the *Suprality*$_{\mathsf{ext}}$ postulate that makes use of an adequate consequence operator for the considered formalism. For the same reason we introduce the postulate *Relative-Left-Absorption*$_{\mathsf{ext}}$ for extension families in the same way we did for belief operators. We propose the *Closure*$_{\mathsf{ext}}$ postulate which demands that any extension is closed under consequences from the epistemic component augmented by the extension under consideration. This is a common property for many extension-based formalisms. The properties *Consistency*$_{\mathsf{ext}}$ and *Separation*$_{\mathsf{ext}}$ are used in the same form in [173] and are satisfied by most but not all formalisms. For our ASP instance we can show the following.

Proposition 3.6.6. The ASP extension family function *AS* satisfies the postulates *Suprality*$_{\mathsf{ext}}$, *Relative-Left-Absorption*$_{\mathsf{ext}}$, *Closure*$_{\mathsf{ext}}$, *Consistency*$_{\mathsf{ext}}$ and *Separation*$_{\mathsf{ext}}$.

Proof. See Appendix A.1.1 on Page 224. □

As mentioned before, the formalization in [173] is limited to skeptical inference operators, i. e., the set of inferences is formed by the intersection of all extensions. Here, we formalize that extension based belief operators result from the application of a *combination operator*, e. g., the intersection operator, $\mathsf{comb} : \mathcal{P}(\mathcal{L}_{BS}) \to \mathcal{L}_{BS}$ to an extension family function ext. Given a set of combination operators Γ and a set of extension family functions $\mathcal{E}$ a set of belief operators can be constructed as

$$\Xi(\mathcal{E}, \Gamma) = \{\mathsf{ext} \cdot \mathsf{comb} \mid \mathsf{comb} \in \Gamma, \mathsf{ext} \in \mathcal{E}\}.$$

Hereby, the operator $\cdot$ denotes the function composition and is defined as in Equation (3.4.2) (Page 48), i. e.,

$$\mathsf{ext} \cdot \mathsf{comb}(E) = \mathsf{comb}(\mathsf{ext}(E)).$$

For our ASP instance we use the singleton set of extension family functions $\mathcal{E}_{\mathsf{asp}} =_{def} \{AS\}$ and the commonly considered combination operators $\Gamma = \{\cap, \cup\}$. These choices result in the following belief operator family.

ASP Instance 6 (ASP Belief Operator Family). The *ASP belief operator family* is given by $\Xi_{\Gamma}^{\mathsf{asp},\{AS\}} = \{\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}\}$, $\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}(P) = \cup AS(P)$, $\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}(P) = \cap AS(P)$ and

$$\preceq^{\mathsf{asp}}_{\mathsf{bel}} = \{(\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}), (\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}), (\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}})\}.$$

For short we also write $\Xi^{\mathsf{asp}} = \Xi_{\Gamma}^{\mathsf{asp},\{AS\}}$.

For the ASP belief operator family we just defined we can show that the following properties hold.

Proposition 3.6.7. For the *ASP belief operator family* the following results hold:

1. $\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}$ satisfies *Suprality*$_{\mathsf{Bel}}$, *Consistency*$_{\mathsf{Bel}}$ and *Relative-Left-Absorption*$_{\mathsf{Bel}}$.
2. $\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}$ satisfies *Suprality*$_{\mathsf{Bel}}$, *Relative-Left-Absorption*$_{\mathsf{Bel}}$.

3. $\preceq_{\mathsf{bel}}$ satisfies $\mathit{Credulity}_{\preceq_{\mathsf{bel}}}$.

Proof. See Appendix A.1.1 on Page 225. □

The set of extension family functions can be extended by other semantics for extended logic programs [99], and extensions of it such as preferences [59] or sequences of programs [151, 8]. Combination operators can be build from the intersection or union of a subset of the set of extensions. The subset of extensions to be considered can be determined by preference information, for answer set programming several semantics for determining the preferences on answer sets have been developed, e. g., [58, 59, 181]. Argumentation theory is another well developed example of extension based formalisms. It features a particularly rich landscape of extension types [21].

3.7 RELATED WORK

The vast majority of agent models in the literature are either abstract and for instance formulated in a modal logic, e. g., interpreted systems [136] or the theoretical formulation of BDI agents [194]; or they are practical agent programming languages, see [51] for an overview. The BDI model is also realized in many practical models. There is, however, a big gap between the abstract consideration of agent models and practical systems in general, and the modal logic formulation of BDI agents and practical BDI systems in particular, as noted for instance in [200]. Several works are dedicated to narrow this gap, e. g., [182, 164, 195]. They intend to do so by using logic programming languages for the specification and execution of agents. Our model can also be instantiated with logic programming languages, and in fact we define instantiations based on answer set programming. In contrast to those works, our model of epistemic agents is not fixed to a BDI agent architecture or particular knowledge representation formalisms.

The work on agent models that comes closest to our approach is the KGP (Knowledge, Goals and Plan) model of agency [139, 140]. The internal state of KGP agents comprises a composed knowledge base, a goals, and a plan component, analogously to the components of the BDI model. The KGP model was motivated by the gap between the theory and practice of the BDI model and was designed to use logical approaches for the design of agents. A KGP knowledge base consists of different modules to support different reasoning capabilities such as planning or deliberation (called *goal-decision* in [139]). The agent cycle is controlled by a context-sensitive cycle theory which determines the execution order of a set of transitions. The transitions on the KGP model correspond to the sub-functions of the functional component in our model, the cycle theory to the composition of these. In our case, this composition is realized by a function composition that does not change during runtime of an agent. The KGP model relies on abductive logic programing, logic programming and an event calculus approach to program an agent. It does not directly allow the use of other knowledge representation formalisms, other modules of the knowledge base, other capabilities or transitions. Thus the KGP model can be seen as a partial instantiation of our epistemic agent model that constraints the structure of the epistemic state and the functional component to the KGP model. Solely the context sensitive agent cycle cannot be modeled directly in our framework, since we use a fixed composition of the functional component. Indirectly, a context sensitive cycle could be realized inside the functions of the functional component.

Another flexible agent model with a focus on the use of (non-monotonic) knowledge representation has been proposed in [138]. It is an abstract component based agent model that is based entirely on argumentation theory. It is called *integrated argumentation-based agent architecture* (ABA). Each component is an argumentation system and the interaction between these components is organized by an argumentation system as well. No other knowledge representation formalisms are considered and also no instantiations of the abstract framework. In our framework it is possible to instantiate all components by argumentation formalisms as well, but the definition of an agent cycle has to be defined in terms of a composed functional component.

The BDI^+ model we presented is similar to various other BDI architectures [51]. Unique to our approach is the combination and integration of motivation and know-how, its flexibility with respect to the possibility to add further components and that it can be instantiated for a variety of knowledge representation formalisms.

Different types of belief operators and their properties have been subject of various works e. g., [104, 173, 43]. The formulation and usage of ordered families of such operators in the generality we present them here has, to our knowledge, not been proposed before. We have shown that and how we can incorporate the formalization of extension families [173] in our formalization of belief operators. The propositional belief operator family we present on Page 63 is based on the relation of the number of models in which the formula is satisfied, to the total number of models. The same idea is used in the *random worlds* approach [13] to define *degrees of belief* for formulae given a knowledge base. These degrees of belief are parametrized with the size of the world and a tolerance vector. Our definition uses a threshold p for this relation to define belief operators, also we only consider propositional epistemic component in contrast to the statistical knowledge bases considered in [13].

3.8 CONCLUSION

In this chapter we developed our general model of epistemic agents. It generalizes diverse agent architectures and combines an abstract agent model with composed agent architectures for the realization of agents. It can be used to capture other agent architectures from purely deductive agents with a monolithic epistemic state to structured ones such as the beliefs, desires, intentions (BDI) model. We stressed that a communicating agent has to have the ability to reflect the anticipated changes of its actions in its epistemic state to perform well and made this process explicit by the introduction of a change operator for actions $\circ^{a}$ a part of the agent cycle. Moreover, we developed an extended BDI agent model, named BDI^{+} model, on the basis of our general epistemic agent model that incorporates motivation and know-how. We instantiated the BDI^{+} model with basic operators and motivation and know-how components. This model can be fully instantiated by fixing the knowledge representation formalism and the evaluation function for the action selection.

In our epistemic agent framework we introduced epistemic components of an epistemic state. Epistemic components contain the information that is used by a belief operator to form a belief set. We elaborated on such belief operators and defined a general notion of belief operator families whose operators can be ordered based on how credulous the inference behavior is. The assumptions we make are general enough to capture a big variety of knowledge representation formalisms. We defined desirable properties for belief operators and ordered families of these. Our main focus hereby are non-monotonic reasoning formalisms that are based on extension families [173]. We further instantiated the knowledge representation formalisms for our epistemic agent model for propositional logic and answer set programming (ASP) [112] and formally showed which properties these instances satisfy.

The work presented in this chapter provides a framework for the development and the analysis of epistemic agents that are based on non-monotonic reasoning. It is general enough to consider abstract properties of agent systems and provides instantiations for concrete agent models. As one of these we presented our BDI^{+} agent model that extends BDI agent models by a motivation and a know-how component. We presented basic versions of the operators of the model.

4 BELIEF CHANGE OPERATIONS

While we defined the composition of change operators for epistemic states in Chapter 3 we left the details of the operators for epistemic components unspecified. In this chapter, we develop a general concept of how the percepts of an agent are processed and used to change a particular epistemic component. We develop a model of compound belief change operations in the spirit of our general agent model presented in Section 3. Further, we relate it to existing approaches in belief change theory and develop new operators to match the requirements of our considered scenario. Hereby we extend the state-of-the-art in belief revision theory by extending existing operations and by combining them with non-monotonic logics in two ways; by developing operators by the use of, and for, non-monotonic logics. Moreover we narrow the gap between operators considered in belief change theory and operators that can be used in practical multiagent systems. We do so by integrating operators from belief change theory in our agent framework by our compound change operators, by implementing a selection function that enable the agent to evaluate new information, and by applying belief change theory to answer set programs, which are highly effective and used in practice.

We define a compound change operator that includes in particular an operator for the evaluation of incoming information. It has to determine if and to which degree, or in which form, the input shall be accepted. A general framework for this two step revision process has been presented in [96] under the name of *selective revision*. It defines a selective revision operator (called *outer revision operation* in the following) to comprise a selection function that determines the part and form of the input that shall be accepted, and a prioritized belief revision operator (called *inner revision operation* in the following) that revises the epistemic component with the output of the selection function, giving priority to the latter. This framework fits perfectly into our agent model. However, it has not been instantiated before we first presented our instantiation of it in [160], which forms the basis for the approach we present in the following. We generalize the original framework to inputs consisting not only of one single sentence, but consisting of a set of sentences. Operations that take sets of sentences as input are called *multiple revision operations*.

The construction and many of the results we present in this chapter are general for many forms of epistemic components. For more concrete properties and constructions we have to fix a particular formalization of epistemic components. We consider in particular the change of non-deductively closed belief bases [142, 183, 122, 127, 128, 119]. We denote an epistemic component that is a belief base by $\mathcal{B}$. Especially for resource-bounded reasoning a finite belief base representation is essential. Moreover, belief base change theory complies well with the intuitions on how humans represent and change their beliefs [122, 127]. We instantiate the selective revision framework by the use of an approach of deductive argumentation for propositional belief bases, and apply belief base revision theory to answer set programs.

Also for the definition of change operators for answer set programming a multiple revision operator for answer set programs is needed. We apply the theory of base revision [127] to non-monotonic answer set programs, which has not been presented before our first presentation of ours in [154]. We reconsider the postulates from base revision for propositional belief bases with respect to non-monotonic belief bases in general, and answer set programming in particular. We modify and extend the set of postulates to capture the characteristics of non-monotonic formalisms. We compare the resulting postulates with postulates that have been developed specifically for change operations to answer set programs. Moreover, besides the declarative description, we develop constructions base revision operators by extending existing constructions for the propositional case, and we show that our constructions satisfy all of the desirable properties we consider.

The remainder of this chapter is structured as follows. In Section 4.1 we introduce our general composition of a change operator. In Section 4.2 we define properties of the selection function of the change operator and present our instantiation on the basis of the selective revision framework and argumentation theory. In Section 4.3 we elaborate on the inner revision operation of the change operator for non-monotonic logics and ASP in particular. In Section 4.4 we discuss related work and in Section 4.5 we conclude this chapter.

4.1 STRUCTURE OF BELIEF CHANGE OPERATIONS

Classically the problem considered in belief revision is that of adapting a belief set, or an epistemic state or a belief base, by a propositional formula. In addition, it is commonly assumed that the information represented by the input formula can be given priority over existing information such that the latter has to be accommodated to be consistent with the former. Such an operator is called *prioritized revision operator*. These are a idealized situations in which a simplified problem is considered. As the sheer amount of articles on this topic shows, this simplified problem already exposes a variety of technical problems and variations to explore. We want to make use of the research on this problem and integrate it into change operators that can be used in agent systems. For the development of change operators for our epistemic agents we have to come up with solutions for our considered setting, that is:

(a) The epistemic state might be compound, not a single epistemic component

(b) The input of the change operator is a tuple that represents a speech act, not a propositional formula

(c) Other agents might not be credible, the input should not be generally accepted

(d) The epistemic components are not propositional and in particular based on non-monotonic logics

To address these challenges of our considered scenario we define a general construction of change operators for epistemic states in our framework. Hereby, we define sub-operations that correspond to operators which have been addressed theoretically in other works. This way we are able to benefit from existing results and to build on these.

To cover aspect (a) we use compound change operators that break the task of changing the entire epistemic state down to change operations on the individual epistemic components. For these we cover aspect (b) by introducing interpretation functions that interpret percepts and produce a set of logical formulae. Since the result of the interpretation functions might be a set of sentences the following operators have to be able to operate with these as input, which is called multiple revision [94, 95, 103] in the literature. To cover aspect (c) we need to use non-prioritized revision operators [126], i. e., operators that do not give unconditional priority to the new information. In particular we adapt the non-prioritized revision approach of selective revision [96] for our setting. In the selective revision approach a selection function on the revision input is used to decide in which form the input is to be accepted. While the theoretic work on this approach presents rationality postulates it does not give any constructional approach. We elaborate actual selection functions for belief bases by use of deductive argumentation in Section 4.2. To address aspect (d) we consider in particular the rationality postulates for propositional belief bases [127] and adapt them to the non-monotonic formalims. We do this in Section 4.3 where we develop general properties for non-monotonic formalisms based on postulates for propositional belief bases, adapt and develop specific postulates for ASP, and develop constructive approaches for which we show that they satisfy our proposed properties.

We build on the general concept of compound change operator for epistemic states as introduced in Section 3.3. There the general change operator is defined as

$$\circ =_{def} \circ_1 \cdot \ldots \cdot \circ_{n'}.$$

The actual composition of the operator has to match the composition of the epistemic state of the agent.

Example 4.1.1. Our considered composition of the BDI^+ agents lead to a change operator as defined in Section 3.5.3 for our BDI^+ agent type:

$$\circ_{\mathsf{BDI}^+} =_{def} \circ^{\dagger}_{\mathfrak{B}} \cdot \circ^{\dagger}_{\mathfrak{D}} \cdot \circ^{\dagger}_{\mathfrak{I}}.$$

And a change operator that modifies the epistemic state to reflect the anticipated changes of the execution of the action:

$$\circ^{a} : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}.$$

That is, a *BDI*$^+$*-epistemic-state* $\mathcal{K}_{\mathsf{BDI}^+} = \langle \mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathfrak{M}, \mathfrak{KH} \rangle$ is changed by a percept p such that $\mathcal{K}_{\mathsf{BDI}^+} \circ_{\mathsf{BDI}^+} p = \circ^{\dagger}_{\mathfrak{I}}(\circ^{\dagger}_{\mathfrak{D}}(\mathcal{K}_{\mathsf{BDI}^+} \circ^{\dagger}_{\mathfrak{B}} p))$. ♢

The further structure and properties of the particular change operators are left unspecified this far. In the following we consider the construction of a change operator for an epistemic component $E \subseteq \mathcal{P}(\mathcal{L}_E)$ and a set of speech acts Σ. A percept might represent an act of communication between agents, as introduced in Section 3.1, and comprise information about the sender of the information, the recipients, and the logical content. The interpretation function has to process this complex information into some sentence or a set of sentences in the target language. For the interpretation of the input and interpretation in the language of the respective belief component we formally introduce *interpretation functions* as

$$t : \Sigma \to \mathcal{P}(\mathcal{L}_E).$$

The set of sentences resulting from the interpretation of the input speech act is then evaluated in the light of the respective belief base and decided if and to which extent it should be accepted. This is formalized by the explicit introduction of a *selection function* for an epistemic component $E \subseteq \mathcal{L}_E$ that determines, on the basis of the epistemic component, in which form an input set of sentences shall be accepted. It is formalized as

$$f_E : \mathcal{P}(\mathcal{L}_E) \to \mathcal{P}(\mathcal{L}_E).$$

The result of the selection function represents the information which shall be accepted and thus be incorporated into the belief base with priority over the information in the belief base. This task of incorporating new information into a belief base with priority given to new information is exactly the task mainly studied in classic belief revision theory. Thus, we can make use of results from this field. However, the result of the selection function is a set of sentences such that we have to consider multiple revision operations or generalize approaches to the multiple revision case. The particular type of operation we are looking for is called *prioritized multiple revision* and we denote such an operator by

$$* : \mathcal{P}(\mathcal{L}_E) \times \mathcal{P}(\mathcal{L}_E) \to \mathcal{P}(\mathcal{L}_E).$$

That is, we slightly abuse notation by using the $*$ symbol for prioritized revision operators for single sentences and for sets of sentences as input. Hence in total, for an epistemic component $E \subseteq \mathcal{L}_E$ and a speech act $\iota \in \Sigma$ the change operation is given by

$$E \circ_E \iota = E * f_E(t(\iota))$$

The actual instantiations of the operators is highly dependent on the used formalism and the considered set of speech acts. The concrete set of speech acts is application-specific, such that interpretation functions should be designed with an application domain in mind. Hence, we postpone the further elaboration of the interpretation functions and treat the selection and the multiple prioritized base revision operators and their interaction in the following two sections. That is, in the following we consider epistemic components that are belief bases. We denote a propositional belief base by $\mathcal{B}$ and a belief base for answer set programming by P. As we note also in the following several formulations, constructions and results can be generalized to other forms of epistemic components.

4.2 SELECTIVE REVISION

Selection operators have been introduced and studied in [96]. In that work, a *selective revision operator* $\star_{f_\mathcal{B}}$ is defined by the composition

$$\mathcal{B} \star_{f_\mathcal{B}} \phi = \mathcal{B} * f_\mathcal{B}(\phi) \tag{4.2.1}$$

with a selection function f and a prioritized base revision operator $*$. We denote the selective revision operator $\star_{f_\mathcal{B}}$ as the *outer revision operator* and the prioritized base revision operator $*$ as the *inner revision operator*.

In [96], diverse properties for the selection functions are introduced and several results are proven that show how specific properties for the selection function and the *inner* prioritized revision translate to specific properties for the *outer* non-prioritized revision. However, no concrete implementations of the selection function (which is called a transformation function in [96]) are given in said work, nor in other works.

In this section we develop a specific implementation of a selection function that makes use of *deductive argumentation* [26], which we described in Section 2.3. A deductive argumentation theory is a set of propositional sentences and an argument for some sentence ϕ is a minimal proof for ϕ. If the theory is inconsistent there may also be proofs for the complement of a sentence $\neg\phi$. In order to decide whether ϕ or $\neg\phi$ is to be believed, an argumentative evaluation is performed that compares arguments with counterarguments. We use the framework of [26] to implement a selection function for selective revision that decides for each individual piece of information whether to accept it as input for the prioritized inner revision or not, based on its argumentative evaluation. In particular, we consider the case that revision is to be performed based on a set of pieces of information instead of just a single piece of information. By doing so, we allow new information to contain arguments. As a result, an agent decides whether to accept some new information on the basis of its own evaluation of the information and the arguments that may be contained in this information. Consider the following example.

Example 4.2.1. Imagine the agent *Anna* wants to spend her holidays on Hawaii. She is aware of the fact that there has been some volcano activity on Hawaii recently but is convinced there is no immediate danger. Anna's boss *Bob* doesn't want Anna to go on vacation at this time of the year and tells her that she has to do some work here and should not go to Hawaii. However, Anna wants to go surfing and to go to Hawaii instead of staying at work. As a consequence she rejects Bob's argument to stay and does not revise her beliefs. Consider now that *Carl*, a good friend of *Anna*, is a vulcanologist and tells Anna that there is actually an immediate danger of an eruption. Anna does not have sufficient arguments to defeat Carls information, thus accepts the new information and revises her beliefs accordingly. ◊

In the previous example the decisions of the agent Anna resulted in either accepting or rejecting the new information completely. However, it may also be the case that some of the new information is accepted and some is rejected. Consider the following example.

Example 4.2.2. Imagine Bob tells Anna that she has to stay for work because all her colleagues are having a vacation at the same time and she has to fill in for them. Suppose Anna knows that there is no work to do during her planned vacation as all clients of her company are on vacation as well. Then Anna would reject the conclusion of Bob's argument that she has to stay, but might very well accept that all her colleagues will be on vacation as well. ◊

The approach we develop here is capable of deciding whether to accept, reject, or partially accept some new information, based on deductive argumentation. In order to do so we extend the notions of selective revision to the problem of multiple base revision, i. e., the problem of revising a belief base (instead of a belief set) by a set of sentences in the following.

4.2.1 *Selective Multiple Base Revision*

We extend the approach of selective revision [96], which is formulated for belief sets and single formulae as input, to selective multiple (base) revision. That is, we consider the new information being represented by sets of formulae $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ instead of a single formula $\phi \in \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$. Even though we target belief base operations our formalization of the postulates and construction of the operation, as well as the results are equally applicable for the belief set case, unless stated differently explicitly. This brings us to the following definition, in accordance with [96], of a change operator $\star_{f_{\mathcal{B}}} : \mathcal{P}(\mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}) \times \mathcal{P}(\mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}) \to \mathcal{P}(\mathcal{L}^{\mathsf{prop}}_{\mathsf{At}})$ that is a selective multiple base revision via

$$\mathcal{B} \star_{f_{\mathcal{B}}} \Phi = \mathcal{B} * f_{\mathcal{B}}(\Phi) \tag{4.2.2}$$

with a selection function $f_{\mathcal{B}} : \mathcal{P}(\mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}) \to \mathcal{P}(\mathcal{L}^{\mathsf{prop}}_{\mathsf{At}})$ and some prioritized multiple base revision $* : \mathcal{P}(\mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}) \times \mathcal{P}(\mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}) \to \mathcal{P}(\mathcal{L}^{\mathsf{prop}}_{\mathsf{At}})$. In the following we define the general notions of a prioritized multiple base revision (inner revision operator), a non-prioritized multiple base revision (outer revision operator), and a selection function by means of postulates based on the considerations in [96]. Then we adapt a result of [96] showing that we obtain an outer revision operator, satisfying all desired properties by a selective revision construction with the defined inner revision and selection functions, which satisfy certain sets of properties. For this, we consider postulates from non-prioritized revision and the selective revision framework for the case of multiple base revision. At the same time we slightly adapt them to the case of multiple base revision.

The first property we consider is *Extensionality*, which we introduced for belief sets as the (K*6) postulate in Section 2.5.1 (Page 28). It expresses that equal inputs for the same belief base should lead to equal results, similarly as the *Uniformity* postulate in belief base revision theory, as introduced in Section 2.5.3 (Page 30). As discussed in Section 2.5, *Extensionality* is usually not considered for the problem of base revision as base revision is motivated by observing syntax and not (only) semantic contents. We include the postulate here to be compatible with the original formulation of selective revision for belief sets. Further we discuss and modify it to the case of multiple base revision. We phrase *Extensionality* for multiple base revision as follows:

EXTENSIONALITY$_\Phi$ If $\Phi \equiv^{p} \Psi$, then $\mathcal{B} * \Phi \equiv^{p} \mathcal{B} * \Psi$.

We denote postulates for multiple revision operations by a subscript Φ. We defined two notions of equivalence for sets of propositional formalae, $\equiv^{p}$ and $\cong^{p}$, in Section 2.2. It holds that $\Phi \equiv^{p} \Phi'$, if and only if it holds that $\Phi \vdash \Phi'$ and $\Phi' \vdash \Phi$. And it holds that $\Phi \cong^{p} \Phi'$ if and only if there is a bijection $\sigma : \Phi \to \Phi'$ such that for every

$\phi \in \Phi$ it holds that $\phi \equiv^p \sigma(\phi)$, i.e., if Φ and Φ' are *element-wise* equivalent.

For the case of multiple base revision, the satisfaction of *Extensionality* imposes that $\mathcal{B} * \{a, b\} \equiv^p \mathcal{B} * \{a \wedge b\}$ as $\{a, b\} \equiv^p \{a \wedge b\}$. It has been argued, that even for the belief set case $\{a, b\}$ expresses that a and b are two independent pieces of information while $\{a \wedge b\}$ expresses that they are not. Hence, both inputs might, and should, lead to different results, see e.g. [69] for a discussion. For our means and the inner revision operator we define the following weakened form of *Extensionality*$_\Phi$.

WEAK EXTENSIONALITY$_\Phi$ If $\Phi \cong^p \Phi'$ then $\mathcal{B} * \Phi \equiv^p \mathcal{B} * \Phi'$.

The property *Weak extensionality*$_\Phi$ only demands that the outcomes of the revisions $\mathcal{B} * \Phi$ and $\mathcal{B} * \Phi'$ are equivalent if Φ and Φ' are element-wise equivalent. This formulation does not have the same problem since $\{a, b\} \ncong^p \{a \wedge b\}$. All other postulates are direct generalizations of the base revision postulates introduced in Section 2.5.3 (Page 30) for a set of sentences as input:

SUCCESS$_\Phi$: $\Phi \subseteq \mathcal{B} * \Phi$

INCLUSION$_\Phi$: $\mathcal{B} * \Phi \subseteq \mathcal{B} \cup \Phi$

VACUITY$_\Phi$: If $\mathcal{B} \cup \Phi$ is consistent, then $\mathcal{B} \cup \Phi \subseteq \mathcal{B} * \Phi$

CONSISTENCY$_\Phi$: If Φ is consistent, then $\mathcal{B} * \Phi$ is consistent

RELEVANCE$_\Phi$: If $\Phi' \subseteq (\mathcal{B} \cup \Phi) \setminus (\mathcal{B} * \Phi)$, then there is a set H such that $\mathcal{B} * \Phi \subseteq H \subseteq \mathcal{B} \cup \Phi$ and H is consistent but $H \cup \Phi'$ is inconsistent

Given these postulates we can define the prioritized multiple base revision operator.

Definition 4.2.3. A revision operator $*$ is called a *prioritized multiple base revision operator* if $*$ satisfies *Success*$_\Phi$, *Inclusion*$_\Phi$, *Vacuity*$_\Phi$, *Consistency*$_\Phi$, *Relevance*$_\Phi$, and *Weak extensionality*$_\Phi$.

Having defined the inner revision operator we turn to the outer revision operator, the non-prioritized multiple base revision operator. For non-prioritized multiple base revision the properties *Inclusion*$_\Phi$, *Consistency*$_\Phi$, *Relevance*$_\Phi$, and *Weak extensionality*$_\Phi$ are desirable [126]. The *Vacuity* postulate is also not desirable for selective revision since even if the new information is consistent with the belief base there might be non-logical reasons to reject parts of it. This is not the case for the *Success*$_\Phi$ postulate since it demands to give absolute priority to the new information. *Success*$_\Phi$ and *Vacuity*$_\Phi$ can be replaced by weaker postulates, cf. [126]. Here, we consider the following two.

WEAK SUCCESS$_\Phi$ If $\mathcal{B} \cup \Phi \nvdash \perp$ then $\mathcal{B} \star_{f_\mathcal{B}} \Phi \vdash \Phi$.

CONSISTENT EXPANSION$_\Phi$ If $\mathcal{B} \not\subseteq \mathcal{B} \star_{f_\mathcal{B}} \Phi$ then $\mathcal{B} \cup (\mathcal{B} \star_{f_\mathcal{B}} \Phi) \vdash \perp$.

Note that *Weak success*$_\Phi$ follows from *Vacuity*$_\Phi$, and that *Consistent expansion*$_\Phi$ follows from *Vacuity*$_\Phi$ and *Success*$_\Phi$, as shown in [96].

Definition 4.2.4. A revision operator $\star_{f_\mathcal{B}}$ is called *non-prioritized multiple base revision operator* if $\star_{f_\mathcal{B}}$ satisfies *Inclusion*$_\Phi$, *Consistency*$_\Phi$, *Weak extensionality*$_\Phi$, *Weak success*$_\Phi$, and *Consistent expansion*$_\Phi$.

The last operator to be defined is the selection function. In [96] several properties for selection functions in the context of belief set revision are discussed. We rephrase some of them here slightly to fit the framework of multiple base revision. Let $\mathcal{B} \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathrm{At}}$ be consistent and let $\Phi, \Phi' \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathrm{At}}$.

INCLUSION$_f$ $f_\mathcal{B}(\Phi) \subseteq \Phi$

WEAK INCLUSION$_f$ If $\mathcal{B} \cup \Phi$ is consistent then $f_\mathcal{B}(\Phi) \subseteq \Phi$

EXTENSIONALITY$_f$ If $\Phi \equiv^p \Phi'$ then $f_\mathcal{B}(\Phi) \equiv^p f_\mathcal{B}(\Phi')$

CONSISTENCY PRESERVATION$_f$ If Φ is consistent then $f_\mathcal{B}(\Phi)$ is consistent

CONSISTENCY$_f$ $f_\mathcal{B}(\Phi)$ is consistent

MAXIMALITY$_f$ $f_\mathcal{B}(\Phi) = \Phi$

WEAK MAXIMALITY$_f$ If $\mathcal{B} \cup \Phi$ is consistent then $f_\mathcal{B}(\Phi) = \Phi$

In addition to the adapted postulates from [96] we also consider a weakened version of *Extensionality*$_f$.

WEAK EXTENSIONALITY$_f$ If $\Phi \cong^p \Phi'$ then $f_\mathcal{B}(\Phi) \cong^p f_\mathcal{B}(\Phi')$

The above properties are considered as possible properties for a selective revision operation and are not assumed to be satisfied by every such operator. For example, the property *maximality*$_f$ states that $f_\mathcal{B}$ should not modify the set Φ. Satisfaction of this property leads to the equivalence of the selective revision operator, as defined in (4.2.2), and the used prioritized revision operator, such that

$$\mathcal{B} \star_{f_\mathcal{B}} \Phi = \mathcal{B} * f_\mathcal{B}(\Phi) = \mathcal{B} * \Phi.$$

As $*$ is meant to be a prioritized revision function we lose the possibility for non-prioritized revision.

Here, we consider the postulates *Inclusion*$_f$, *Weak extensionality*$_f$, *consistency preservation*$_f$, and *Weak maximality*$_f$ as desirable for a selection function. As for the multiple base revision operator, the satisfaction of *Extensionality*$_f$ is not desirable for a selection function. The satisfaction of it would force the results for the inputs $\Phi = \{a, b\}$ and $\Phi' = \{a \wedge b\}$ always to be equal. This is not generally desirable since

$\{a, b\}$ expresses that a and b are two independent pieces of information while $\{a \wedge b\}$ expresses that they are not. Moreover, for the case of selection functions another problem with the satisfaction of *Extensionality*$_f$ arises. Consider again $\Phi = \{a, b\}$ and $\Phi' = \{a \wedge b\}$. It follows that $\Phi \equiv^p \Phi'$ and if $f_{\mathcal{B}}$ satisfies *Extensionality*$_f$ this results in $f_{\mathcal{B}}(\{a, b\}) \equiv^p f_{\mathcal{B}}(\{a \wedge b\})$. If $f_{\mathcal{B}}$ also satisfies *Inclusion*$_f$ it follows that $f_{\mathcal{B}}(\{a \wedge b\}) \in \{\emptyset, \{a \wedge b\}\}$, i. e., that the entire input is accepted or none of it. This might be adequate in this case. However, for $\{a, b\}$ it follows that $f_{\mathcal{B}}(\{a, b\}) \in \{\emptyset, \{a, b\}\}$ and surely the options to keep at least one of the input elements should be also considerable here. Hence, for *Weak extensionality*$_f$ we demand $f_{\mathcal{B}}(\Phi)$ and $f_{\mathcal{B}}(\Phi')$ to be element-wise equivalent (in contrast to the property *Weak extensionality*$_\Phi$ for revision).

In general, if $f_{\mathcal{B}}$ satisfies both *Inclusion*$_f$ and *Extensionality*$_f$ it follows that either $f_{\mathcal{B}}(\Phi) = \emptyset$ or $f_{\mathcal{B}}(\Phi) = \Phi$ for every $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ (as Φ is equivalent to a Φ' that consists of a single formula that is the conjunction of the formulae in Φ such that, by *Inclusion*$_f$, only the options $f_{\mathcal{B}}(\Phi') = \emptyset$ or $f_{\mathcal{B}}(\Phi') = \Phi'$ are possible). As we are interested in a more graded approach to belief revision we want to be able to accept or reject specific parts of Φ and not just Φ completely. Consequently, we consider *Weak extensionality*$_f$ as a desirable property instead of *Extensionality*$_f$. Note that *Extensionality*$_f$ implies *Weak extensionality*$_f$ as $\Phi \cong^p \Phi'$ implies $\Phi \equiv^p \Phi'$.

In [96] several representation theorems are given that characterize non-prioritized belief revision by selective revision via (4.2.2) and specific properties of $*$ and $f_{\mathcal{B}}$. In particular, it is shown that a reasonable non-prioritized belief revision operator $\star_{f_{\mathcal{B}}}$ can be characterized by an AGM revision $*$ and a selection function $f_{\mathcal{B}}$ that satisfies *Extensionality*$_f$, *Consistency preservation*$_f$, and *Weak maximality*. We can carry over the results of [96] to the problem of multiple base revision and obtain the following result.

Proposition 4.2.5. Let $*$ be a prioritized multiple base revision operator and let $f_{\mathcal{B}}$ satisfy *Inclusion*$_f$, *Weak extensionality*$_f$, *Consistency preservation*$_f$, and *Weak maximality*$_f$. Then $\star_{f_{\mathcal{B}}}$ defined via (4.2.2) is a non-prioritized multiple base revision operator.

Proof. See Appendix A.1.2 on Page 226. □

The *Relevance*$_\Phi$ postulate does not hold for $\mathcal{B} \star_{f_{\mathcal{B}}} \Phi$ defined via (4.2.2) in general. It is arguable if relevance should hold, it would constrain the selection function by demanding, roughly, that it can only refuse pieces of information that would lead to inconsistency. However, for a selection function it makes perfect sense to reject the entire input if only a part of it is inconsistent with the belief base. Consider for example a set of convictions $\mathcal{B}_R \subseteq \mathcal{B}$ of a belief base and the selection function $f^0_{\mathcal{B}}$ defined via $f^0_{\mathcal{B}}(\Phi) = \Phi$ if $\mathcal{B}_R \cup \Phi$ is consistent and $f^0_{\mathcal{B}}(\Phi) = \emptyset$ otherwise. The selection function $f^0_{\mathcal{B}}$ satisfies all properties for selection functions except *Maximality*$_f$. But it is easy to see that $\mathcal{B} \star_{f_{\mathcal{B}}} \Phi$ defined via (4.2.2) using $f^0_{\mathcal{B}}$ and a prioritized multiple base revision operator $*$ fails to satisfy *Relevance*$_\Phi$. This selection function might be desirable in settings in which the convictions are strongly believed and any information source that contradicts these convictions is regarded as not credible. While such an operation is arguably desirable in all cases, it is clearly desirable that the selective revision framework should be able to capture such an operator.

Up to here we extended the selective revision framework and characterized the operators we consider desirable. In the following, we aim at implementing a selective multiple base revision using deductive argumentation.

4.2.2 *Selective Revision by Deductive Argumentation*

The deductive argumentation framework presented in Section 2.3 (Page 22) allows to decide for each sentence $\alpha \in \Phi$ whether α is justifiable with respect to Φ. Here, we use this to define selection functions. The framework of deductive argumentation heavily depends on the actual instances of categorizers and accumulators, which are left uninstantiated in Section 2.3. Here, we demand minimal requirements of both functions to be used in a selection function.

Definition 4.2.6 (Well-behaving categorizer). Let τ and τ' be argument trees. A categorizer γ is called *well-behaving* if $\gamma(\tau) > \gamma(\tau')$ whenever τ consists only of one single node and τ' consists of at least two nodes.

In other words, a categorizer γ is well-behaving if the argument tree that has no undercuts for its root is considered the best justification for the root.

Definition 4.2.7 (Well-behaving accumulator). Let τ and τ' be argument trees. An accumulator κ is called *well-behaving* if and only if $\kappa((\mathcal{T}^+, \mathcal{T}^-)) > 0$ whenever $\mathcal{T}^+ \neq \emptyset$ and $\mathcal{T}^- = \emptyset$.

This means, that if there are no arguments against a claim α and at least one argument for α in Φ then α should be accepted in Φ. Both, γ_0 and κ_0 as defined in Section 2.3 are well-behaving as well as all categorizers and accumulators considered in [26]. If Φ is consistent then every sentence $\alpha \in \Phi$ is accepted by Φ with respect to every well-behaving categorizer and well-behaving accumulator.

Let $\mathcal{B} \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ be a consistent set of sentences, and let γ be some well-behaving categorizer and let κ be some well-behaving accumulator. We consider a selective revision $\star_{f_{\mathcal{B}}}$ of the form (4.2.2). In order to determine the outcome of the non-prioritized revision $\mathcal{B} \star_{f_{\mathcal{B}}} \Phi$ for some $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$ we implement a selection function f that checks for every sentence $\alpha \in \Phi$ it α is accepted in $\mathcal{B} \cup \Phi$. Although $\mathcal{B}$ is consistent, the union $\mathcal{B} \cup \Phi$ is not necessarily consistent which gives rise to an argumentative evaluation. We consider the following *two* different selection functions based on deductive argumentation.

Definition 4.2.8 (Skeptical Selection Function). We define the *skeptical selection function* $S^{\gamma,\kappa}_{\mathcal{B}}$ via

$$S^{\gamma,\kappa}_{\mathcal{B}}(\Phi) = \{\alpha \in \Phi \mid \mathcal{B} \cup \Phi \mid\!\sim_{\kappa,\gamma} \alpha\}$$

for every $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$.

Definition 4.2.9 (Credulous Selection Function). We define the *credulous selection function* $C^{\gamma,\kappa}_{\mathcal{B}}$ via

$$C^{\gamma,\kappa}_{\mathcal{B}}(\Phi) = \{\alpha \in \Phi \mid \mathcal{B} \cup \Phi \not\mid\!\sim_{\kappa,\gamma} \neg\alpha\}$$

for every $\Phi \subseteq \mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$.

In other words, the value of $S^{\gamma,\kappa}_{\mathcal{B}}(\Phi)$ consists of those sentences of Φ that are accepted in $\mathcal{B} \cup \Phi$ and the value of $C^{\gamma,\kappa}_{\mathcal{B}}(\Phi)$ consists of those sentences of Φ that are not rejected in $\mathcal{B} \cup \Phi$. There is a subtle, but important, difference in the behavior of those two selection functions as the following example shows.

Example 4.2.10. Let $\mathcal{B}_1 = \{a\}$ and $\Phi_1 = \{\neg a\}$. There is exactly one argument tree τ_1 for $\neg a$ and one argument tree τ_2 for a in $\mathcal{B}_1 \cup \Phi$. In τ_1 the root is the argument $A = \langle\{\neg a\}, \neg a\rangle$ which has the single canonical undercut $B = \langle\{a\}, a\rangle$. In τ_2 the situation is reversed and the root of τ_2 is the argument B which has the single canonical undercut A. Therefore, the argument structure for $\neg a$ is given via $\Gamma_{\mathcal{B}\cup\Phi}(\neg a) = (\{\tau_1\}, \{\tau_2\})$. We use the categorizer γ_0 and accumulator κ_0 we defined in Example 2.3.11 (Page 2.3.11). It follows that $\gamma_0(\tau_1) = \gamma_0(\tau_2) = 0$ and

$$\kappa_0(\gamma_0(\Gamma_{\mathcal{B}\cup\Phi}(a))) = \kappa_0(\langle 0, 0\rangle) = 0.$$

It follows that $\mathcal{B} \cup \Phi$ is undecided about both $\neg a$ and a. Consequently, it follows that

$$S^{\gamma_0,\kappa_0}_{\mathcal{B}_1}(\Phi_1) = \emptyset \qquad\qquad C^{\gamma_0,\kappa_0}_{\mathcal{B}_1}(\Phi_1) = \{\neg a\}.$$

That is, if the evaluation of a formula is undecided, the skeptical selection function rejects the formula and the credulous selection function accepts it. ♢

Let $*$ be some (prioritized) multiple base revision operator, γ some categorizer, and κ some accumulator. Using the skeptical selection function we can define the *skeptical argumentative revision* $\circ_S^{\gamma,\kappa}$ following (4.2.2) via

$$\mathcal{B} \circ_S^{\gamma,\kappa} \Phi = \mathcal{B} * S_{\mathcal{B}}^{\gamma,\kappa}(\Phi) \tag{4.2.3}$$

for every $\Phi \subseteq \mathcal{L}_{\mathsf{At}}^{\mathsf{prop}}$ and using the credulous selection function we can define the *credulous argumentative revision* $\circ_C^{\gamma,\kappa}$ via

$$\mathcal{B} \circ_C^{\gamma,\kappa} \Phi = \mathcal{B} * C_{\mathcal{B}}^{\gamma,\kappa}(\Phi) \tag{4.2.4}$$

for every $\Phi \subseteq \mathcal{L}_{\mathsf{At}}^{\mathsf{prop}}$.

We exemplify these two types of argumentative revision operators by the following simple example.

Example 4.2.11. We continue Example 4.2.10. Let $*$ be some prioritized multiple base revision operator. Then it follows that $\mathcal{B}_1 \circ_S^{\gamma_0,\kappa_0} \Phi_1 = \{a\}$ and $\mathcal{B}_1 \circ_C^{\gamma_0,\kappa_0} \Phi_1 = \{\neg a\}$. ♢

Next we give a more complex example of a selective revision operation.

Example 4.2.12. We continue and formalize Examples 4.2.1 and 4.2.2. We consider the atoms $\mathsf{At} = \{s, h, l, m, f, v\}$ with the following informal interpretations.

s :	Anna is a surf fanatic
h :	Anna travels to Hawaii
f :	Anna has financial problems
l :	Anna takes a loan
m :	Anna has a lot of money
v :	There is volcano activity on Hawaii

Now consider Anna's belief base $\mathcal{B}_1$ given via

$$\mathcal{B}_1 = \{s,\ \ s \Rightarrow h,\ \ l,\ \ l \Rightarrow m,\ \ m \Rightarrow h,\ \ m \Rightarrow \neg f\}.$$

Anna is a surf fanatic (s) and believes that a surf fanatic should travel to Hawaii $(s \Rightarrow h)$. Anna has taken a loan (l), and taking a loan means having money available $(l \Rightarrow m)$. Having money implies she should travel to Hawaii $(m \Rightarrow h)$, and having money also implies she does not have financial problems $(m \Rightarrow \neg f)$. Note that $\mathcal{B}_1 \vdash h$, i. e., from $\mathcal{B}_1$ Anna concludes she should go to Hawaii.

Consider the new information $\Phi_1 = \{f,\ f \Rightarrow \neg h,\ v,\ v \Rightarrow \neg h\}$ stemming from communication with Anna's mother. With Φ_1 the mother of Anna wants to convince Anna not to travel to Hawaii. In particular, Φ_1 states that Anna has financial problems (f), that having financial problems Anna should not travel to Hawaii ($f \Rightarrow \neg h$), that there is also volcano activity on Hawaii (v), and that given volcano activity Anna should not travel to Hawaii ($v \Rightarrow \neg h$).

As one can see there are several arguments for and against h in $\mathcal{B}_1 \cup \Phi_1$, e. g., $\langle s, s \Rightarrow h, h\rangle$, $\langle f, f \Rightarrow \neg h, \neg h\rangle$. If follows that $\mathcal{B}_1 \cup \Phi_1$

accepts $f \Rightarrow \neg h$, but rejects f, v, and $v \Rightarrow \neg h$ with respect to γ_0 and κ_0. Furthermore, $\mathcal{B}_1 \cup \Phi_1$ accepts $\neg f$ and rejects $\neg v$ and $\neg(v \Rightarrow \neg h)$ with respect to γ_0 and κ_0 which means that both v and $v \Rightarrow \neg h$ are credulously accepted. Consequently, the values of $S^{\gamma_0,\kappa_0}_{\mathcal{B}_1}(\Phi_1)$ and $C^{\gamma_0,\kappa_0}_{\mathcal{B}_1}(\Phi_1)$ are given via

$$S^{\gamma_0,\kappa_0}_{\mathcal{B}_1}(\Phi_1) = \Phi_1 \setminus \{f, v, v \Rightarrow \neg h\} \quad \text{and} \quad C^{\gamma_0,\kappa_0}_{\mathcal{B}_1}(\Phi_1) = \Phi_1 \setminus \{f\}.$$

Let $*$ be some prioritized multiple base revision operator and define $\circ^{\gamma_0,\kappa_0}_S$ and $\circ^{\gamma_0,\kappa_0}_C$ via (4.2.3) and (4.2.4), respectively. Then some possible revisions of $\mathcal{B}_1$ with Φ_1 are given via

$$\mathcal{B}_1 \circ^{\gamma_0,\kappa_0}_S \Phi_1 = \{s,\ s \Rightarrow h,\ l,\ l \Rightarrow m,\ m \Rightarrow h,\ m \Rightarrow \neg f,\ f \Rightarrow \neg h\}$$
and
$$\mathcal{B}_1 \circ^{\gamma_0,\kappa_0}_C \Phi_1 = \{s,\ l \Rightarrow m,\ m \Rightarrow h,\ m \Rightarrow \neg f,\ f \Rightarrow \neg h,\ v \Rightarrow \neg h,\ v\}.$$

Note that it holds that $\mathcal{B}_1 \circ^{\gamma_0,\kappa_0}_S \Phi_1 \vdash h$ and $\mathcal{B}_1 \circ^{\gamma_0,\kappa_0}_C \Phi_1 \vdash \neg h$. ◇

For the evaluation of our approach the sophisticated algorithms presented in [28] can be used. However, the underlying problems of deciding whether a set of propositional formulae is consistent is NP-complete and deciding whether it entails a given formula is co-NP-complete [107] such that no generally effective implementation can be expected.

For the selection functions $S^{\gamma,\kappa}_{\mathcal{B}}$ and $C^{\gamma,\kappa}_{\mathcal{B}}$ and the resulting revision operators $\circ^{\gamma,\kappa}_S$ and $\circ^{\gamma,\kappa}_C$ we can show the following results.

Proposition 4.2.13. Let γ be a well-behaving categorizer and κ be a well-behaving accumulator. Then the selection functions $S^{\gamma,\kappa}_{\mathcal{B}}$ and $C^{\gamma,\kappa}_{\mathcal{B}}$ satisfy *Inclusion*$_f$, *Weak inclusion*$_f$, *Weak extensionality*$_f$, *Consistency preservation*$_f$ and *Weak maximality*$_f$.

Proof. See Appendix A.1.2 on Page 227. □

In particular, note that both $S^{\gamma,\kappa}_{\mathcal{B}}$ and $C^{\gamma,\kappa}_{\mathcal{B}}$ do not satisfy either *Consistency*$_f$ or *Maximality*$_f$ in general. On the basis of the Propositions 4.2.5 and 4.2.13 the following corollary can be shown.

Corollary 4.2.14. Let γ be a well-behaving categorizer and κ be a well-behaving accumulator. Then both $\circ^{\gamma,\kappa}_S$ and $\circ^{\gamma,\kappa}_C$ are *non-prioritized multiple base revision operators*.

Proof. See Appendix A.1.2 on Page 228. □

Hence, we have shown that our construction of a selection function on the basis of deductive argumentation leads to non-prioritized multiple base revision operator. In the next section we consider multiple base revision operators for answer set programming.

4.3 MULTIPLE BASE REVISION FOR ASP

We described the state of the art of change operations for ASP in Section 2.5.4. While several approaches to find change operators that satisfy the AGM Postulates for belief sets or epistemic states have been made, there is no work on the consideration of the belief base approach for ASP apart the first publication our approach, which we present in the following, in [154].

We focus on the revision operation and base it on a consolidation operation, which can be used to specify other types of change operations as well. The well developed classical base revision approach has not been considered in the light of ASP before. In fact, we argue that the belief base approach is the intuitive one for ASP. AGM change operations on belief sets can be seen as operations on the knowledge level, abstractly describing how an ideal reasoner would change its beliefs. This underlies the assumption of a perfect reasoner while ASP's main features are effective computation of finite programs with finite answer sets. The deductive closure, a crucial property of belief sets, is defined neither for programs nor for answer sets. Belief bases are also more expressive; since on the knowledge level one cannot distinguish between inferred beliefs and fundamental, or self-supporting, ones. While this abstraction from the fundamental beliefs and their syntactic representation has advantages for the global picture of belief change we argue that ASP is primarily a syntax based approach. A key feature of ASP is that beliefs are formulated in form of easily understandable rules that allow for explicit exceptions and the explanation of inferences. From the base revision perspective the result of a change operation for ASP should be founded, understandable and close to the original syntax. As we described in Section 2.5.4 the SE-model based approach to the revision of answer set programs inspired by the AGM belief set approach [72] leads to unintuitive results from the ASP perspective. The example for this taken from [216] is as follows. Consider the programs $P_1 = \{p., \ q.\}$ and $P_2 = \{p \leftarrow q., \ q.\}$, they have the same SE-models and therefore also the results of the revision by the program $Q = \{\neg q.\}$ are the same. It holds that the answer sets for $AS(P_1 *^{SE} Q) = AS(P_2 *^{SE} Q) = \{\{p\}\}$. While for P_1 this is a desired result, for P_2 it is not since p is not justified if q is not in the answer set.

Here, we present a general exploration of the application of classic base revision theory to change operations on ASP. We discuss ASP specific postulates from the literature in the light of a base revision approach, proofs of the relationships among both and formulation of adapted postulates. Finally, we develop a new base revision construction via a screened consolidation operator which is applicable to ASP. We make use of global selection functions that lead to the definition of general operator that can be used iteratively. We prove a characterization theorem for our construction.

The remainder of this section is structured as follows. In the next section we develop base revision postulates for ASP, discuss them and relate them to specific postulates for change operations ASP from the literature. After that we present our construction of multiple base revision operators and show its applicability to ASP and the correspondence to the postulates.

4.3.1 *Postulates for ASP Base Revision*

In contrast to postulates for the revision of belief sets the postulates for belief base revision are formulated without the assumption of deductively closed sets, or belief operators that result in such as for the postulates for epistemic states, cf. Section 2.5. Hence they are applicable to other formalisms as propositional logic, in this case, extended logic programs. However, the crucial notion to be defined for this is the one of consistency of a set of formulae. Further, the appropriateness of the postulates for the considered formalism has to be evaluated and the postulates have to be adapted and extended accordingly.

Revision aims at solving conflicts between prior beliefs and the input on the basis of inconsistency. In the context of logic programs the decisive property of inconsistency is that the state of inconsistency of a program can change non-monotonically. While in the propositional case any superset of an inconsistent set of formulae is inconsistent, for an inconsistent logic program P there can be a superset Q, $P \subset Q$, such that Q is consistent. We illustrate this in the following example.

Example 4.3.1. The program $P = \{a., \neg a \leftarrow \mathsf{not}\ b.\}$ is obviously inconsistent. $P' = P \cup \{b \leftarrow \mathsf{not}\ c.\}$ is consistent with $AS(P') = \{\{a, b\}\}$ while $P'' = P' \cup \{c.\}$ is inconsistent again. ♢

We shall see the implications of this in the following.

We define base revision postulates for a *multiple base revision operator* $* : \mathcal{P}(\mathcal{L}^{\mathsf{asp}}_{At}) \times \mathcal{P}(\mathcal{L}^{\mathsf{asp}}_{At}) \to \mathcal{P}(\mathcal{L}^{\mathsf{asp}}_{At})$ for ASP. The base revision postulates given in Section 2.5.3 (Page 30) can be directly translated to the logic programming case with $+$ being the *non-closing expansion* $P + Q = P \cup Q$. For $*$ being a multiple base revision operator for logic programs we define the following postulates:

SUCCESS$_Q$: $Q \subseteq P * Q$

INCLUSION$_Q$: $P * Q \subseteq P + Q$

VACUITY$_Q$: If $P + Q$ is consistent, then $P + Q \subseteq P * Q$

CONSISTENCY$_Q$: If Q is consistent, then $P * Q$ is consistent.

RELEVANCE$_Q$: If $r \in (P \cup Q) \setminus (P * Q)$, then there is a program H such that $P * Q \subseteq H \subseteq P \cup Q$ and H is consistent but $H \cup \{r\}$ is inconsistent.

FULLNESS$_Q$: If $r \in (P \cup Q) \setminus (P * Q)$, then $P * Q$ is consistent and $(P * Q) \cup \{r\}$ is inconsistent.

UNIFORMITY$_Q$: If for all $P' \subseteq P$, $P' \cup \{Q\}$ is inconsistent if and only if $P' \cup \{R\}$ is inconsistent, then $P \cap (P * Q) = P \cap (P * R)$

The postulate of *Fullness*$_Q$ is a stronger version of *Relevance*$_Q$ and resembles a stronger requirement to the minimality of change.

Lemma 4.3.2. If $*$ satisfies *Fullness*$_Q$, then it satisfies *Relevance*$_Q$.

Proof. See Appendix A.1.2 on Page 228. □

For the classical case *Fullness*$_Q$ is considered as too strong [127]. However for weaker logics it has also been proven to be useful, for instance for Horn logic in [73]. For logic programing it also does not have the same undesirable implications as we shall see later on.

Due to the non-monotonicity of inconsistency we have to adapt the postulates that base on a notion of consistency such that the adapted ones capture the same idea for ASP as the original ones have for the propositional case.

The first postulate based on the notion of consistency is *Vacuity*$_Q$. The idea it implements, that if the expansion is consistent then no information shall be discarded, is equally captured for non-monotonic consistency.

The *Consistency*$_Q$ postulate expresses in the classical case that the outcome of the revision shall be consistent whenever possible. For classical propositional logic this is possible if and only if the input is consistent. Due to the *Success*$_Q$ postulate the inconsistent set of formulae has to be part of the revised belief base. For a propositional belief base it holds that it is inconsistent if there is a subset of it that is inconsistent. Hence any revision with an inconsistent input is inconsistent in the case of propositional logic. In logic programming, however, the input can be inconsistent, the *Success*$_Q$ postulate satisfied and yet the revision outcome can be consistent. This is shown by the following example.

Example 4.3.3. Let $P = \{b., \neg a.\}$ and $Q = \{a., \neg a \leftarrow \text{not } b.\}$. The program Q is inconsistent but the revision $P * Q = \{b., a., \neg a \leftarrow \text{not } b.\}$ is consistent and satisfies all postulates. ◊

Hence we strengthen the *Consistency*$_Q$ postulate to adequately capture non-monotonic inconsistency by use of an appropriate premise for the possibility of consistency.

NM-CONSISTENCY$_Q$: If there exists some consistent X, $Q \subseteq X \subseteq P \cup Q$, then $P * Q$ is consistent.

The postulate *NM-Consistency*$_Q$ is stronger than the *Consistency*$_Q$ postulate as we show in the following proposition.

Proposition 4.3.4. If a revision operator $*$ satisfies *NM-Consistency*$_Q$, then it satisfies *Consistency*$_Q$.

Proof. See Appendix A.1.2 on Page 88. □

The idea behind the postulates of *Relevance*$_Q$ and *Fullness*$_Q$ is to implement a notion of minimal change to obtain consistency. Due to the non-monotonicity of consistency, i.e., a superset of an inconsistent program can be consistent, the straightforward adoption from the classical case is not sufficient. In particular, the notion of consistency we consider here does not satisfy the following property: If P is consistent and for all $r \in R$ it holds that $P \cup \{r\}$ is inconsistent, then $P \cup R$ is inconsistent. Here is a counterexample:

Example 4.3.5. Let $P = \{a \leftarrow b, \text{not } a., \ a \leftarrow c, \text{not } a., \ a \leftarrow b, c.\}$ and $R = \{b., \ c.\}$. Then, $P \cup \{b.\}$ is inconsistent and $P \cup \{c.\}$ is inconsistent but $P \cup R$ is consistent. It follows that in this case, a revision operator $*$ such that $R * P = P$ satisfies *Fullness*$_Q$. Obviously, such an operator does not comply with the intention of the *Fullness*$_Q$ postulate. ♢

This example shows that for the ASP case the consideration of single rules inappropriate. The complete program has to be considered. The idea of minimal change expressed in the *Fullness*$_Q$ postulate is that none of the discarded information can be added without loosing consistency. For non-monotonic notions of consistency we formulate the following appropriate adaption of *Fullness*$_Q$.

NM-FULLNESS$_Q$: If $R = (P \cup Q) \setminus P * Q \neq \emptyset$, then $P * Q$ is consistent and for all $R' \subseteq R$ it holds that $(P * Q) \cup R'$ is inconsistent.

The postulate *NM-Fullness*$_Q$ is stronger than the *Fullness*$_Q$ postulate as we show in the following proposition.

Proposition 4.3.6. If a revision operator $*$ satisfies *NM-Fullness*$_Q$, then it satisfies *Fullness*$_Q$ and *Vacuity*$_Q$.

Proof. See Appendix A.1.2 on Page 228. □

The basic set of postulates for belief base revision as we introduced them in Section 2.5 (Page 27) have been accepted as characterizations of desirable revision operations for classical belief bases. The postulates can be applied to belief bases represented as logic programs as just shown. Next we consider further postulates for the connection of the revision on the program level and the resulting answer sets. Such

ASP specific postulates for change operators have been proposed in [87] and adopted by several authors for the evaluation of their approaches afterwards [75, 68, 74, 188]. For these postulates difference notions of equivalence of logic programs have been considered, see Section 2.4 (Page 25) for the definition of the different notions of equivalence. Those postulates base on the notion of ordinary equivalence based on the identity of the sets of answer sets (AS) in [87][1], on strong equivalence (SE) in [68] and partially on uniform equivalence (UE) [189] in [74]. Here, we generalize the postulates by formulating them with the notion of equivalence as a parameter, denoted by $\bullet$. We obtain the original postulates with $\bullet = AS$ but consider a family of equivalences $\bullet \in \{AS, UE, SE, P\}$; with $\equiv_P$ being the *syntactic equivalence of programs*, i.e., $P \equiv_{SE} P'$ if and only if $P = P'$. The equivalences are increasingly stronger with the order given above. More precisely, it holds that for any two programs P and P' that:

- if $P \equiv_P P'$ then $P \equiv_{SE} P'$,
- if $P \equiv_{SE} P'$ then $P \equiv_{UE} P'$ and
- if $P \equiv_{UE} P'$ then $P \equiv_{AS} P'$.

Thus apart from the original postulates we also consider stronger versions of these. This family of notions of equivalence can be extended by all intermediate notions as formalized in [246].

For one of the postulates, namely the *Tautology* postulate, we need to define the notion of tautological programs. For propositional logic tautologies are defined as being true in all interpretations. Formally a sentence $\phi_\top$ is a tautology if and only if $\phi_\top \in Cn^{\mathsf{prop}}(\emptyset)$, due to $Mod(\emptyset) \subseteq Mod(\phi_\top)$. We define a tautological program as follows.

Definition 4.3.7. Let P be a program over the set of literals Lit. The program P is *tautological*, denoted by $P_\top$, if and only if for each state $I \subseteq \mathsf{Lit}$, for all rules $r \in P_\top$ it holds that if $\mathsf{body}^+(r) \subseteq I$ and $\mathsf{body}^-(r) \cap I = \emptyset$ then $\mathsf{head}(r) \in I$.

Example 4.3.8. The program $P_\top = \{a \leftarrow a.\}$ is tautological. ◇

Given this definition of we can formulate the generalized postulates for ASP revision from [87]:

INITIALISATION$_\bullet$: $\emptyset * P \equiv_\bullet P$

IDEMPOTENCE$_\bullet$: $P * P \equiv_\bullet P$

ABSORPTION$_\bullet$: $(P * Q) * Q \equiv_\bullet P * Q$

TAUTOLOGY$_\bullet$: $P * P_\top \equiv_\bullet P$

1 For finite alphabets.

DISJOINTNESS$_\bullet$: If $P = P_1 \cup P_2$ and P_1 and P_2 have disjoint sets of literals, then $P * Q \equiv_\bullet (P_1 * Q) \cup (P_2 * Q)$.

PARALLELISM$_\bullet$: If Q_1 and Q_2 have disjoint sets of literals, then $P * (Q_1 \cup Q_2) \equiv_\bullet (P * Q_1) \cup (P * Q_2)$.

The implications on the equivalences given above lead to implications of the resulting postulates. Most importantly we get that if an operator $*$ satisfies a postulate for some notion of equivalence, then it also satisfies the variants of the postulates for all weaker notions of equivalence, e. g., *Initialisation*$_P$ implies *Initialisation*$_{SE}$, *Initialisation*$_{UE}$ and *Initialisation*$_{AS}$. On the other hand, if $*$ is shown to violate *Initialisation*$_{AS}$, then it also violates *Initialisation*$_{UE}$, *Initialisation*$_{SE}$ and *Initialisation*$_P$. The same holds for all other postulates. In the following we only show results for the strongest version of a postulate and omit the implications of them. We obtain the following results for the connection of base revision postulates and ASP change postulates:

Proposition 4.3.9. Let $*$ be a revision operator on logic programs.

1. If $*$ satisfies Success$_Q$ and *Inclusion*$_Q$, then it satisfies Initialisation$_P$.
2. If $*$ satisfies Success$_Q$ and *Inclusion*$_Q$, then it satisfies Idempotence$_P$.
3. If $*$ satisfies Success$_Q$, NM-Consistency$_Q$, *Inclusion*$_Q$ and *NM-Fullness*$_Q$, then it satisfies Absorption$_P$.

Proof. See Appendix A.1.2 on Page 229. □

Hence, we have just shown that the first three ASP postulates follow for all considered notions of equivalence from very basic base revision postulates. This supports the adequateness of the base revision approach for ASP. The remaining three ASP postulates do not follow from the base revision postulates and are in conflict with some of them. This might be due to the fact that they were formulated for approaches for handling update sequences of logic programs which were shown to "neither have update nor revision flavor" in the classical sense [87]. However, the underlying ideas of these postulates are useful and can be adapted to the base revision setting, as we show in the following.

The *Tautology*$_{AS}$ postulate is violated if the belief base is inconsistent before and consistent after revising by a tautology, as we illustrate in the following example.

Example 4.3.10. Let $P = \{a., \neg a.\}$ and $Q = \{b \leftarrow b.\}$ with $AS(P) = \emptyset$ and $AS(Q) = \{\emptyset\}$. For any revision operator $*$ satisfying Tautology$_{AS}$ we get $AS(P * Q) = \emptyset$, e. g., $P * Q = \{a., \neg a., b \leftarrow b.\}$. Consistency$_Q$ on the other hand demands changes to make the belief base consistent.

For any revision operator $*'$ satisfying *Consistency*$_Q$ we get $AS(P *' Q) \neq \emptyset$. In this case $P *' Q = \{a., b \leftarrow b.\}$ or $P *' Q = \{\neg a., b \leftarrow b.\}$. $\diamondsuit$

Thus *Tautology*$_{AS}$ cannot be satisfied by any operator satisfying *Consistency*$_Q$ because of the case in which an inconsistent belief base is made consistent by the revision by a tautology. On the program level, i. e., with respect to $\equiv_P$, the *Tautology*$_P$ postulate is not compatible with the *Success*$_Q$ or the *Vacuity*$_Q$ postulate.

Proposition 4.3.11. Let $*$ be a revision operator on logic programs.

1. If $*$ satisfies *Consistency*$_Q$, then it violates *Tautology*$_{AS}$.
2. If $*$ satisfies *Success*$_Q$, then it violates *Tautology*$_P$.
3. If $*$ satisfies *Vacuity*$_Q$, then it violates *Tautology*$_P$.

Proof. See Appendix A.1.2 on Page 229. $\square$

The *Tautology*$_{AS}$ in general addresses an important issue in ASP revision, namely that if the belief base is consistent, the revision by a tautology should not lead to any changes of the semantics. This is not satisfied by many approaches to dynamics in ASP. We adapt the *Tautology*$_{AS}$ postulate to be compatible with the *Consistency*$_Q$ postulate. It is desirable that *Tautology*$_{AS}$ is satisfied, except for the case in which the belief base is inconsistent. To this end we introduce the following weakening of *Tautology*.

CONSISTENT TAUTOLOGY$_\bullet$: If P is consistent, then $P * P_\top \equiv_\bullet P$.

The problem of the approaches not satisfying the *Tautology* postulate is that they make unnecessary changes not only for tautological revisions. Any revision by a program that has no semantic influence should not add any answer sets. This idea has been discussed in [9] and [237] and formalized for dynamic logic programming as the *refined extension principle* [9]. Here we formalize this idea in the following postulate for all $\bullet$ equivalences:

CONSISTENT IRRELEVANCE$_\bullet$: If P is consistent and $P \equiv_\bullet P \cup Q$, then $P * Q \equiv_\bullet P$.

Clearly *Consistent Tautology*$_\bullet$ follows from *Consistent Irrelevance*$_\bullet$ if and only if for all $P \in \mathcal{P}(\mathcal{L}^{\mathsf{asp}}_{At})$, $P \cup P_\top \equiv_\bullet P$. This holds for $\equiv_{AS}$, $\equiv_{UE}$, $\equiv_{SE}$ but not for $\equiv_P$.

Proposition 4.3.12. If $*$ satisfies Consistent Irrelevance$_\bullet$ and $\bullet \in \{AS, UE, SE\}$, then $*$ satisfies Consistent Tautology$_\bullet$.

Proof. See Appendix A.1.2 on Page 230. $\square$

Consistent Irrelevance$_\bullet$ is a postulate formulating some form of minimal change on the semantical level. Two basic postulates for minimal change of the base revision postulates, *Inclusion*$_Q$ and *Vacuity*$_Q$, are sufficient to guarantee *Consistent Irrelevance*$_\bullet$.

Proposition 4.3.13. If $*$ satisfies *Inclusion*$_Q$ and *Vacuity*$_Q$, then it satisfies Consistent Irrelevance$_\bullet$ for $\bullet \in \{AS, UE, SE, P\}$.

Proof. See Appendix A.1.2 on Page 230. □

From the previous two proposition follows directly that *Inclusion*$_Q$ and *Vacuity*$_Q$, are sufficient to guarantee *Consistent Tautology*$_\bullet$.

Corollary 4.3.14. If $*$ satisfies *Inclusion*$_Q$ and *Vacuity*$_Q$, then it satisfies *Consistent Tautology*$_\bullet$ for $\bullet \in \{AS, UE, SE\}$.

Proof. See Appendix A.1.2 on Page 230. □

These results are consistent with Proposition 4.3.11 which includes a negative result for the $\bullet = P$ case of Corollary 4.3.14.

We consider *Parallelism*$_\bullet$ and *Disjointness*$_\bullet$ together since the underlying idea is similar. The idea of these postulates, that parts of programs with disjoint alphabets should be independent in the revision process, is generally desirable. The problem with *Parallelism*$_\bullet$ and *Disjointness*$_\bullet$ is that the respective third program Q (P respectively) can contain rules connecting the disjoint sets of literals such that inconsistencies arise in combination of both sets of literals but not based on a single one. Consider the following example which demonstrates that *Disjointness*$_{AS}$ is in conflict with the principle of minimal change in base revision.

Example 4.3.15. Let $P = \{a., b.\}$ such that $P = P_1 \cup P_2$ with $P_1 = \{a.\}$ and $P_2 = \{b.\}$, and $Q = \{\neg a \leftarrow b.\}$. For the revision of $P_1 * Q$ we can note that there is no conflict and $AS(P_1 \cup Q) = \{\{a.\}\}$ such that there is no need to change anything, such that $P_1 * Q = P_1 \cup Q$ seems to be a reasonable revision. The same holds for $P_2 * Q$ with $AS(P_2 \cup Q) = \{\{b.\}\}$. The Disjointness$_{AS}$ postulate demands that $AS(P * Q) = AS((P_1 * Q) \cup (P_2 * Q))$. In this case $AS((P_1 * Q) \cup (P_2 * Q)) = \emptyset$ which is clearly not a desirable outcome for $AS(P * Q)$. ◇

We can show the following conflicts of *Parallelism*$_\bullet$ and *Disjointness*$_\bullet$ with the base revision postulates.

Proposition 4.3.16. Let $*$ be a revision operator on logic programs.

1. If $*$ satisfies *Success*$_Q$, *Consistency*$_Q$ and *Vacuity*$_Q$, then it violates *Parallelism*$_{AS}$.

2. If $*$ satisfies *Vacuity*$_Q$ and *Consistency*$_Q$, then it violates *Disjointness*$_{AS}$.

Proof. See Appendix A.1.2 on Page 230. □

Disjointness$_{AS}$ is in conflict with the minimal change of the revisions of $P_1 * Q$ and $P_2 * Q$. In order to satisfy *Disjointness*$_{AS}$ the revisions of $P_1 * Q$ and $P_2 * Q$ have to be cautious enough to anticipate possible inconsistencies with additional input. Here we consider that inconsistencies with some input should be handled by a revision operator applied to this input and not by a previous revision.

We weaken the postulates as follows:

WEAK DISJOINTNESS$_\bullet$: If $P = P_1 \cup P_2$ and P_1 and P_2 have disjoint sets of literals $\mathcal{A}_1$ and $\mathcal{A}_2$ and for each set of literals $\mathcal{A}_r$ of a rule $r \in Q$ it holds $\mathcal{A}_r \cap \mathcal{A}_1 = \emptyset$ or $\mathcal{A}_r \cap \mathcal{A}_2 = \emptyset$, then $P * Q \equiv_\bullet (P_1 * Q) \cup (P_2 * Q)$.

WEAK PARALLELISM$_\bullet$: If Q_1 and Q_2 have disjoint sets of literals $\mathcal{A}_1$ and $\mathcal{A}_2$, and for each set of literals $\mathcal{A}_r$ of a rule $r \in P$ it holds $\mathcal{A}_r \cap \mathcal{A}_1 = \emptyset$ or $\mathcal{A}_r \cap \mathcal{A}_2 = \emptyset$, then $P * (Q_1 \cup Q_2) \equiv_\bullet (P * Q_1) \cup (P * Q_2)$.

These weakened versions of *Disjointness*$_\bullet$ and *Parallelism*$_\bullet$ still express the idea of independence as formulated in the original postulates but are not in conflict with the proposed set of base revision postulates. They are also still strong enough such that they do not follow from the base revision postulate set. Therefore we add them to our set of desirable postulates. We show later that they are implied by a postulate for belief base consolidation operators.

To sum up, we have shown that all postulates for ASP revision that are not in conflict with the base revision setting follow from the base revision postulates. For those ASP revision postulates that are in conflict with the base revision postulates we gave adequately weakened versions. Therefore, in the next section we are looking for a constructive approach for an *ASP multiple base revision operator* defined as follows.

Definition 4.3.17. An *ASP multiple base revision operator* is a multiple base revision operator $*$ that satisfies
Success$_Q$, Inclusion$_Q$, *NM-Consistency*$_Q$, *Fullness*$_Q$, Uniformity$_Q$, Weak Disjointness$_P$ and *Weak Parallelism*$_P$.

As shown above, such an operator also satisfies

Vacuity$_Q$, Consistency$_Q$, Relevance$_Q$, *Initialisation*$_\bullet$, *Idempotence*$_\bullet$, *Absorption*$_\bullet$, *Consistent Irrelevance*$_\bullet$

for $\bullet \in \{AS, UE, SE, P\}$, and

Consistent Tautology$_\bullet$

for $\bullet \in \{AS, UE, SE\}$.

4.3.2 *Construction of ASP Base Revision*

In Section 2.5.3 (Page 30) we described the well known construction of a revision operator $*$ by means of a contraction operator $-$ and an expansion operator $+$. It is called the *Levi-Identity* and formally defined as:

$$\mathcal{B} * \phi = (\mathcal{B} - \neg\phi) + \phi.$$

The direct application of this construction for answer set programming does not work, because neither the negation nor the inference of a rule is defined and inconsistency cannot be reduced to complementary literals. Even in the very restricted case of a revision of some program P by new information $Q = \{L.\}$ consisting of a single fact, it would not be sufficient to contract such that $\neg L \notin \cap AS(P)$.

So we have to look for different constructions which implement the idea of the Levi-Identity while being adequate for the ASP case. The idea of the Levi-Identity is, that after contracting by $\neg\alpha$ the belief base is consistent with α. That is, the belief base is made consistent with the information to be added before adding it. This does not work in general for the logic programming case because the dependencies within a program are complex and cannot be anticipated without including the input program. The inconsistency of a program P with a new program Q can only be determined by considering $P \cup Q$, as the interaction of rules of both programs generates the inconsistency. Base revision constructions of this type are called *external revision* since a sub-operation takes place outside of the original set. Hence the base revision construction for logic programs has to consider $P \cup Q$ to determine inconsistency. From $P \cup Q$ rules are removed such that the resulting program is consistent with Q. This idea amounts to the consolidation of $P \cup Q$ under certain constraints. In the base revision literature the unary operator ! is called a consolidation operator and results in a consistent subset of the input. A consolidation operator has been used in [125] to define *semi-revision*, which is defined as $\mathcal{B} *_{?} \alpha = (\mathcal{B} \cup \alpha)\,!$. The problem with semi-revision for our means is that it is non-prioritized, i. e., the *Success*$_Q$ postulate is not satisfied in general.

We extend the idea of the semi-revision construction to be able to define a prioritized revision operator for logic programs. To this end we define a *screened consolidation operator* $!_R$ with R being a set of core sentences that are immune to change like in screened revision [172]. We propose the following set of postulates for a screened consolidation operator:

SCREEN$_!$: $R \subseteq P!_R$.

SCREEN-CONSISTENCY$_!$: If there exists some consistent X, $R \subseteq X \subseteq P$ then $P\,!_R$ is consistent.

INCLUSION$_!$: $P\,!_R \subseteq P$

RELEVANCE$_!$: If $r \in P$ and $r \notin P!_R$, then there is a set P' such that $P\,!_R \subseteq P' \subseteq P$ and that P' is consistent but $P' \cup \{r\}$ is inconsistent.

FULLNESS$_!$: If $r \in P$ and $r \notin P!_R$, then $P!_R$ is consistent and $P!_R \cup \{r\}$ is inconsistent.

NM-FULLNESS$_!$: If $R' = P \setminus P!_R \neq \emptyset$, then $P!_R$ is consistent and for all $R'' \subseteq R'$ it holds that $P!_R \cup R''$ is inconsistent.

SCREEN-UNIFORMITY$_!$: Let R, R' and P be sets of rules and R and R' be consistent. If for all $X \subseteq P$, $R \cup X$ is consistent iff $R' \cup X$ is consistent, then $P \cap (P \cup R)!_R = P \cap (P \cup R')!_{R'}$.

Note that as in the classic case satisfaction of *Fullness*$_!$ implies satisfaction of *Relevance*$_!$.

Proposition 4.3.18. Let $!_R$ be a screened consolidation operator. If $!_R$ satisfies *Fullness*$_!$, then it satisfies *Relevance*$_!$.

Proof. See Appendix A.1.2 on Page 231. □

Further, as we already showed for the corresponding revision postulates, the satisfaction of *NM-Fullness*$_!$ implies the satisfaction of *Fullness*$_!$.

Proposition 4.3.19. Let $!_R$ be a screened consolidation operator. If $!_R$ satisfies *NM-Fullness*$_!$, then it satisfies *Fullness*$_!$.

Proof. See Appendix A.1.2 on Page 231. □

We consider this as a basic set of postulates for a screened consolidation operation.

Definition 4.3.20 (Screened Consolidation). An operator $!_R$ is an operator of screened consolidation if and only if it satisfies *Screen*$_!$, *Screen-Consistency*$_!$, *Inclusion*$_!$, *NM-Fullness*$_!$ *and Screen-Uniformity*$_!$.

As stated before, the new postulates *Weak-Disjointness*$_\bullet$ and *Weak-Parallelism*$_\bullet$ do not follow from the basic set of belief base postulates. This is also true for the postulate set for screened consolidation presented here. In the following we show that the property of *topic independence* as defined in [125] implies both postulates. The formulation of *topic independence* bases on the notion of *topicalizations*, which we define in the next definition.

Definition 4.3.21. [125] Let P be a set and $\mathfrak{P}(P)$ the powerset of P. A set $\mathfrak{B} \subseteq \mathfrak{P}(P)$ is a *topicalization* of P if and only if:

1. $P = \bigcup \mathfrak{B}$

2. If $R \subseteq P$, then R is consistent if $R \cap B$ is consistent for all $B \in \mathfrak{B}$

A topicalization $\mathcal{B}$ of P is hence a cover of P for which it holds that the consistency of each subset of P is only dependent on the consistency of its projection for each topic $B \in \mathcal{B}$.

TOPIC-INDEPENDENCE$_!$: [125] If $\mathfrak{B}$ is a topicalization of P, then
$P \setminus P!_R = \bigcup_{B \in \mathcal{B}}(B \setminus B!_R)$

The postulate of *Topic-Independence*$_!$ demands that given a topicalization of P each rule that has been removed from P by the consolidation operation $!_R$ is also removed from each topic $B \in \mathcal{B}$ with $r \in B$ by the respective consolidation of the topic and each rule removed by all consolidations of topics of P containing it is removed from P.

Definition 4.3.22. Given a screened consolidation operator $!_R$ we define a *multiple ASP base revision operator* as:

$$P * Q = (P \cup Q)!_Q$$

We can show that this construction leads to a *multiple ASP base revision operator* that satisfies all desirable properties, if $!_R$ satisfies all desirable properties.

Proposition 4.3.23. Let $*$ be a multiple base revision operator defined as $P * Q = (P \cup Q)!_Q$. If $!_R$ is a screened consolidation operator that satisfies Topic-Independence$_!$, then $*$ satisfies
Success$_Q$, *Inclusion*$_Q$, *Vacuity*$_Q$, *Consistency*$_Q$, *NM-Consistency*$_Q$, *Relevance*$_Q$, *Fullness*$_Q$, *NM-Fullness*$_Q$, *Uniformity*$_Q$, *Weak-Disjointness*$_P$ and *Weak-Parallelism*$_P$.

Proof. See Appendix A.1.2 on Page 231. □

We based the revision operation on a consolidation operation and characterized the latter by a set of postulates. We need a construction of a screened consolidation operator that is suitable for the application to logic programs. Two constructions of consolidation operations are available from belief base theory: partial meet and kernel consolidation [125]. We use a partial meet construction here. We start by defining a screened version of remainder sets for logic programs in which all remainder sets contain the screened set of rules.

Definition 4.3.24 (Screened Remainder Sets). For sets of sentences P and R with $R \subseteq P$ the set of *screened consistent remainder sets* of P, denoted by $P \perp_! R$ is such that for each $X \in P \perp_! R$:

1. $R \subseteq X \subseteq P$

2. X is consistent

3. There is no X' such that $X \subset X' \subseteq P$ and X' is consistent

The definition of screened remainder sets differs from the original formulation only in the first condition, which is $X \subseteq P$ originally. In contrast to the classical case, the intersection of remainder sets of logic programs is not necessarily consistent due to the non-monotonicity of logic programs.

Example 4.3.25. Let $P = \{a \leftarrow b., \neg a., b., \leftarrow \mathsf{not}\ \neg a, \mathsf{not}\ b.\}$. The set of remainder sets with the empty screen are

$$\begin{aligned} P \perp_! \emptyset = \{ & \{a \leftarrow b., b., \leftarrow \mathsf{not}\ \neg a, \mathsf{not}\ b.\}, \\ & \{\neg a., b., \leftarrow \mathsf{not}\ \neg a, \mathsf{not}\ b.\}, \\ & \{a \leftarrow b., \neg a., \leftarrow \mathsf{not}\ \neg a, \mathsf{not}\ b.\}\} \end{aligned}$$

The intersection of the first two remainders, $\{b., \leftarrow \mathsf{not}\ \neg a, \mathsf{not}\ b.\}$, is consistent. The intersection of the first and the last remainder, $\{a \leftarrow b., \leftarrow \mathsf{not}\ \neg a, \mathsf{not}\ b.\}$, is inconsistent. $\diamondsuit$

This leaves us with the option of selecting exactly one of the remainders, which also has the advantage of leading implying less change. A selection function is specific to a certain belief base. To be able to iterate the change process we define a global selection function by making use of a *two place selection function* [106] with the belief base as a parameter.

Definition 4.3.26 (Global maxichoice selection function). Let P be a set of sentences. The function γ_P is a *selection function* for P if and only if for all sets of sentences R

1. If $P \perp_! R \neq \emptyset$, then $\gamma_P(P \perp_! R) = X$ for some $X \in P \perp_! R$.
2. If $P \perp_! R = \emptyset$, then $\gamma_P(P \perp_! R) = P$.

A *global maxichoice selection function* is a function γ such that for each $P \subseteq \mathcal{L}$, $\gamma(P, \cdot) = \gamma_P(\cdot)$ is a selection function for P. We drop the index P if it is obvious.

A global maxichoice selection function is globally defined for all belief bases. Hence, in contrast to one place selection functions, global maxichoice selection functions are not specific to one particular belief base. This makes it possible to apply the resulting consolidation operator, and the resulting revision operator, iteratively. Moreover, properties of consolidation operators which connect the consolidation results of several dependent belief bases can only be expressed for global maxichoice selection functions. We define a screened maxichoice consolidation operation based on a global maxichoice selection function.

Definition 4.3.27. Let P and R be sets of sentences and γ a maxichoice selection function for P. The operation $P\,!_R$ such that

$$P\,!_R = \gamma(P \perp_! R)$$

is a *screened maxichoice consolidation* based on γ.

In the following representation theorem we show that any screened maxichoice consolidation satisfies the set of postulates for screened consolidation and, moreover, that any operation satisfying these postulates can be constructed as a screened maxichoice consolidation.

Proposition 4.3.28. An operation $!_R$ is an operation of *screened maxichoice consolidation* if and only if it satisfies *Inclusion*$_!$,
Screen-Consistency$_!$, *Screen*$_!$, *NM-Fullness*$_!$ and *C-Uniformity*$_Q$.

Proof. See Appendix A.1.2 on Page 234. □

Finally, we link *maxichoice consolidation operation* to the postulate *Topic-Independence*$_!$ by a notion of *monotony* for selection functions. First we define what it means for a *global selection function* to be *monotone*.

Definition 4.3.29. A *global selection function* γ is *monotone* if and only if for all $P, P' \subseteq \mathcal{L}$, if for each $X \in P \perp_! \emptyset$ there exists some $X' \in P' \perp_! \emptyset$ such that $P \setminus X = P \setminus X'$, then $P \setminus \gamma(P \perp_! \emptyset) = P \setminus \gamma(P' \perp_! \emptyset)$.

An operator of screened consolidation is *monotone* if and only if it is based on a monotone selection function.

Now we can show that *monotone* selection functions lead to consolidation operators that satisfy *topic-indepence*.

Proposition 4.3.30. If a *screened maxichoice consolidation operator* is based on a *monotone maxichoice selection function*,
then it satisfies *Topic-Independence*$_!$.

Proof. See Appendix A.1.2 on Page 235. □

Hence, we just completed to show that our construction of a *multiple ASP base revision operator* on the basis of a *screened maxichoice consolidation operator* based on a *monotone maxichoice selection function* satisfy all of the desired properties.

4.4 RELATED WORK

In this chapter we elaborated a general conceptualization of the structure of change operations for epistemic agents. Such a general concept with a strong connection to belief change theory has, to the best of our knowledge, not been presented previously. In any multiagent programming framework percepts have to be interpreted and used to change the internal state of an agent. This is usually done specifically for the target system in each framework without following a general concept [51]. Closest to a general formulation of an interpretation function comes the specification of semantics of agent communication languages. Diverse languages and semantics thereof have been developed, see [248] for an overview. For instance for the widespread *Fipa ACL* communication language the *rational effect* of each speech act is expressed by a logical formula of the *SL* language [101]. Hereby the intended effect of a speech act is specified, but autonomous agents cannot be guaranteed to adhere to this intended effect. That is, what the receiving agent makes out of a received speech act is dependent of that specific agent. This is exactly what we formalize here by means of the structure of the change operator, the effect of the interpretation function in combination with the selection and revision operator and the knowledge representation and reasoning of the agent.

In terms of related work for our work on the selective revision operation there are mainly two areas that are related. On the one hand, non-prioritized belief revision and on the other hand belief revision by argumentation. In relation to the former area we instantiated and extended the non-prioritized revision operator of selective revision presented in [96] towards multiple revision and to revision of belief bases.

In [126] an overview and classification of *non-prioritized revision operators* is given. By this classification *selective revision* is one of the most general *non-prioritized revision operator* of the type *decision+revision* . Moreover it allows for partial acceptance of the input, in contrast to most other approaches. Apart from *decision+revision* approaches there are *expansion+consolidation* approaches to non-prioritized belief revision. These perform a simple expansion by the new information, i.e. $\mathcal{B} \cup \Phi$, and then apply a consolidation operator ! that restores consistency, i.e. $\mathcal{B} * \Phi = (\mathcal{B} \cup \Phi)!$. This approach is limited to belief bases and cannot be used for belief sets. This is the case since inconsistent belief sets cannot be distinguished, they have to be deductively closed and they are inconsistent, hence (*ex falso quodlibet*) if follows that they consist of the entire propositional language $\mathcal{L}^{\mathsf{prop}}_{\mathtt{At}}$, i. e., $\mathrm{Cn}(\perp) = \mathcal{L}^{\mathsf{prop}}_{\mathtt{At}}$. An instantiation of such an operator that is similar to the setup used in this section has been presented in [90]. The considered input to the revision consists of a set of sentences that form an explanation of some claim in the same form as the argument definition used here. However, as with all approaches of the type *expansion+consolidation*, new and old information are completely equal to the consolidation operator. In contrast, the approach presented here makes use of two different mechanisms to first decide about if, and which part, of the input shall be accepted just considering the new information, and then performing prioritized belief revision of the old information. Also, there are *integrated choice* approaches that do not feature a two step process but a single step process applying the same technique for the selection and revision process. Mostly these approaches need some meta-information, e. g., an epistemic entrenchment relation, and thus differ on the basic process as well as on the information needed.

While there has been some work on the revision of argumentation systems, very little work on the application of argumentation techniques for the revision process has been done so far, cf. [91]. In fact, the work most related to the work presented here makes use of negotiation techniques for belief revision [48, 251], without argumentation. In the general setup of [48] a symmetric merging of information from two sources is performed by means of a negotiation procedure that determines which source has to reduce its information in each round. The information to be given up is determined by another function. The negotiation ends when a consistent union of the information from both sources is reached. While this can be seen as a one step process of merging or consolidation in general, the formalism also allows to differentiate between the information given up from the first source and the second source. In [48], this setting is then successively biased towards prioritizing the second source which leads to representation theorems for operations that are equivalent to selective revision operators that satisfy *Consistent expansion*, and for classic AGM operators. While those results are interesting, the negotiation framework used in [48] is very different from the argumentation formalism used here and also very different from the setup of selective revision. Moreover, the functions for the negotiation and concession are left abstract.

In [251] mutual belief revision is considered where two agents revise their respective belief state by information of the other agent. Both agents agree in a negotiation on the information that is accepted by each agent. The revisions of the agents are split into a selection function and two iterated revision functions which leads to operators satisfying *Consistent expansion*. The selection function is then a negotiation function on two sets of believes that represent the sets of belief that each agent is willing to accept from the other agent that might obey game theoretic principles. This setting has a very different focus as ours and also does not specify the selection function.

The majority of work on the dynamics in logic programming has focused on the implementation of syntactic inconsistency handling in sequences of logic programs via logic programs, e. g., [9, 75, 87, 252, 152, 151]. Most of these appraoches have not been considered in a principled way. Some of them have been considered with respect to the AGM postulates for belief set revision and adaptations of these for logic programming. It was shown that they fail to satisfy most of them in [87]. Alternative postulates were developed thereupon, e. g., [87, 9]. We discussed these thoroughly in our work. In terms of the satisfaction of the AGM postulates for belief set revision semantic approach that make use of the monotonic SE-Models of logic programs where more successful and could be shown to satisfy them [68, 72, 30].

The closest work to a base revision approach for logic programs is the state transition system approach presented in [163] which was meant to satisfy base revision postulates for the revision, but no formal results are shown. In [133] the problem of belief base merging is considered for logic programs. In particular the syntactic approach of *removed sets fusion* for propositional belief base merging [132] is applied to belief bases represented by logic programs. The base revision approach and the corresponding postulates for it have not been considered, neither the postulates for change operations on ASP presented in [87]. Approaches to belief revision in other non-monotonic formalisms are often not based on principles and very specific to the underlying logic [245, 29]. A principle based approach to contraction operations in logic programs has been considered by us in [153].

4.5 CONCLUSION

In this chapter we fleshed out the general concept of a change operator, which we introduced in Chapter 3. We did so by proposing a compound change operator, and by elaborating on its sub-operations for belief bases in combination with non-monotonic reasoning.

The change operator for a belief base comprises an interpretation function, a selection function and a prioritized multiple base revision operator. The last two operators form a non-prioritized multiple base revision operator as an instance of the selective revision scheme. We described the functionality of the interpretation function we introduced.

Then, we took up the existing research on selective revision and combined it with deductive argumentation in order to implement selection functions that only accept those parts of the new information that are not refuted in the light of the belief base under consideration. We took some first steps in investigating the properties of our proposed construction and were able to show that the resulting operator complies with many desirable properties for non-prioritized revision. We discussed the performance of our operators by examples and compared our approach to related work.

Following on this we turned to the elaboration of multiple base revision operators as needed as the inner revision operator in combination with a selection function to form a non-prioritized multiple base revision. In particular, we developed a declarative description of multiple base revision operators for answer set programming. For this, we formalized adequate adaptions and extensions to the base revision postulates for propositional belief bases for the non-monotonic formalism of answer set programming. Further, we considered a set of postulates specific to ASP change operations proposed in [87]. We generalized these for different notions of equivalence of answer set programs and adapted them to our formalization of change operations. We showed that our adapted set of base revision postulates implies several of them. Moreover, we presented a novel constructive approach for a prioritized multiple base revision operator on the basis of a novel *screened consolidation operation*. We used a global selection function leads to a general revision operator that can be used iteratively. We proved a representation theorem for our construction. Moreover, we showed that the property of *Topic Independence* that has been formulated for propositional belief bases implies the ASP specific postulates of *Weak Disjointness* and *Weak Parallelism*. We defined a *monotony* property for selection functions on remainder sets and showed that monotone selection functions lead to consolidation operators that satisfy *Topic Independence*. The results we presented show that and how the base revision approach is applicable to non-monotonic formalisms, by example of answer set programming.

To sum up, we presented a complete view of the change process for epistemic components of epistemic agents, which we introduced in Chapter 3. We made use of, and extended, existing work in belief revision theory for some sub-operations of our considered change process. In particular we combined belief revision operators with non-monotonic formalisms. We concretized the abstract theory of selective revision by use of argumentation theory. Further, we applied the base revision scheme to non-monotonic formalisms in general and answer set programming in particular. We presented constructive approaches for the change operators we considered. These can be used in combination with the existing implementations of deductive argumentation and answer set programming. Hence these can be used in practical multiagent systems where uncertain dynamic information has to be handled adequately.

5 AGENT-BASED SECRECY

In this chapter, we present an approach to secrecy for autonomous epistemic agents with incomplete and uncertain information in a dynamic environment. Existing work on secrecy in multiagent systems is focused on the specification of strong notions of secrecy. It is based on the assumptions that complete information about the system is available and that agents are perfect reasoners. In addition, the notions of secrecy that are defined in this setting are very strict and cautious, which leads to high constraints on the information flow [121, 232].

The conditions in realistic scenarios of autonomous agents in dynamic, uncertain environments, however, stand in stark contrast to the assumptions of a complete view on the system and perfect reasoners. Furthermore, having to preserve secrecy restricts the possibilities of an agent to act and thereby to achieve its other goals. Hence, secrecy requirements that are way more cautious than necessary lead to an unnecessary loss of performance of the agents. Ideally, agents should be able to take secrecy into consideration and should be able to evaluate the effects of their actions with respect to secrecy in comparison with their utility – they should be *secrecy aware*. As also observed in [121] a major task for future work on secrecy is the "careful consideration of how secrecy definitions can be weakened to make them more useful in practice".

While a substantial body of work on the definition of secrecy exists, mechanisms for secrecy preservation in multiagent systems are lacking. The only elaborate practical approaches to secrecy preservation can be found in the database community, e. g., [213, 32, 46, 38, 34]. These consider the *inference* problem in different database settings and also take *meta-inferences* of attackers into consideration. However, these are also based on the assumption of a complete view and a perfect attacking agent. Further, they are limited to a client-server interaction in which the client poses queries to the server, to which the server replies.

We develop a new *agent-based* notion of secrecy as well as a complete secrecy agent model and algorithms for it. Our approach differs largely from existing approaches to secrecy. In particular, this notion of secrecy is based on the uncertain information of an agent with limited reasoning capabilities. Moreover, our secrecy agent model incorporates secrecy into the agent's reasoning and deliberation.

A defending agent has a view on the world-view of potential attacking agents, which we call an *agent-view*. In contrast to other works we consider a symmetric setting in which all agents might have, individual, secrets with respect to other agents. We consider two *dimensions of uncertainty* for agent-views. The uncertainty of the attacking agent about the state of the world, the *uncertainty of attacker's world-view*, and the uncertainty of the defending agent about the uncertain world-view of the attacking agent, *uncertainty of defenders agent-view*. Also, different from other approaches, we define a secret with respect to a particular reasoning behavior. This way we declare sensitive information, represented by a logical formula, as secret to different degrees. That is, a *secret* declares a logical formula to be kept secret, and a reasoning behavior, including meta-inferences, against which it shall be protected. We also consider that an agent might change its secrets during its course of action.

Based on this notion of secrets and uncertain agent-views we define a notion of a secrecy preserving agent that never performs any secrecy violating action. Further, we go beyond this strict preservation of secrecy and, realistically, consider agents that have to, or decide to, violate secrecy. In practical agent systems, agents are usually not assumed to have the means to maintain a state with certain properties [82]. They have to have the means to prevent violations as good as possible, to violate secrecy as little as possible, and to recover secrecy as good as possible. For this, we design agents to take secrecy into consideration in their reasoning and deliberation on all levels. We call such agents *secrecy aware agents*. *Secrecy preserving agents* are then a special case of *secrecy aware agents*.

We enable our secrecy aware agents to evaluate their options for action with respect to secrecy not only on a binary (secrecy violating or not) basis, but by ordering the considered options on the basis of their potential degree of secrecy violation as fine grained as possible. We do this in a principled manner by first determining a set of principles for such a classification of actions. Then, we develop constructive approach and present a classification algorithm that satisfies the principles.

We consider general settings for secrecy and define properties of these. In particular we consider settings in which an agent is able to preserve secrecy and what it takes to guarantee the preservation. We show that in general it is necessary to consider the possible future actions of the agents by use of notions from game theory. An agent has to act such that its attackers do not get a winning strategy to violate its secrecy.

We consider a query-answer setting that correspond to a client-server database scenario [213, 32, 46, 38, 34]. We show that the key result of [213], that it is necessary to protect the disjunction of all potentially secret formulae, also holds for our model in this setting. In our setting this result is a special case of our consideration of future actions and winning strategies of the attacker.

We consider different sets of options for action that an agent considers and preferences it has over alternative options based on secrecy and informativity. These considerations amount to the consideration of *distortion strategies* such as *lying* and *refusing* implemented in the literature on confidentiality in information systems [32] from an agent based perspective.

Apart from the consideration on the general level we develop concrete types of agents. In particular we instantiate our general model for the BDI paradigm as our focus for agent architectures, and ASP and propositional logic as our foci for (non-monotonic) knowledge representation and reasoning. We use our BDI^+ agent model to define *secrecy aware* agents and instantiate them with ASP for the knowledge representation. We develop an ASP based approach to model the *meta-information* of *speech acts* and the *meta-inferences* of the agents. We exemplify our approaches by completely modeling our *strike committee meeting* example, which we introduced in Section 1. Furthermore, we compare our approach to the relevant, most important works from computer security, security in information systems, and multi-agent systems.

The remainder of this chapter is structured as follows. In Section 5.1 we motivate and describe the challenges we address and the properties we demand to be satisfied in our notion of agent-based secrecy. In Section 5.2 we formalize our notion of secrecy and secrecy preservation for epistemic agents based on the general epistemic agent framework we elaborated in Chapter 3. Then, we illuminate the role of change operations for secrecy in our setting in Section 5.3 and develop properties and operators for them. In Section 5.4 we elaborate on the possibilities and requirements for the preservation of secrecy by defining different types of settings. We define a game-theoretic notion of our secrecy scenario and show formal results for different settings. In Section 5.5 we turn to the deliberation of an agent and consider a general model of preferences on actions with respect to secrecy preservation. We define principles for the classification of action with respect to secrecy and devise an algorithm that satisfies all principles. In Section 5.6 we concretize our considerations by combining the BDI^+ agents we defined in Chapter 3 with our secrecy model and define concrete ASP based BDI^+ agents for secrecy preservation. We develop an ASP based representation of communication on the information and the meta-information level. We discuss relations to other works in Section 5.7 and conclude in Section 5.8.

5.1 DESIDERATA OF AGENT-BASED SECRECY

In this section we introduce the fundamental ideas and properties of an agent-based notion of secrecy. We start by describing our scenario and its challenges with respect to secrecy. Then, we motivate and illustrate the main aspects of our approach by means of examples.

In our considered scenario of an autonomous agent, agents reason under uncertainty about the state of the environment and possible courses of action. The agent pursues its goals by performing actions in the environment, in our case by exchanging speech acts with other agents. On the one hand, the exchange of information with other agents is often essential for an agent in order to achieve its goals, especially if the agent is part of a coalition. On the other hand the agent has a particular interested not to reveal specific pieces of information. Restriction of communication leads to a loss in performance of the individual agents, coalitions and the whole multiagent system. Therefore, a good solution of the implied conflict between the agent's goal to preserve secrecy and its other goals is one that restricts communication as little as necessary. We investigate how an agent can preserve secrecy given its uncertainty and the dynamics of the multiagent system. We call an agent *secrecy preserving* if it never performs an action that leads to the violation of a secret. In practice, however, it is likely that it is not possible for an agent to protect all of its secrets, or that it has to violate some secrets to some degree in order to achieve important other goals. For this it is important that the agent takes secrecy into consideration when acting and is aware of the effects of its actions with respect to secrecy and only performs a secrecy violating action if there is no better option for its means. In particular the agent has to be able to distinguish between different degrees of secrecy violation. We call such an agent *secrecy aware*. In this work we elaborate on *secrecy aware agents* and on properties such that they are *secrecy preserving*.

For the consideration of secrecy in this scenario the agent has to be aware of, and be able to reason about, the information available to other agents and their reasoning on the basis of this information. The beliefs of an agent change continuously upon reception of new information. For a secrecy aware agent this implies that, besides the changes of the agent's view on the world, the changes of the information available to other agents, changes to secrets, and the adaptation of the preferences on actions according to secrecy have to be considered. Secrets might, for instance, be changed if it is evident that they cannot be preserved in their current form and have to be weakened, or given up.

The fundamental questions for secrecy is the one of the definition of *secrets* and the definition of the *secrecy preservation*. We first motivate the requirements for the definition of secrets and for the definition of secrecy preservation by means of our example, as introduced in Chapter 1.

Example 5.1.1. *Emma* is working in a company for her boss *Beatriz*. She wants to attend a strike committee meeting (*scm*) next day and has to ask her boss for a day off in order to attend. It is general knowledge that the agent *Beatriz* puts every agent who attends the *scm* on her blacklist of employees to be fired next. *Emma* has to communicate with *Beatriz* in order to achieve her goal of getting a day off, but wants to keep secret from *Beatriz* that she wants to attend the *scm*. ♢

The first aspect of secrets we want to introduce and motivate is that they are not global, i. e., an agent has different secrets with respect to different agents.

Example 5.1.2. In our example, *Emma* wants to keep her attendance to the *scm* secret from *Beatriz* but not from other employees who also want to attend the *scm*. ♢

Secrets are also not uniform with respect to their strength. That is, an agent should be able to declare pieces of sensitive information, represented by logical formulae as secret to different degrees. We represent these differences by declaring an inference behavior against which the formula should be protected. This inference behavior can be more or less credulous, whereby a more credulous inference behavior represents a higher degree of protection of the formula.

Example 5.1.3. *Emma* does not even want her boss *Beatriz* to be suspicious about her attending the *scm* (secret with respect to a credulous reasoner). She also does not want other employees that are against the strike to know that she attends the *scm*.

However, with respect to her colleagues she considers it sufficient that they do not know for sure that she attends the *scm* (secrect with respect to a skeptical reasoner). ♢

These differences in the *strength of secrets* arise naturally from the value of the secret information. The value of secret information depends on the severeness of the negative effects, or the cost, for the agent resulting from the disclosure of the secret information.

Secrets are also not static, they arise, change and disappear during runtime of an agent such that it has to be able to handle these changes adequately.

Example 5.1.4. If *Emma* realizes that *Beatriz* is suspicious about her attending the *strike committee meeting* her secret, with a credulous inference behavior, is violated. Then she should still try to keep *Beatriz* from being sure that she attends the meeting. This can be reflected by changing the inference behavior against which the information shall be protected from a credulous behavior to a skeptical one. This way her secret is *weakened* but not given up. If *Emma* gets to know that *Beatriz* overheard her phone call with the strike committee, and hence is sure that *Emma* attends, *Emma* should give up her corresponding secret. ♢

These considerations lead to the following formulation of properties of secrets:

(S1) Secrets can be held with respect to specific agents

(S2) Secrets can vary in strength

(S3) Secrets can change over time

Having formulated properties of secrets, we continue with the formulation of properties of a secrecy preserving agent. We assume a multiagent system with a set of agents Ag . We use the agent identifier $\mathcal{X} \in \mathsf{Ag}$ to denote an arbitrary agent. Each agent has an epistemic state that contains a representation of the agent's views of the view on the world of other agents in Ag . For the presentation of the secrecy scenario we mostly focus on the interaction in a two agent system $\mathsf{Ag} = \{\mathcal{X}_1, \mathcal{X}_2\}$. When we model a concrete agent it has the role of the defending agent $\mathcal{D}$ and the other agent has the role of the attacking agent $\mathcal{A}$. Since we consider a symmetric scenario, in which all agents can be seen as defending agents and the other agents as attacking agents, we describe all agents this way. However, we formulate our definitions for the instantiated system general for arbitrary sets of agents. We do not explicitly consider the potential information flow between $\mathcal{A}$ and other agents. We do so implicitly by our consideration of uncertain, incomplete and changing views on other agents.

The intuitive formulation of our notion of secrecy preservation can be formulated as:

> *An agent $\mathcal{D}$ preserves secrecy if, in all possible situations, no secret formula ϕ that it believes to be true and that it wants to hide from agent $\mathcal{A}$ is, from $\mathcal{D}$'s perspective, believed by $\mathcal{A}$ in all possible situations after an action of $\mathcal{D}$ (given that $\mathcal{A}$ does not believe ϕ already).*

That is, we declare secrets as *potential secrets* (cf. e. g., [36]). This means that the formula contained in a secret only has to be protected if it is believed to be true by the agent holding the secret.

The actual quality of secrecy preservation is highly dependent on the accuracy of $\mathcal{D}$'s *attacker model*, i. e., the view of $\mathcal{D}$ on agent $\mathcal{A}$ including its reasoning capabilities. It also depends on $\mathcal{D}$'s information processing and adaptation of its view on $\mathcal{A}$ in the dynamic scenario. This includes a model of the meta-inferences drawn by $\mathcal{A}$ from the behavior of $\mathcal{D}$ and the background information of $\mathcal{A}$. In particular, a *secrecy aware agent* $\mathcal{D}$ should have the following properties:

(P1) Agent $\mathcal{D}$ is aware of the information it communicated to other agents and the meta-information conveyed to agent $\mathcal{A}$ by its actions

(P2) Agent $\mathcal{D}$ simulates the reasoning of agent $\mathcal{A}$

(P3) Agent $\mathcal{D}$ considers possible meta-inferences by $\mathcal{A}$ from its own behavior

A *secrecy preserving agent* should additionally have the following two properties:

(P4) Agent $\mathcal{D}$ never performs an action that leads to secrecy violation

(P5) Agent $\mathcal{D}$ weakens secrets only if new percepts imply violations of secrets and only as much as necessary to be able to preserve secrecy

These five properties are the foundation of our notion of secrecy preservation. As we shall see, the properties (P1) and (P5) are related to the belief change component of $\mathcal{D}$, (P2) and (P3) are related to the way $\mathcal{D}$ models $\mathcal{A}$, and (P4) is related to all stages of the deliberation and means-ends reasoning behavior of $\mathcal{D}$. In the remainder of this chapter, starting top-down with an abstract model, we elaborate the properties we describe here, formalize them, and show how they can be satisfied.

5.2 ABSTRACT AGENT MODEL FOR SECRECY

We build on the general model of an epistemic agent of Chapter 3 and start by defining secrecy relevant aspects of an epistemic state. In particular an epistemic state needs to contain a representation of the information available to other agents and a representation of the secrets of the agent. These can be represented explicitly, as components of the epistemic state, or implicitly as part of the information represented in a monolithic epistemic state. We capture both variants in our model by assuming that there are mappings from the epistemic state to epistemic components that represent certain aspects, such as the view on another agent.

A *secrecy epistemic state* $\mathcal{K}^s_\mathcal{D}$ is an epistemic state such that satisfies the following conditions: Agent $\mathcal{D}$'s view on the world, its *world-view*, is an epistemic component given by

$$V_W(\mathcal{K}^s_\mathcal{D}) \in \mathcal{L}_W.$$

The view agent $\mathcal{D}$ has on the world-view of agent $\mathcal{A}$, an *agent-view*, is an epistemic component given by

$$V_\mathcal{A}(\mathcal{K}^s_\mathcal{D}) \in \mathcal{L}_V.$$

The set of secrets of the agent is given by

$$\mathcal{S}(\mathcal{K}^s_\mathcal{D}) \in \mathcal{L}_\mathcal{S}.$$

The world-view language $\mathcal{L}_W$, the agent-view language $\mathcal{L}_V$ and the language of secrets $\mathcal{L}_\mathcal{S}$ are to be specified wrt. the used knowledge representation formalism.

In the following, we drop the subscript $\mathcal{D}$ and the superscript s from $\mathcal{K}^s_{\mathcal{D}}$ since it is clear that we consider the defending agent $\mathcal{D}$ and secrecy epistemic states in this chapter. The world- and agent-view are epistemic components as introduced in Section 3.2 (Page 39). We assume that $\mathcal{D}$ uses the belief operator $\mathsf{Bel}_{\mathcal{D}} : \mathcal{L}_W \to \mathcal{L}_{BS}$ to reason on its world-view.

Next, we detail our notion of secrets. Three aspects have to be specified. First, the information to be kept secret has to be expressed; this is done in the belief set language $\mathcal{L}_{BS}$. Second, the agent from which the information has to be kept secret has to be defined, as required in (S1), which is done by means of an agent identifier $\mathcal{X} \in \mathsf{Ag}$. Third, the strength of the secret, represented by the reasoning behavior towards which the information shall be protected, has to be expressed; as required in (S2). The reasoning behavior is represented by means of belief operators that are chosen from a given ordered family of belief operators $(\Xi, \preceq_{\mathsf{bel}})$, as defined in Definition 3.6.2 (Page 61), for the agent-view language $\mathcal{L}_V$, i. e., all $\mathsf{Bel} \in \Xi$ are of the type $\mathsf{Bel} : \mathcal{L}_V \to \mathcal{L}_{BS}$. Further, we assume that $(\Xi, \preceq_{\mathsf{bel}})$ satisfies *credulity* as defined in Section 3.6 (Page 61). Hence, in our framework secrets are triples specifying the information to be kept secret, the belief operator with respect to which the information shall be kept secret and the agent towards which the agent holds the secret.

Definition 5.2.1 (Secrets). A *secret* is a triple $(\phi, \mathsf{Bel}, \mathcal{A})$ and consists of a formula $\phi \in \mathcal{L}_{BS}$, a belief operator $\mathsf{Bel} : \mathcal{L}_V \to \mathcal{L}_{BS}$, $\mathsf{Bel} \in \Xi$ and an agent identifier $\mathcal{A} \in \mathsf{Ag}$. The set of secrets of agent $\mathcal{D}$ is denoted by $\mathcal{S}(\mathcal{K}_{\mathcal{D}})$. The language $\mathcal{L}_{\mathcal{S}}$ is the set of all such triples.

We call a secret whose formula is believed by the agent holding the secret an *active secret*. A formula is a *potentially secret formula* if it is contained in a secret, and it is called a *secret formula* if it is contained in a secret and it is believed by the agent holding the secret. We call a formula *sensitive* if itself or its negation is contained in a secret. In the following we assume that $\mathcal{D}$ beliefs all of the formulae of its secrets to be true, i. e., all secrets are *active* and all formulae of secrets are *secret formulae*. Further, we assume that for each formula ϕ there is at most one secret with respect to this formula in the set of secrets.

The semantics of a secret is that if agent $\mathcal{D}$ has the secret $(\phi, \mathsf{Bel}, \mathcal{A})$, i. e., $(\phi, \mathsf{Bel}, \mathcal{A}) \in \mathcal{S}(\mathcal{K}_{\mathcal{D}})$, and it believes ϕ to be true, i. e., $\phi \in \mathsf{Bel}_{\mathcal{D}}(V_W(\mathcal{K}))$, then it does not want that it is possible to infer ϕ from its view on agent $\mathcal{A}$ by use of the belief operator Bel, i. e., $\phi \notin \mathsf{Bel}(V_{\mathcal{A}}(\mathcal{K}_{\mathcal{D}}))$.

Example 5.2.2. We can formalize the secrets of *Emma* as $\mathcal{S}(\mathcal{K}_{Emma}) = \{(\mathsf{attend\_scm}, \mathsf{Bel}_{\mathsf{cred}}, \mathcal{B})\}$ with the proposition attend_scm expressing that *Emma* intends to attend the *strike committee meeting*. ◇

Since the belief operator family satisfies the *credulity* property for all $(V_{\mathcal{A}}(\mathcal{K}_{\mathcal{D}}))$ and all $\mathsf{Bel}, \mathsf{Bel}' \in \Xi$ with $\mathsf{Bel}' \preceq_{\mathsf{bel}} \mathsf{Bel}$ it holds that if

$$\phi \in \mathsf{Bel}(V_{\mathcal{A}}(\mathcal{K}_{\mathcal{D}})), \text{ then } \phi \in \mathsf{Bel}'(V_{\mathcal{A}}(\mathcal{K}_{\mathcal{D}})).$$

If we consider the two secrets $(\phi, \mathsf{Bel}, \mathcal{A})$ and $(\phi, \mathsf{Bel}', \mathcal{A})$ such that $\mathsf{Bel}' \preceq_{\mathsf{bel}} \mathsf{Bel}$, we can thus note that whenever the first one is violated, then also the second one is violated. But not vice versa. Therefore, the protection of the formula ϕ implemented by the secret $(\phi, \mathsf{Bel}, \mathcal{A})$ is *at least as strong as* the protection implemented by the secret $(\phi, \mathsf{Bel}', \mathcal{A})$, if $\mathsf{Bel}' \preceq_{\mathsf{bel}} \mathsf{Bel}$. The choice of a more credulous belief operator represents a stronger protection of the formula than the choice of a more skeptical belief operator.

Example 5.2.3. We formalize Example 5.1.4. We assume that *Beatriz* is "suspicious" about ϕ if she infers ϕ with the credulous belief operator $\mathsf{Bel}_{\mathsf{cred}}$, and that she "is sure" about ϕ if she infers ϕ with the skeptical belief operator $\mathsf{Bel}_{\mathsf{skep}}$. It holds that $\mathsf{Bel}_{\mathsf{skep}} \preceq_{\mathsf{bel}} \mathsf{Bel}_{\mathsf{cred}}$. Whether *Emma* has the secret $(\mathsf{attend\_scm}, \mathsf{Bel}_{\mathsf{cred}}, \mathcal{B})$ or the secret $(\mathsf{attend\_scm}, \mathsf{Bel}_{\mathsf{skep}}, \mathcal{B})$ might make a big difference. Assume that asking to get the day of the *strike committee meeting* off makes *Beatriz* being suspicious that *Emma* intends to attend the *strike committee meeting* but not certain. That is, the first secret would be violated but not the latter. If *Emma* has the first secret she can perform the action of asking for a day of without any restriction, if she has the second secret she cannot perform the same action without violating secrecy. Hence, this example shows that protecting the same formula with respect to a more credulous belief operator can lead to a stronger restriction on the actions of *Emma*. ♢

The belief operator that is declared in a secret represents the most credulous reasoning behavior against which the agent wants to protect the formula declared in the secret.

Observation 5.2.4. The properties (P2) and (P3) are addressed by the use of belief operators in combination with the agent-view and the change operators for the agent-view.

Secrecy is based on the subjective view $V_{\mathcal{A}}(\mathcal{K})$ agent $\mathcal{D}$ has on the beliefs of agent $\mathcal{A}$. If secrecy is violated or not, only depends on the epistemic state of the defending agent. These considerations lead us to the following definition of a *safe epistemic state*.

Definition 5.2.5 (Safe Epistemic State). An epistemic state $\mathcal{K}$ is *safe* if and only if for all $(\phi, \mathsf{Bel}, \mathcal{A}) \in \mathcal{S}(\mathcal{K})$ with $\phi \in \mathsf{Bel}_{\mathcal{D}}(V_{\mathcal{W}}(\mathcal{K}))$ it holds that $\phi \notin \mathsf{Bel}(V_{\mathcal{A}}(\mathcal{K}))$.

For a given set of secrets, we call the subset of secrets whose formula is believed to be true by $\mathcal{D}$ its *active secrets*. We formalize these as follows: Let V_W be agent $\mathcal{D}$'s world-view and $\mathcal{S}$ its set of secrets, then

$$\mathcal{S}_{\mathsf{active}}(V_W, \mathcal{S}) =_{def} \{(\phi, \mathsf{Bel}, \mathcal{A}) \in \mathcal{S} \mid \phi \in \mathsf{Bel}_{\mathcal{D}}(V_W)\} \tag{5.2.1}$$

is the set of active secrets with respect to V_W .

This far, we defined an epistemic state and a notion of a safe state for a fixed point in time. We want to define secrecy of an epistemic agent in a dynamic environment with the intuition that an agent *preserves secrecy* if it never performs any action that leads to an unsafe epistemic state. Given the agent's uncertainty about the actual state of the world and about the actual state of other agents it might receive percepts that reveal that a secret is in fact violated. This notion of the preservation of secrecy is hence dependent on the subjective view of the defending agent.

To formalize this intuition of secrecy we have to be able to make statements about the general behavior of an agent. To this end we define the set of *possible epistemic states* of an agent with respect to an *initial epistemic state* of the agent and a *set of possible percepts*. An agent starts with its initial state and then receives percepts as a result of the actions of other agents. A sequence of percepts resembles the agent's perspective on a run of the system. The percepts the agent receives are generated by the actions of the other agents of the multiagent system. We abstract away from the actual system and other agents and assume them to be represented by a set of possible percepts Per for the agent. That is, we assume that this set contains all percepts that the agent could receive in the system from other agents.

In a system the percepts result from the interaction of the agents such that the sequences of percepts are determined by the behavior and interaction of all agents in the system. All agents have the capability to execute any speech act at anytime. Consequently an agent can receive any speech act that is in its set of percepts at any time such that it can receive all of its possible percepts in any order. We formalize the set of possible states of an agent with respect to a set of possible percepts. This way we can describe properties of the behavior of an agent with initial agent state, given a set of possible percepts. These considerations result in the following definition:

Definition 5.2.6 (Possible states of an Agent). Let $(\mathcal{K}^0, \xi)$ be an agent state such that $\xi = (\circ, \circ^a, \mathsf{act})$ and the types of the operators are $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, $\circ^a : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$. We define the *possible epistemic states* of an agent with initial agent state $(\mathcal{K}^0, \xi)$ *possible epistemic states* $\Omega_\xi(\mathcal{K}^0, \mathsf{Per})$, denoted as $\Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ recursively as follows:

For a single percept $p \in \mathsf{Per}$ we define
$\Omega_\xi(\mathcal{K}, p) = \{\mathcal{K}\} \cup \{\mathcal{K} \circ p \circ^a \mathsf{act}(\mathcal{K} \circ p)\}$,
for a sequence of percepts $(p_1, \ldots, p_n)$ it is

$$\Omega_\xi(\mathcal{K},(p_1,\dots,p_n)) = \Omega_\xi(\mathcal{K},p_1) \cup \Omega_\xi(\Omega_\xi(\mathcal{K},p_1),(p_2,\dots,p_n))$$
and for a set of percepts
$$\Omega_\xi(\mathcal{K}^0,\mathsf{Per}) = \bigcup\{\Omega_\xi(\mathcal{K}^0,(p_1,\dots,p_n)) \mid \{p_1,\dots,p_n\} \subseteq \mathsf{Per}\}.$$

An agent preserves secrecy if its initial agent state is such that no unsafe epistemic state can be reached by any possible progression of its epistemic states.

Definition 5.2.7 (Secrecy-preserving Agent). Let $(\mathcal{K}^0,\xi)$ be a secrecy agent state such that the functional component $\xi = (\circ,\circ^a,\mathsf{act})$ is of the type $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, $\circ^a : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$. We call an agent with an initial agent state $(\mathcal{K}^0,\xi)$ *secrecy preserving* if and only if for all $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0,\mathsf{Per})$ it holds that $\mathcal{K}$ is safe.

Observation 5.2.8. The property (P3) is satisfied by this definition of a secrecy preserving agent.

The definition of secrecy preservation is dependent on the epistemic state of the agent and the agent's way of information processing. For instance, an agent $\mathcal{D}$ that does not reflect on the effects of its actions and that rejects any input information would never subjectively violate secrecy, as its view on $\mathcal{A}$ is static, as the following example illustrates.

Example 5.2.9. If *Emma* ignores that *Beatriz* said that she has evidence that *Emma* is going to attend the *strike committee meeting*, then *Emma* would, from her point of view, not violate secrecy. $\diamondsuit$

To exclude such irrational agents from our consideration we define and require basic properties of the epistemic states and the functional components of agents. As the example indicates in particular the operations that change the world-view of $\mathcal{D}$ and the view on the world-view of $\mathcal{A}$ are important to define rational secrecy aware agents.

5.3 PROPERTIES OF CHANGE OPERATIONS AND SECRECY

In this section we define desirable properties of change operators and their effect on the resulting notion of secrecy. As defined in Section 3.3 the change operators of an agent are capable of changing the epistemic state of an agent on the basis of a percept, or on the basis of an action of the agent itself. Both operations should have different properties, which we detail in the following.

There are two causes for changes of the agent's epistemic state, and thus two possible causes to become unsafe: after a change caused by a percept and after a change caused by an action of the agent. We demand, as formulated in (P4), that an agent's epistemic state does not become unsafe by a change induced by its own actions. Thus, the agent must not be able to preserve a safe epistemic state while acting by modifying its secrets. We formalize this by the following property.

SECRETS ACTION INVARIANCE$_{\circ^a}$ For all $a \in \mathsf{Act}$ it holds that
$\mathcal{S}(\mathcal{K} \circ^a a) = \mathcal{S}(\mathcal{K})$

As given by the definition of possible states, Definition 5.2.6, the actions by which an epistemic state is revised are the actions of the agent itself. This revision realizes the agent's consideration of the effects of its actions. The property *Secrets Action Invariance*$_{\circ^a}$ demands that no action of the agent should lead to a modification of the agent's secrets.

Example 5.3.1. When *Emma* considers to say to *Beatriz* that she intends to attend the *strike committee meeting* and realizes that this would violate her secret, she should not just give up her secret to be able to perform the action and preserve secrecy. ♢

The defending agent has an uncertain view on the information that is available to an attacking agent. Further, it does not have complete control of the information flow, i. e., there might be other sources of information for $\mathcal{A}$. Therefore, it is not possible for the defending agent to avoid violations of secrets based on information that comes from percepts. Hence, for the consideration of secrecy preserving agents we define the following property, requiring that the change of an epistemic state by a percept always results in a safe epistemic state.

ACKNOWLEDGMENT$_{\circ}$ For all $p \in \mathsf{Per}$ it holds that $\mathcal{K} \circ p$ is safe.

Consequently, the epistemic state of an agent with a change operator that satisfies *Acknowledgment*$_{\circ}$ can only become unsafe through the actions of the agent itself. This implies that all intermediate states, i. e., those after reception of a percept but before execution of an action, of the agent cycle are safe.

To be able to define further properties for the belief change operation of an epistemic state we decompose it into sub-operations on the epistemic state's components. For epistemic components we use the decomposition into sub-operations as introduced in Section 4.1. Based on these we motivate and define properties with respect to secrecy preservation, which formalize and concretize the ideas given in (P1) and (P5).

As introduced in Section 4.1 the change operation for an epistemic component of the language $\mathcal{L}_E$ for the set of speech acts Σ is composed by three sub-operations:

$$\circ_{\mathcal{L}_E,\Sigma} =_{def} t_{\mathcal{L}_E,\Sigma} \cdot f_{\mathcal{L}_E} \cdot *_{\mathcal{L}_E}.$$

First the interpretation function $t_{\mathcal{L}_E,\Sigma}$ interprets a speech act which results in a set of sentences of the language of the epistemic component under consideration. Then the selection function $f_{\mathcal{L}_E}$ evaluates the input and produces a set of sentences to be incorporated into the epistemic component. The latter task is carried out the by the inner revision operator $*_{\mathcal{L}_E}$. To ease the presentation, we disregard the selection function $f_{\mathcal{L}_E}$ in the following and assume it to be part of the inner revision operator, i. e., the inner revision operator might be a selective revision operator as defined in Section 4.2.1 (Page 77).

We assume that for an epistemic state $\mathcal{K}$ the world-view $V_W(\mathcal{K}) \in \mathcal{L}_W$ and the agent-views $V_{\mathcal{A}}(\mathcal{K}) \in \mathcal{L}_V$ of the agent are epistemic components and have epistemic component change operators $*_{\mathcal{L}_W}$ and $*_{\mathcal{L}_V}$, respectively, assigned to them. Then we assume that there are four interpretation functions that transform a speech act, as a percept or action, into a set of sentences of the epistemic component language of a world-view or an agent-view. In particular the interpretation function t_W is for percepts for the world-view of the agent, the function t_W^a for actions for the world-view, the function t_V one for the percepts for an agent-view, and t_V^a for actions for an agent-view.

Secrets are changed on the basis of the agent's changed beliefs and views such that the update operator for secrets is dependent on these and thereby indirectly also on the new information. Formally the *secrets change operator* has the form:

$$*_S : \mathcal{L}_S \times \mathcal{L}_W \times \mathcal{L}_V \to \mathcal{L}_S$$

We assume that the effect of $\circ$ and $\circ^a$ can be expressed by means of the component specific operators, such that

$$\begin{aligned} V_W(\mathcal{K} \circ p) &= V_W(\mathcal{K}) *_{\mathcal{L}_W} t_W(p), & (5.3.1)\\ V_{\mathcal{A}}(\mathcal{K} \circ p) &= V_{\mathcal{A}}(\mathcal{K}) *_{\mathcal{L}_V} t_V(p) \text{ and} & (5.3.2)\\ \mathcal{S}(\mathcal{K} \circ p), &= *_S(\mathcal{S}(\mathcal{K}), V_W(\mathcal{K} \circ p), V(\mathcal{K} \circ p)) & (5.3.3) \end{aligned}$$

and

$$\begin{aligned} V_W(\mathcal{K} \circ^a a) &= V_W(\mathcal{K}) *_{\mathcal{L}_W} t_W^a(a), & (5.3.4)\\ V_{\mathcal{A}}(\mathcal{K} \circ^a a) &= V_{\mathcal{A}}(\mathcal{K}) *_{\mathcal{L}_V} t_V^a(a) \text{ and} & (5.3.5)\\ \mathcal{S}(\mathcal{K} \circ^a a) &= \mathcal{S}(\mathcal{K}). & (5.3.6) \end{aligned}$$

The changes to the set of secrets in order to achieve a safe epistemic state should be minimal. That is, a secret should not be modified without a reason. A reason is only given if the respective secret is active and violated, then it should be weakened minimally such that it is not violated anymore.

We consider weakening by replacing the belief operator of a secret by a more skeptical belief operator. Only if there is no belief operator that could be used as replacement in a secret such that the secret is not violated, then the secret should be removed. The following property formalizes these considerations.

MIN-SECRECY-WEAKENING$_\circ$ (5.3.7)

If $(\phi, \mathsf{Bel}, \mathcal{A}) \in \mathcal{S}(\mathcal{K})$ and $(\phi, \mathsf{Bel}, \mathcal{A}) \notin \mathcal{S}(\mathcal{K} \circ p)$, then

1. $\phi \in \mathsf{Bel}(V_{\mathcal{A}}(\mathcal{K} \circ p))$ and $\phi \in \mathsf{Bel}_{\mathcal{D}}(V_{W}(\mathcal{K} \circ p))$
2. if for some $\mathsf{Bel}' \in \Xi$ it holds that
   a) $\phi \notin \mathsf{Bel}'(V_{\mathcal{A}}(\mathcal{K} \circ p))$ and
   b) there is no $\mathsf{Bel}'' \in \Xi$, $\mathsf{Bel}' \preceq_{\mathsf{bel}} \mathsf{Bel}''$ such that $\phi \notin \mathsf{Bel}''(V_{\mathcal{A}}(\mathcal{K} \circ p))$,

   then $(\phi, \mathsf{Bel}', \mathcal{A}) \in \mathcal{S}(\mathcal{K} \circ p)$.

Another secrecy relevant property of belief change arises from the changes to views of other agents. An agent should not be able to preserve secrecy by ignoring the effects of its own actions on the beliefs of potentially attacking agents. In particular the information for some agent $\mathcal{A}$ contained in an action of $\mathcal{D}$ should be incorporated into $\mathcal{D}$'s view on $\mathcal{A}$. We formalize this for $V_W(\mathcal{K})$ and $V_{\mathcal{A}}(\mathcal{K})$ being belief bases as follows:

AWARENESS$_{\circ^a}$ If $a \in \mathsf{Act}$ then $t^a_W(a) \subseteq V_W(\mathcal{K} \circ^a a)$ and for each $\mathcal{A} \in \mathsf{Ag}$ it is $t^a_V(a) \subseteq V_{\mathcal{A}}(\mathcal{K}_{\mathcal{D}} \circ a)$.

There might very well be actions that are not received by all agents and therefore should also not affect the view on all agents.

Example 5.3.2. If agent $\mathcal{D}$ communicates privately with some agent $\mathcal{A}$ it should change its view on $\mathcal{A}$ but not its view on other agents. $\diamondsuit$

This is achieved by use of appropriate interpretation functions that select the relevant information for each agent. A more concrete analysis of change operations depends on the actual formalization of actions and percepts and the resulting interpretation function and their properties. We do this when we consider instantiations of the knowledge representation. This leads to the following observation.

Observation 5.3.3. The satisfaction of *Min-Secrecy-Weakening*$_\circ$ relates to the satisfaction of property (S3) and (P5). The satisfaction of *Secrets Action Invariance*$_{\circ^a}$, *Acknowledgment*$_\circ$, and *Awareness*$_{\circ^a}$ of the change operator of an agent relates to the satisfaction of (P1), (P5) and (S3).

The satisfaction of the properties (P1) to (P5) are not realized by single properties but depend on several aspects of our model, the observations above point out the most important ones. Property (P1) is addressed by the agent-view of the agent in combination with the change operator The meta-information part of property (P1) and property (P3) on meta-inferences can only be concretized in instantiations of the general model that feature definitions of meta-information and meta-inference.

The *Awareness*$_{\circ^a}$ postulate for belief bases is related to the *Success*$_\Phi$ postulate for multiple belief base revision operators, as defined in Section 4.2.1 (Page 77).

Observation 5.3.4. Let $V_W(\mathcal{K})$ and $V_{\mathcal{A}}(\mathcal{K})$ be belief bases, let $\circ^a$ be a change operator, let $*^a_{\mathcal{L}_W}, *^a_{\mathcal{L}_V}$ be multiple belief base change operators, and t^a_W and t^a_V interpretation functions such that Equations 5.3.1 and 5.3.2 are satisfied. If $*^a_{\mathcal{L}_W}, *^a_{\mathcal{L}_V}$ satisfy *Success*$_\Phi$, then $\circ$ satisfies *Awareness*$_{\circ^a}$.

Proof. See Appendix A.1.3, Page 235. □

In the following we consider the construction of a change operator for secrets that leads to the satisfaction of *Min-Secrecy-Weakening*$_\circ$, and that is independent from the used knowledge representation formalism. As specified by the properties *Secrets Action Invariance*$_{\circ^a}$ and *Acknowledgement*$_\circ$ secrets are only changed upon reception of percepts. The property *Min-Secrecy-Weakening*$_\circ$ formalizes that the secrets shall be modified minimally such that secrecy is preserved with respect to the modified set of secrets, i. e., they are weakened. A secret $(\phi, \mathsf{Bel}, \mathcal{A})$ is violated in an epistemic state $\mathcal{K}$ if the formula ϕ is inferred by the belief operator Bel from the agent-view on the attacker, i. e., if $\phi \in \mathsf{Bel}(V_{\mathcal{A}}(\mathcal{K}))$.

We weaken the secret by replacing the belief operator Bel by a maximally credulous belief operator operator Bel' such that the resulting secret is not violated. Only if there is no operator for which the secret is not violated the secret is removed from the set of secrets. We make use of the belief operator family $(\Xi, \preceq_{\mathsf{bel}})$ and the credulity order to determine a different belief operator Bel' by which the secret formula cannot be inferred. The operator Bel' cannot be more credulous than the operator Bel since it would follow that $\mathsf{Bel}(V_{\mathcal{A}}(\mathcal{K})) \subseteq \mathsf{Bel}'(V_{\mathcal{A}}(\mathcal{K}))$ which implies that $\phi \in \mathsf{Bel}'(V_{\mathcal{A}}(\mathcal{K}))$. Hence, we determine a maximally credulous belief operator that is not as credulous as Bel that does not violate the secret. Since this operator is not uniquely defined in general, since $\preceq_{\mathsf{bel}}$ need not be total, one of the maximal ones is selected.

The set of maximal belief operators of a set of belief operators Ξ is given as:

$$\max_{\preceq_{\mathsf{bel}}}(\Xi) =_{def} \{\mathsf{Bel} \in \Xi \mid \text{ there is no } \mathsf{Bel}' \in \Xi \text{ such that } \mathsf{Bel} \prec_{\mathsf{bel}} \mathsf{Bel}'\}.$$

The selection of an arbitrary maximal operator is realized by a selection function σ as introduced in Section 3.5.4 (Page 56). Given these considerations and the definition of $\max_{\preceq_{\mathsf{bel}}}(\Xi)$, we define a function for the selection of a minimally weakened belief operator next.

Let $V \in \mathcal{L}_V$ be an agent-view, $\phi \in \mathcal{L}_{BS}$ be a secret formula and $\mathsf{Bel} \in \Xi$ a belief operator. The operator $\mathsf{maxbel}_{(\Xi, \preceq_{\mathsf{bel}})}$ determines a belief operator from Ξ for the arguments V, ϕ, Bel and is defined as follows:

$$\mathsf{maxbel}_{(\Xi,\preceq_{\mathsf{bel}})}(V,\phi,\mathsf{Bel}) =_{def} \tag{5.3.8}$$

$$\begin{cases} \mathsf{Bel} & \text{if } \phi \notin \mathsf{Bel}(V) \\ \sigma(\max_{\preceq_{\mathsf{bel}}}(\{\mathsf{Bel}' \in \Xi \mid \phi \notin \mathsf{Bel}'(V)\})) & \text{if } \phi \in \mathsf{Bel}(V) \text{ and} \\ & \text{exists } \mathsf{Bel}' \in \Xi \\ & \text{s.t. } \phi \notin \mathsf{Bel}'(V) \\ \perp & \text{if not exists } \mathsf{Bel}' \in \Xi \\ & \text{s.t. } \phi \notin \mathsf{Bel}'(V) \end{cases}$$

If there is no belief operator in Ξ with respect to which ϕ would not be inferred from V the dedicated symbol $\perp$ is returned to indicate this case. Otherwise, the belief operator determined by the $\mathsf{maxbel}_{(\Xi,\preceq_{\mathsf{bel}})}$ function is, if it is different from Bel, guaranteed to be not as credulous as Bel, as we show in the following proposition.

Proposition 5.3.5. Let Ξ be a set of belief operators, V an agent-view and $(\phi,\mathsf{Bel},\mathcal{A})$ a secret. If $\mathsf{Bel}' = \mathsf{maxbel}_{(\Xi,\preceq_{\mathsf{bel}})}(V,\phi,\mathsf{Bel})$, $\mathsf{Bel}' \neq \perp$, then $\mathsf{Bel} \not\prec_{\mathsf{bel}} \mathsf{Bel}'$.

Proof. See Appendix A.1.3, Page 236. □

The change operator for secrets $*_{\mathcal{S}}$ uses the $\mathsf{maxbel}_{(\Xi,\preceq_{\mathsf{bel}})}$ function to determine the new belief operator for each secret and is defined as follows. Let $\mathcal{S} \subseteq \mathcal{L}_{\mathcal{S}}$ be a set of secrets, $V_W \in \mathcal{L}_W$ a world-view and $V \in \mathcal{L}_V$ be an agent-view, then

$$*_{\mathcal{S}}(\mathcal{S},V_W,V) =_{def} \{(\phi,\mathsf{Bel}',\mathcal{A}) \mid (\phi,\mathsf{Bel},\mathcal{A}) \in \mathcal{S}(\mathcal{K}), \; \mathsf{Bel}' = \mathsf{maxbel}_{(\Xi,\preceq_{\mathsf{bel}})}(V,\phi,\mathsf{Bel}), \; \mathsf{Bel}' \neq \perp\} \tag{5.3.9}$$

From Proposition 5.3.5 follows that secrets are only weakened and not strengthened. Strengthening secrets would imply limiting the possibilities of an agent to achieve its other goals more than by the declared secrets, i. e., more than necessary.

In the next definition we show that the specification of the change operator $\circ$ as defined in this section satisfies the properties postulated above.

Proposition 5.3.6. Let $\mathcal{D}$ be an agent, with $\circ$ and $\circ^{a}$ defined as in Equations 5.3.1 to 5.3.6 (Page 115) with $*_{\mathcal{S}}$ as defined in Equation (5.3.9) on Page 118 above and inner revision operators $*_{\mathcal{L}_W}$ and $*_{\mathcal{L}_V}$ satisfying *Success*$_{\Phi}$ (defined in Section 2.5). The change operator $\circ$ satisfies *Acknowledgment*$_{\circ}$, *Min-Secrecy-Weakening*$_{\circ}$ and the change operator for actions $\circ^{a}$ satisfies *Secrets Action Invariance*$_{\circ^{a}}$, *Awareness*$_{\circ^{a}}$.

Proof. See Appendix A.1.3, Page 236. □

In this section we elaborated on the secrecy-relevant aspects of the change operator of an agent and gave a specification of it that is independent of the knowledge representation formalism to be used. In the next section we continue to study secrecy relevant aspects of the general model of epistemic agents.

5.4 CHARACTERIZATION OF SECRECY PRESERVING AGENTS

In this section we characterize secrecy preserving agents by means of properties. To this end we define a notion of *settings* in which we consider our agents. A setting is determined by the set of possible initial agent states and possible percepts. These parameters are chosen to satisfy certain properties for example that the considered change operators satisfy the properties specified in the previous section. A setting also restricts the possible initial agent-views and thereby the initial information available to attackers and their reasoning behavior. We formalize such a setting as follows.

Definition 5.4.1. Let Per be a set of percepts, Act a set of actions and $\mathcal{L}_{\mathrm{ES}}$ a language of epistemic states. A *setting* $\mathcal{T}$ with respect to Per and Act is determined by a tuple $(\Lambda, \mathfrak{F})$ that comprises a set of epistemic states $\Lambda \subseteq \mathcal{L}_{\mathrm{ES}}$, and a set of functional components $\mathfrak{F}$ such that each $(\circ, \circ^{a}, \mathsf{act}) \in \mathfrak{F}$ is of the type:

$$\circ : \mathcal{L}_{\mathrm{ES}} \times \mathsf{Per} \to \mathcal{L}_{\mathrm{ES}}, \circ^{a} : \mathcal{L}_{\mathrm{ES}} \times \mathsf{Act} \to \mathcal{L}_{\mathrm{ES}} \text{ and } \mathsf{act} : \mathcal{L}_{\mathrm{ES}} \to \mathsf{Act}.$$

The setting consists of all initial agent states with an epistemic state from Λ and a functional component from $\mathfrak{F}$. Formally:

$$\mathcal{T} = (\Lambda, \mathfrak{F}) =_{def} \Lambda \times \mathfrak{F}$$

In this section we consider settings with functional components for which all change operators satisfy *Secrets Action Invariance*$_{\circ^a}$ and *Acknowledgement*$_{\circ}$ and for which a sub-setting is secrecy-preserving. A setting $(\Lambda, \mathfrak{F})$ is *secrecy preserving*, if for all $(\mathcal{K}^0, \xi) \in (\Lambda, \mathfrak{F})$ an agent with initial agent state $(\mathcal{K}^0, \xi)$ is secrecy preserving. A setting $(\Lambda, \mathfrak{F})$ is *partially secrecy preserving*, if there is some sub-setting $(\Lambda', \mathfrak{F}') \subseteq (\Lambda, \mathfrak{F})$ that is secrecy preserving. We define properties of settings with the aim to characterize secrecy preserving agents in these.

5.4.1 *Local Properties for Secrecy Preserving Agents*

We start with the consideration of a property that enables agents that only consider their next action without consideration of future courses of action to be secrecy preserving.

Definition 5.4.2. Let $(\Lambda, \mathfrak{F})$ be a setting such that $\Lambda \subseteq \mathcal{L}_{ES}$ and all $(\circ, \circ^{a}, \mathsf{act}) \in \mathfrak{F}$ are of the type $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, $\circ^{a} : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$.

The setting $(\Lambda, \mathfrak{F})$ is *plain*, if for all $(\mathcal{K}, \xi) \in (\Lambda, \mathfrak{F})$ for all safe epistemic states $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}, \mathsf{Per})$ it holds that for all $p \in \mathsf{Per}$ there is some action $a \in \mathsf{Act}$ such that $(\mathcal{K} \circ p) \circ^{a} a$ is safe.

In a *plain setting* it hence holds that, in a safe epistemic state, there always is a secrecy preserving action available, no matter which actions have been performed previously. This implies that the choice of an action that results in a safe successor state does not influence the availability of secrecy preserving actions in the following. We call an action a safe in the context of an epistemic state $\mathcal{K}$ if $\mathcal{K} \circ^{a} a$ is safe. For plain settings we can show that a local selection of *safe actions* of the functional component is sufficient to guarantee secrecy preservation.

Proposition 5.4.3. Let $(\Lambda, \mathfrak{F})$ be a setting such that $\Lambda \subseteq \mathcal{L}_{ES}$ and all $(\circ, \circ^{a}, \mathsf{act}) \in \mathfrak{F}$ are of the type $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, $\circ^{a} : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$.

If $(\Lambda, \mathfrak{F})$ is plain, then any agent with an initial agent state $(\mathcal{K}^0, \xi) \in (\Lambda, \mathfrak{F})$ that has a safe initial epistemic state $\mathcal{K}^0$ and that selects a safe action if possible, i. e., for all $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}^0, \mathsf{Per})$ and all $p \in \mathsf{Per}$ if there is an $a \in \mathsf{Act}$ such that $(\mathcal{K} \circ p) \circ^{a} a$ is safe, then $\mathcal{K} \circ p \circ^{a} \mathsf{act}(\mathcal{K} \circ p)$ is safe, is secrecy preserving.

Proof. See Appendix A.1.3, Page 237. □

If it is not possible for the agent to select a safe action, it cannot preserve secrecy. In this case it should choose the action that violates secrecy as little as possible. We show how this is can be done in Section 5.5.2.

The requirement of a plain setting seems to be a very strong one since it requires the existence of a safe action in any epistemic state of the agent, no matter what the agent did previously. It is not as strong as it seems since in many environments this is satisfied, for instance in environments in which the empty action does not have any secrecy violating effects. We come back to this later. In the next section we consider secrecy preserving agents in settings that are not plain.

5.4.2 *Look ahead and Secrecy Preserving Agents*

In general agent $\mathcal{D}$ has to be able to avoid to maneuver itself into states in which it is left without a secrecy preserving action to choose. To this end $\mathcal{D}$ has to take possible future actions of itself and other agents into consideration when deciding on its next action. To formalize this, we formulate a game theoretic view on our setting on the

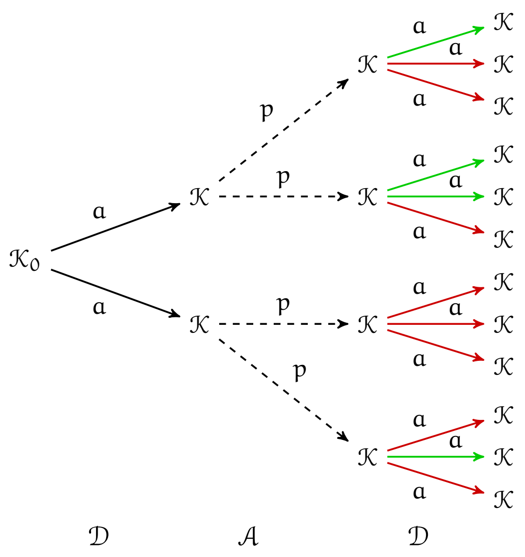


Figure 5.4.1: Sketch of a game tree

basis concepts from game theory [187]. We define a two-player game with a defending agent $\mathcal{D}$ and an attacking agent $\mathcal{A}$. Perceptions of the defending agent are assumed to be the actions of the attacking agent.

Definition 5.4.4. Let $\mathcal{L}_{ES}$ be a language of epistemic states, $\mathcal{K}_0 \in \mathcal{L}_{ES}$ an epistemic state, Per a set of percepts, Act a set of actions, and $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$ and $\circ^a : \mathcal{L}_{ES} \times \mathsf{Act}$ change operators. We define a *two-player game* $g_{\mathcal{L}_{ES},\mathcal{K}_0,\mathsf{Per},\mathsf{Act},\circ,\circ^a}$ with a set of agents $Ag_2 = \{\mathcal{D}, \mathcal{A}\}$, a game tree T, a player function P and a set of *utility functions* $U = \{u_\mathcal{D}, u_\mathcal{A}\}$. The *game tree* $T = (\mathcal{L}_{ES}, E, \mathcal{K}_0)$ comprises a language of epistemic states $\mathcal{L}_{ES}$, edges $E \subseteq \mathcal{L}_{ES} \times \mathcal{L}_{ES}$ and root $\mathcal{K}_0 \in \mathcal{L}_{ES}$. The parent of a node $\mathcal{K}$ is given by $pa(\mathcal{K})$. The player function $P : \mathcal{L}_{ES} \to Ag_2$ is defined as

$$P(\mathcal{K}_0) = \mathcal{A} \text{ and } P(\mathcal{K}) = \begin{cases} \mathcal{A} & \text{if } P(pa(\mathcal{K})) = \mathcal{D} \\ \mathcal{D} & \text{if } P(pa(\mathcal{K})) = \mathcal{A} \end{cases}. \qquad (5.4.1)$$

The tuple $(\mathcal{K}, \mathcal{K}_{i-1})$ is an element of E if and only if $\mathcal{K}_{i-1} = \mathcal{K} \circ p$ with $p \in \mathsf{Per}$ and $P(\mathcal{K}) = \mathcal{A}$, or $\mathcal{K}_{i-1} = \mathcal{K} \circ^a a$ with $a \in \mathsf{Act}$ and $P(\mathcal{K}) = \mathcal{D}$.

Example 5.4.5. Figure 5.4.1 sketches a game tree with the root node $\mathcal{K}_0$ and the possible actions a of agent $\mathcal{D}$ and the possible actions p of agent $\mathcal{D}$. The green edges indicate safe actions and the red edges actions that violate secrecy (only marked on the last level). ♢

Based on this formulation of a game we introduce the notion of a strategy of an agent and in particular the notion of a winning strategy of the attacking agent.

A *path* π is a sequence of nodes

$$\pi = (\mathcal{K}_0, \dots, \mathcal{K})$$

such that $(\mathcal{K}_i, \mathcal{K}_{i+1}) \in E$ for all $i \in [0, k-1]$. The set of paths is denoted by W and the sequence of the first i, $i \leqslant k$, elements of a sequence π is denoted by $\pi[i]$. The *utility functions* $\{u_\mathcal{D}, u_\mathcal{A}\} = U$ assign 0 or 1 to each path, formally $u_\mathcal{X} : W \to \{0, 1\}$ with $u_\mathcal{D}(\pi) = 0$ if there exists $\mathcal{K} \in \pi$ which is not safe, else $u_\mathcal{D}(\pi) = 1$. It is $u_\mathcal{A}(\pi) = 1 - u_\mathcal{D}(\pi)$. That is, $\mathcal{D}$ gets a utility of 1 if it preserves secrecy with respect to π and a utility of 0 if not. For agent $\mathcal{A}$ it is the other way round.

A *strategy* for player $\mathcal{X}$ is a partial function $s_\mathcal{X} : W \to \Sigma$ which assigns to each finite path $\pi = (\mathcal{K}_0, \dots, \mathcal{K}_n)$ with $P(\mathcal{K}_n) = \mathcal{X}$, a percept p if $\mathcal{X} = \mathcal{A}$ and $p \in \{p' \mid (\mathcal{K}_n, \mathcal{K}_n \circ p') \in E\}$, and an action a if $\mathcal{X} = \mathcal{D}$ and $a \in \{a' \mid (\mathcal{K}_n, \mathcal{K}_n \circ^a a') \in E\}$.

A path $\pi = (\mathcal{K}_0, \dots, \mathcal{K}_n) \in W$ is called *compliant* with strategy $s_\mathcal{X}$ if for all $\mathcal{K}_i \in \pi$ with $0 \leqslant i < n$ with $P(\mathcal{K}_i) = \mathcal{X}$ it is the case that, if $\mathcal{X} = \mathcal{D}$: $\mathcal{K}_{i+1} = \mathcal{K}_i \circ^a s_\mathcal{D}(\pi)$, and if $\mathcal{X} = \mathcal{A}$: $\mathcal{K}_{i+1} = \mathcal{K}_i \circ s_\mathcal{A}(\pi)$. A strategy $s_\mathcal{X}$ for player $\mathcal{X}$ is called a *winning strategy* in state $\mathcal{K}$ if for all paths $\pi = (\mathcal{K}_0, \dots, \mathcal{K}_n)$ with $\mathcal{K}_0 = \mathcal{K}$ that are compliant with $s_\mathcal{X}$ the utility of $\mathcal{X}$ is $u_\mathcal{X}(\pi) = 1$. That is, $\mathcal{A}$ has a winning strategy in state $\mathcal{K}$ if it can, by means of a sequence of percepts for $\mathcal{D}$, force $\mathcal{D}$ into an unsafe state no matter what $\mathcal{D}$ does.

Example 5.4.6. In the game sketched in Figure 5.4.1 agent $\mathcal{D}$ should not choose the lower action in $\mathcal{K}_0$ since $\mathcal{A}$ has a winning strategy in the resulting state. The is, in that state $\mathcal{A}$ can choose the upper p which results in a state in which $\mathcal{D}$ is left with only secrecy violating actions (only red outgoing edges). ◇

Definition 5.4.7. Let $\mathcal{L}_{ES}$ be a language of epistemic states, Per a set of percepts, Act a set of actions, and $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, and $\circ^a : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ change operators.

An epistemic state $\mathcal{K}$ is *sound* with respect to $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$ if $\mathcal{A}$ does not have a winning strategy in $g_{\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a}$.

To ease readability, we omit the context of a sound state, i. e.,'wrt. $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$', in the following, if the context is obvious. We assume in the following that initial epistemic states are sound.

The notion of a sound state generalizes the notion of a safe epistemic state by demanding that $\mathcal{D}$ should have a possibility to maintain a safe epistemic state, as the following lemma shows.

Lemma 5.4.8. Let $\mathcal{K} \in \mathcal{L}_{ES}$ be an epistemic state, Per a set of percepts, Act a set of actions, and $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$ a change operator. If $\mathcal{K}$ is sound with respect to $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^{a})$, then it is safe.

Proof. See Appendix A.1.3, Page 237. □

We can show that an agent that avoids unsound states is not only secrecy preserving but that an agent has to avoid unsound states to be secrecy preserving. That is, maintaining a sound state is not only sufficient, it is necessary for secrecy preservation.

Proposition 5.4.9. Let $(\Lambda, \mathfrak{F})$ be a setting for which $\Lambda \subseteq \mathcal{L}_{ES}$ and all $(\circ, \circ^{a}, \mathsf{act}) \in \mathfrak{F}$ are of the type $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, $\circ^{a} : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$. An agent with initial agent state $(\mathcal{K}^{0}, \xi) \in (\Lambda, \mathfrak{F})$ is secrecy preserving if and only if all states $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}^{0}, \mathsf{Per})$ are sound.

Proof. See Appendix A.1.3, Page 238. □

For *plain* settings we required that there is a secrecy preserving option in any situation. Hence we can show that if the setting under consideration is plain, then any safe state is sound.

Proposition 5.4.10. Let $(\Lambda, \mathfrak{F})$ be a setting that is plain and let $(\mathcal{K}^{0}, \xi) \in (\Lambda, \mathfrak{F})$ be an agent state of this setting. Any epistemic state $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}^{0}, \mathsf{Per})$ that is safe, is also sound.

Proof. See Appendix A.1.3, Page 238. □

The following proposition shows that in general an agent, that always selects an action that leads to a sound state is secrecy preserving.

Proposition 5.4.11. Let $(\mathcal{K}^{0}, \xi)$ be the initial agent state of agent $\mathcal{X}$ with $\xi = (\circ, \circ^{a}, \mathsf{act})$, $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, $\circ^{a} : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$. If $\mathcal{K}^{0}$ is sound, then if $\mathcal{X}$ selects sound actions, i. e., for all $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}^{0}, \mathsf{Per})$ and all $p \in \mathsf{Per}$ if there is some $a \in \mathsf{Act}$ such that $(\mathcal{K} \circ p) \circ^{a} a$ is sound, then $\mathsf{act}(\mathcal{K} \circ p)$ is such that $\mathcal{K} \circ^{a} \mathsf{act}(\mathcal{K} \circ p)$ is sound, then $\mathcal{X}$ is secrecy preserving.

Proof. See Appendix A.1.3, Page 239. □

Proposition 5.4.11 shows that a secrecy preserving agent has to be able to check whether an action it considers to execute leads to a sound state. This is equivalent to checking whether $\mathcal{A}$ has a winning strategy, which is known to be a hard problem in general [210]. Hence, it is useful to determine cases in which the existence of a winning strategy can be excluded easier. We do this in the following section.

5.4.3 *Reducing Look-ahead to Local Properties*

In this section we define properties of a setting whose satisfaction imply that the setting is *plain*. A simple, yet important, property can be expressed informally as *not doing anything does not do any harm*. We formalize *not doing anything* by the empty action ϵ and require that it does not violate secrecy in all possible epistemic states.

Observation 5.4.12. Let $(\Lambda, \mathfrak{F})$ be a setting with $\xi = (\circ, \circ^a, \mathsf{act})$, $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, $\circ^a : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$. Let $\epsilon \in \mathsf{Act}$ be the empty action. If for all $\mathcal{K} \in \Omega_{\xi}s(\Lambda, \mathsf{Per})$ and all $p \in \mathsf{Per}$ it holds that $(\mathcal{K} \circ p) \circ^a \epsilon$ is safe, then $(\Lambda, \mathfrak{F})$ is plain.

Proof. See Appendix A.1.3, Page 239. □

There are scenarios in which the empty action is either not available for the defending agent by design, or the empty action does have effects on the preservation of secrecy. The latter is the case if an agent is expected to react in a certain way by another agent, i. e., to follow a protocol. In these cases not reacting might reveal information which leads to the disclosure of a secret formula. In these cases secrecy is violated due to assumed meta-inferences of $\mathcal{A}$. In the following we consider both scenarios for communicating agents.

5.4.3.1 *Query-answer Protocol*

We start with a simple query-answer Protocol in which in each round $\mathcal{A}$ is posing a question to $\mathcal{D}$, who then responds with an answer to it. $\mathcal{D}$'s only available actions are to answer to each question with one of the defined answer values. From the point of view of agent theory this scenario seems to be very restricted. The defending agent is only reacting to answers posed to it. However, from the point of secrecy, this is already a complex task. It is the typical protocol in database theory in which secrecy has been studied intensively, see, e. g., [46, 38, 32].

For our consideration of $\mathcal{D}$ as an epistemic agent this scenario is modeled by restricting the possible percepts and actions of $\mathcal{D}$. The set of possible percepts of $\mathcal{D}$ is:

$$\mathsf{Per}^{QA} = \{\langle \mathcal{A}, \mathcal{D}, \mathsf{query}, \phi \rangle \mid \phi \in \mathcal{L}_{\mathsf{Base}}\}.$$

And the set of possible actions of $\mathcal{D}$ is:

$$\mathsf{Act}^{QA} = \{\langle \mathcal{D}, \mathcal{A}, \mathsf{answer}, \phi \rangle \mid \phi \in \mathcal{L}_{\mathsf{Base}}\}.$$

The used protocol on the basis of these actions and percepts is that a requesting speech act $\langle \mathcal{A}, \mathcal{D}, \mathsf{query}, \phi \rangle$ requests an answer of the form $\langle \mathcal{D}, \mathcal{A}, \mathsf{answer}, \phi' \rangle$ with ϕ' representing an evaluation of ϕ. We also indicate that a formula represents a formula with undetermined truth value by $\hat{\phi}$. Formally it is $\phi' \in \mathsf{evals}(\hat{\phi})$, whereby $\mathsf{evals}(\hat{\phi})$ is to be specified. In the following we consider $\mathsf{evals}_{\mathsf{bool}}(\hat{\phi}) = \{\phi, \neg\phi\}$, representing the two truth values *true* and *false*. These possible answers correspond to the setting of a *complete* database [46]. For incomplete databases a third answer value exists which expresses undecidedness with respect to the queried formula and is expressed by the value *unkown*.

From the game theoretic perspective, this setting gives the attacker $\mathcal{A}$ the possibility to ask a sequence of queries and $\mathcal{D}$ has to answer with an evaluation of the respective formula. If there is no combination of answers to a sequence of queries such that the resulting epistemic state of $\mathcal{D}$ is safe, then $\mathcal{A}$ has a winning strategy, and $\mathcal{D}$ is in a hopeless situation. A secrecy preserving agent $\mathcal{D}$ has to make sure that none of its actions leads to this situation.

In the database setting an important result with respect to the inference problem is the following. It has been shown that, roughly speaking, if the attacker knows which are the potentially secret formulae of the defender, and that the disjunction of all of these holds, and the defender is only allowed to answer *yes* or *no* to a question, then the defender is in a hopeless situation [46]. More precisely, it has been shown that the defender is in a hopeless situation if and only if these conditions hold. Intuitively this is the case since if the attacker knows the potential secrets of the defender and if it knows that the disjunction of all these secret formulae holds, then it can query all secret formulae in a sequence. The defending agent has to reply *yes* or *no* and is not allowed to violate consistency, which is impossible without violating secrecy if the disjunction of all secrets is known to hold by the attacker. Consequently, to protect secrecy, the disjunction of secrets has to be protected. If we transfer the restrictions of the database setup to our model, we can show that the same result holds for the resulting setting. To model the query-answer protocol as it is given in the database scenario [46] we define a setting on the basis of a set of restrictions to the epistemic states and functional components.

Definition 5.4.13. We define the *query-answer setting* $\mathcal{T}^{QA}$ as the setting that comprises secrecy agent states and corresponding percepts, actions and belief operators that satisfy the following conditions:

1. All languages are instantiated by the standard propositional language: $\mathcal{L}_W = \mathcal{L}_V = \mathcal{L}_{BS} = \mathcal{L}_{\mathsf{Base}} = \mathcal{L}^{\mathsf{prop}}_{At}$
2. The belief operator family $(\Xi, \preceq_{\mathsf{bel}})$ consists only of the propositional consequence operator:
$$\Xi = \{Cn^{\mathsf{prop}}\} \text{ and } \preceq_{\mathsf{bel}} = (Cn^{\mathsf{prop}}, Cn^{\mathsf{prop}})$$
3. The interpretation function for the view on $\mathcal{A}$ is such that

a) queries do not have any effect: For all $\langle \mathcal{A}, \mathcal{D}, \mathsf{query}, \phi \rangle \in \mathsf{Per}^{QA}$ it is $t_V(\langle \mathcal{A}, \mathcal{D}, \mathsf{query}, \phi \rangle) = \emptyset$

b) answers are assumed to carry only their logical content as information: For all $\langle \mathcal{D}, \mathcal{A}, \mathsf{answer}, \phi \rangle \in \mathsf{Act}^{QA}$ it is $t_V(\langle \mathcal{D}, \mathcal{A}, \mathsf{answer}, \phi \rangle) = \{\phi\}$.

4. The inner revision operator $*_{\mathcal{L}_V}$ is a propositional belief base change operator that satisfies *Inclusion* and *Vacuity*, as defined in Section 2.5.

5. $\mathcal{D}$ only answers queries for a formula ϕ with either ϕ or $\neg\phi$: For all $\mathcal{K} \in \mathcal{L}_{ES}$ and $\langle \mathcal{D}, \mathcal{A}, \mathsf{query}, \phi \rangle \in \mathsf{Per}^{QA}$ it holds that

$$\mathsf{act}(\mathcal{K} \circ \langle \mathcal{A}, \mathcal{D}, \mathsf{query}, \phi \rangle) \in \{\langle \mathcal{D}, \mathcal{A}, \mathsf{answer}, \phi \rangle, \langle \mathcal{D}, \mathcal{A}, \mathsf{answer}, \neg\phi \rangle\}$$

6. $\mathcal{D}$ does not give answers that lead to the violation of consistency of the view on $\mathcal{A}$:
For all $\mathcal{K} \in \mathcal{L}_{ES}$ and all $p \in \mathsf{Per}^{QA}$ it holds that

$$V_{\mathcal{A}}(\mathcal{K} \circ^{a} \mathsf{act}(\mathcal{K} \circ p)) \text{ is consistent.}$$

In the database setting it is usually the case that secrets are static. Here this follows for our considered setting from the fact that the queries of the attacker do not lead to the violation of secrets of the defender. This is shown by the following lemma.

Lemma 5.4.14. For each initial agent state $(\mathcal{K}^0, \xi) \in \mathcal{T}^{QA}$ it holds for all $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}^0, \mathsf{Per}^{QA})$ that $\mathcal{S}(\mathcal{K}) = \mathcal{S}(\mathcal{K}^0)$.

Proof. See Appendix A.1.3, Page 240. □

For the setting $\mathcal{T}^{QA}$ we can now show that $\mathcal{A}$ has a winning strategy exactly if it believes the disjunction of all secret formulae. We use the following notation for the *set of secret formulae*:

$$F(\mathcal{S}(\mathcal{K})) =_{def} \{\phi \mid (\phi, \mathsf{Bel}, \mathcal{A}) \in \mathcal{S}(\mathcal{K})\}.$$

Remember that we assumed all secrets to be active.

Proposition 5.4.15. For each initial agent state $(\mathcal{K}^0, \xi) \in \mathcal{T}^{QA}$ it holds for all $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}^0, \mathsf{Per}^{QA})$ that $\mathcal{K}$ is not sound wrt. $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}^{QA}, \mathsf{Act}^{QA}, \circ, \circ^{a})$ if and only if

$$\Big(\bigvee_{\phi \in F(\mathcal{S}(\mathcal{K}))} \phi\Big) \in \mathsf{Bel}(V_{\mathcal{A}}(\mathcal{K})).$$

Proof. See Appendix A.1.3, Page 240. □

This result is very important for the defined scenario since it reduces the computation of winning strategies of attackers to the evaluation of a logical formula.

Example 5.4.16. Suppose that *Beatriz* knows that if *Emma* wants to take a day off, then she wants to either attend a *strike committee meeting* or she has a job interview with another company. Both cases are secrets of *Emma*. Further suppose that *Beatriz* asks *Emma* if she wants to take a day off, and *Emma* is only allowed to answer with *yes* or *no* to questions.

If *Emma* answers *yes*, then she cannot answer to the questions if she intends to attend a *strike committee meeting* and if she intends to go to a job interview without violating a secret of her, or by contradicting the beliefs of *Beatriz*. Therefore, the only way to preserve secrecy for *Emma* is to answer with *no* to the first question; hereby avoiding that *Beatriz* beliefs in the disjunction of the two secret formulae. This way, however, *Emma* cannot satisfy her goal of getting the day of the *strike committee meeting* off. It is clear, that she has to violate one of her secrets in order to satisfy her goal. ♢

Proposition 5.4.15 also implies that if we demand that agents have to protect the disjunction of their secret formulae by making it a secret formula itself, then we obtain a plain setting.

Corollary 5.4.17. Let $\mathcal{T}^{QA}_{\vee} \subseteq \mathcal{T}^{QA}$ be the setting for which it holds that for all $(\mathcal{K}^0, \xi) \in \mathcal{T}^{QA}_{\vee}$ there is some $\phi \in F(\mathcal{S}(\mathcal{K}^0))$ such that $\phi \equiv^p \bigvee_{\phi \in F(\mathcal{S}(\mathcal{K}^0))} \phi$. The setting $\mathcal{T}^{QA}_{\vee}$ is *plain*.

Proof. See Appendix A.1.3, Page 243. □

The result formulated in the corollary is important in the restricted scenario of a query-answer protocol with only boolean answer values. However, it does not hold if the restrictions of this scenario are only slightly weakened. For the scenario in which a third answer value *unknown* is allowed protecting the disjunction of secrets is still sufficient, but not necessary [46]. For more realistic scenarios of autonomous agents the restrictions of this scenario are way too strong such that the result cannot be applied. In the following we consider the problem of preservation of secrets from the other side. That is, we do not consider what is necessary to protect a secret, but what is necessary to infer a secret for an attacking agent. We describe that if the empty action is an option for $\mathcal{D}$ than, in order to infer a secret, $\mathcal{A}$ has to make use of meta-inferences.

5.4.3.2 *Meta-Inferences and Communication*

Meta-inferences are inferences that are not based on explicit informational content of an action but on the context of its execution and information about the context. A very good example is the empty action, because it does has no explicit informational content and does not directly influence anything but still its meta-information can be used by an attacker to draw meta-inferences.

Example 5.4.18. Consider the situation in which *Emma* has asked for a day off and is now asked by *Beatriz* if she intends to attend the *strike committee meeting*. Further, *Beatriz* knows that *Emma* wants to keep secret that she intends to attend the *strike committee meeting*, if she does intend to attend. If *Emma* now does not say anything, then *Beatriz* infers from *Emma*'s behavior that the answer is *yes*. That is, *Beatriz* drew a meta-inference from the behavior of *Emma*. ♢

In our model the meta-information carried by actions of the defending agent are interpreted by the interpretation function and represented in $\mathcal{D}$'s view of $\mathcal{A}$.

Example 5.4.19. *Beatriz* has to recognize that *Emma* does not answer her question if she intends to attend the *strike committee meeting*. This relates to the meta-information of the absence of an answer. Further *Beatriz* has background knowledge allowing her to draw the conclusion that *Emma* intends to attend the *strike committee meeting*. *Emma* has to be aware of this knowledge and reasoning of *Beatriz* in order to preserve secrecy. ♢

In Section 5.1 we formulated these aspects by means of the principles P2 (simulation) and P3 (meta-inferences). It is essential that the modeling of the background information, reasoning methods and capabilities of the attacking agent meet reality as accurately as possible. Both over- and underestimation of the capabilities of an attacker is undesirable as the following example illustrates.

Example 5.4.20. Consider again the situation of Example 5.4.18. *Emma* has to estimate the reasoning behavior of *Beatriz*. Let *Emma* assume that *Beatriz* knows that she does not reply to any question that is related to sensitive information. That is, any question for which one of the possible answers would violate secrecy is not replied to. For example, *Beatriz* knows that if she asks whether *Emma* intend to attend the *strike committee meeting*, she would get not answer, independent of the fact whether *Emma* intends to attend or not. Under this assumption *Beatriz* has no reason to infer that *Emma* intends to attend the *strike committee meeting*. If this assumption of *Emma* is wrong, then *Beatriz* will infer that *Emma* attends the *strike committee meeting* from *Emma*'s refusal to answer her question. Then *Emma* overestimated the reasoning of *Beatriz*.

If *Emma* answers the question by switching the topic to, e. g., the weather, assuming that *Beatriz* will not notice her refusal to answer the question, she might be underestimating the reasoning of *Beatriz*, and possibly notice later that her secret is violated. ♢

We consider the meta-inference as they are handled in the database setting. In the database setting meta-inferences are treated in a slightly extended version of the *query-answer protocol* in which $\mathcal{D}$ is also allowed to answer refuse [38]. We assume that refuse is treated as a proposition of the used language, i. e., $\mathsf{refuse} \in \mathcal{L}^{\mathsf{prop}}$, and modify the assumption 5. in Section 5.4.18

5.′ $\mathcal{D}$ only answers queries for a formula ϕ with either ϕ, $\neg\phi$ or refuse: For all $\mathcal{K} \in \mathcal{L}_{\mathsf{ES}}$ and $\langle \mathcal{D}, \mathcal{A}, \mathsf{query}, \phi \rangle \in \mathsf{Per}$ it holds that

$$\begin{aligned} \mathsf{act}(\mathcal{K} \circ \langle \mathcal{A}, \ \mathcal{D}, \mathsf{query}, \phi \rangle) \in \\ & \{\langle \mathcal{D}, \mathcal{A}, \mathsf{answer}, \phi \rangle, \\ & \langle \mathcal{D}, \mathcal{A}, \mathsf{answer}, \neg\phi \rangle, \\ & \langle \mathcal{D}, \mathcal{A}, \mathsf{answer}, \mathsf{refuse} \rangle\}. \end{aligned}$$

It is then assumed that $\mathcal{D}$ answers refuse whenever any other answer would violate secrecy. That is, it satisfies: If there is $a \in \mathsf{Act}^{QA} \setminus \langle \mathcal{D}, \mathcal{A}, \mathsf{answer}, \mathsf{refuse} \rangle$ such that $\mathcal{K} \circ \langle \mathcal{A}, \mathcal{D}, \mathsf{query}, \phi \rangle \circ^{a} a)$ is not safe, then

$$\mathsf{act}(\mathcal{K} \circ \langle \mathcal{A}, \mathcal{D}, \mathsf{query}, \phi \rangle) = \langle \mathcal{D}, \mathcal{A}, \mathsf{answer}, \mathsf{refuse} \rangle.$$

It is also assumed that $\mathcal{A}$ knows the exact behavior of $\mathcal{D}$. It is argued that $\mathcal{A}$ cannot perform any meta-inference from a refusal since it knows that $\mathcal{D}$ would also refuse to answer if answering truthfully would not violate secrecy. The resulting setting is thus plain, even if the disjunction of the secret formulae is not protected. We illustrate this idea in our running example.

Example 5.4.21. Consider again that *Emma* is asked by *Beatriz* if she intends to attend the *strike committee meeting* and *Emma* does not answer this question. Then we assume that *Beatriz* knows that *Emma* does not answer any question for which there exists an answer that might violate secrecy. Given this knowledge, she does not draw any meta-inference and does not belief that *Emma* is interested in attending the *strike committee meeting*. ♢

The assumption that $\mathcal{A}$ knows the potential secrets of $\mathcal{D}$ and knows that $\mathcal{D}$ does not answer any sensitive queries is not very realistic. For more general scenarios and especially without the assumption that the attacking agent has complete knowledge about the secrets and the decision making of the defending agent the modeling of the attacker has to be specific for a given scenario and knowledge representation formalism. Before we consider more specific instantiations of our model we consider general preference relations on options for action with respect to secrecy.

5.5 PREFERENCES ON OPTIONS FOR ACTION AND SECRECY

In the last sections we considered the existence of actions that do not violate secrecy, and we formalized that an agent that performs such actions preserves secrecy. We did not consider how an agent can determine and select such actions. That is, what we do in this section. More generally speaking, we study the deliberation problem in the light of secrecy.

The agent has to decide on its course of action by deciding among different options. Options result from alternative goals, subgoals, and, ultimately, actions. We abstract from these differences and generally speak about options, and we consider in particular atomic intentions, that directly correspond to actions, or actions as options. We assume that an agent chooses its options in accordance with a preference relation on (a subset of) all possible options. For instance as in the BDI^+ model we introduced in Section 3.5.4. We assume that our agents always consider the empty action as an option and, if answering a query, they always consider the two possible answer values as options to answer the query. These are minimal requirements of the set of options, an agent generally considers additional options as well.

The main question here is then which properties such a preference relation has to satisfy in order to guarantee preservation of secrecy. This is sufficient to define secrecy preserving agents, but as motivated before our secrecy aware agents shall be able to distinguish different levels of violation of secrecy as good as possible, given the information available to them. To this end we formulate a set of principles that shall be satisfied by a preference relation of a secrecy aware agent. Moreover, we define a constructive approach to generate a preference relation that satisfies all principles and we develop an algorithm for it.

5.5.1 *General Model for Preferences on Actions*

To consider the aspects of the evaluation with respect to secrecy separately from other aspects we employ a modular approach to the evaluation of options. The preferences on actions for secrecy preservation and those from other aspects that lead to preferences on actions, such as the current intentions of the agent, are then aggregated to form the overall preference relation of the agent. We assume each criterion to be independent from the others and to be reflected by a separate preference relation $\preceq^{i}_{(\mathcal{K},\xi)}$ that is induced by the current agent state $(\mathcal{K},\xi)$, for instance by means of a utility function [22]. One of these criteria is the violation of secrecy and is represented by the preference relation $\preceq^{s}_{(\mathcal{K},\xi)}$. These preferences are combined by an aggregation function to give a combined preference relation on options $\preceq_{(\mathcal{K},\xi)}$. For simplicity of presentation we consider preference relations on actions in the following. These can be equally formalized for intentions. For atomic intentions AtInt the preference relation on actions can be directly used by considering $\alpha(I) \preceq_{(\mathcal{K},\xi)} \alpha(I')$, with $I, I' \in \mathsf{AtInt}$; since $\alpha(I)$ uniquely determines the action for the atomic intention. We define the overall preference relation $\preceq_{(\mathcal{K},\xi)} \subseteq \mathsf{Act} \times \mathsf{Act}$ of the agent in its current epistemic state $\mathcal{K}$ to be the result of the aggregation of different preference relations by some aggregation function f_{agg}, such that

$$\preceq_{(\mathcal{K},\xi)} =_{def} f_{agg}((\preceq^1_{(\mathcal{K},\xi)}, \ldots, \preceq^n_{(\mathcal{K},\xi)})). \tag{5.5.1}$$

The aggregation of preference relations is studied mostly in the field of social choice and voting e. g., in [61, 239, 54, 10, 226]. In difference to the aggregations studied in social choice, here we consider the aggregation of preferences over actions with respect to different criteria within one agent and not the aggregation of preferences of many agents. Hence we do not go into details on the general issues of preference aggregation and assume that a clear prioritization of the preference relations is given, cf. [249, 250], such that we can use a lexicographic aggregation function. For our means, we define a *lexicographical aggregation function* f^{lex}_{agg} as:

$$\begin{aligned}
&f^{\mathsf{lex}}_{agg}((\preceq^1_{(\mathcal{K},\xi)}, \ldots, \preceq^n_{(\mathcal{K},\xi)})) = \\
&\quad \{(a, a') \mid \text{there exists } i, 1 \leqslant i \leqslant n \text{ such that} \\
&\qquad \text{for all } j, 1 \leqslant j < i: \\
&\qquad\quad (a =^j_{(\mathcal{K},\xi)} a' \text{ or } (a \not\succeq^j_{(\mathcal{K},\xi)} a' \text{ and } a' \not\succeq^j_{(\mathcal{K},\xi)} a)) \\
&\qquad\quad \text{and } a \preceq^i_{(\mathcal{K},\xi)} a' \\
&\quad \}
\end{aligned} \tag{5.5.2}$$

This aggregation function orders the actions giving priority to the preference relations according to their order. It prefers an action a over an action a' if the former is preferred over the latter by one of the relations $\preceq^i_{(\mathcal{K},\xi)}$ and none of the higher prioritized relations $\preceq^j_{(\mathcal{K},\xi)}$ with $j < i$ strictly prefers a' over a.

We are especially interested in properties of the $\preceq^s_{(\mathcal{K},\xi)}$ relation that represents the preferences on options with respect to secrecy, and how these carry over to the $\preceq_{(\mathcal{K},\xi)}$ relation by means of the construction given in (5.5.1) with the lexicographic aggregation function as given in (5.5.2). Given the results from the previous section (Definition 5.4.7, Page 122) we know that, in order to guarantee strict secrecy preservation of an agent $\preceq_{(\mathcal{K},\xi)}$ has to prefer actions that maintain a sound epistemic state. We formalize this requirement in the following property:

SOUND PREFERENCE For all $p \in \mathsf{Per}$ and all $a, a' \in \mathsf{Act}$, if it holds that $\mathcal{K} \circ p \circ^a a$ is sound and $\mathcal{K} \circ p \circ^a a'$ is not sound, then $a' \preceq_{(\mathcal{K} \circ p,\xi)} a$. (5.5.3)

For plain settings it is sufficient to maintain a safe epistemic state (Definition 5.2.5, Page 111), which translates to the following property of $\preceq_{(\mathcal{K},\xi)}$.

SAFE PREFERENCE For all $p \in \mathsf{Per}$ and all $a, a' \in \mathsf{Act}$, if $\mathcal{K} \circ p \circ^a a$ is safe and $\mathcal{K} \circ p \circ^a a'$ is not safe, then $a' \preceq_{(\mathcal{K} \circ p,\xi)} a$. (5.5.4)

If the $\preceq^{s}_{(\mathcal{K},\xi)}$ relation satisfies *sound preference*, then $\preceq_{(\mathcal{K},\xi)}$ inherits the property if the aggregation function has the property that

$$\text{if } a' \prec^{s}_{(\mathcal{K},\xi)} a \text{ then for } \preceq_{(\mathcal{K},\xi)} = f_{agg}((\preceq^{s}_{(\mathcal{K},\xi)}, \dots)) \\ \text{it holds that } a' \prec_{(\mathcal{K},\xi)} a.$$

The lexicographical aggregation function f^{lex}_{agg} apparently satisfied this property. Note that, as follows from Proposition 5.4.10 (Page 123), for plain settings any safe action is a sound action. In the following we focus on plain settings, if not stated otherwise.

Apart from the secrecy preference relation we introduce and define another preference relation as a representative for preferences originating from other criteria than secrecy. It gives preference to actions that are more informative for the receiver. That is, we model a cooperative agent that, besides protecting its own secrets aims to be as *informative* for other agents as possible. For an overview and discussion of informativity and utility see, e. g., [235]. Our use of an informativity relation in the secrecy context is similar to the concept of maximizing the *availability* of information under the preservation of secrecy, as considered in works on secrecy in information systems, see e. g., [32]. In [209] different types of lies are considered and preferences defined for these.

For our means we define the following preferences on our considered speech acts with respect to *informativity*. We consider truthful information most informative and untruthful information contra-informative. Further we consider the empty action to be neutral with respect to informativity, and querying as more informative than the empty action. These considerations lead to the following definition of $\preceq^{a}_{\mathcal{K}}$. Let $\mathcal{K}$ be an epistemic state, $\mathsf{Bel}_{\mathcal{X}}$ the belief operator of agent $\mathcal{X}$, and $\mathsf{Act}_{\mathcal{X}}$ the agent's possible actions, $\preceq^{a}_{\mathcal{K}}$ is the relation that satisfies the following properties:

1. for all $\langle \mathcal{X}, \mathcal{X}_r, \mathsf{type}, \phi \rangle \in \mathsf{Act}_{\mathcal{X}}$ with $\mathsf{type} \in \mathsf{types}_I$: (5.5.5)
   $\epsilon \preceq^{a}_{\mathcal{K}} \langle \mathcal{X}, \mathcal{X}_r, \mathsf{type}, \phi \rangle$
   if $\phi \in \mathsf{Bel}_{\mathcal{X}}(V_W(\mathcal{K}))$

2. for all $\langle \mathcal{X}, \mathcal{X}_r, \mathsf{type}, \phi \rangle \in \mathsf{Act}_{\mathcal{X}}$ with $\mathsf{type} \in \mathsf{types}_I$:
   $\langle \mathcal{X}, \mathcal{X}_r, \mathsf{type}, \phi' \rangle \preceq^{a}_{\mathcal{K}} \epsilon$
   if $\phi \notin \mathsf{Bel}_{\mathcal{X}}(V_W(\mathcal{K}))$

3. for all $\langle \mathcal{X}, \mathcal{X}_r, \mathsf{type}, \phi \rangle \in \mathsf{Act}_{\mathcal{X}}$ with $\mathsf{type} \in \mathsf{types}_R$:
   $\epsilon \preceq^{a}_{\mathcal{K}} \langle \mathcal{X}, \mathcal{X}_r, \mathsf{type}, \phi \rangle$

4. $\preceq^{a}_{\mathcal{K}}$ is reflexive

5. $\preceq^{a}_{\mathcal{K}}$ is transitive

6. $\preceq^{a}_{\mathcal{K}}$ is minimal, that is, there is no $\preceq^{b}_{\mathcal{K}} \subset \preceq^{a}_{\mathcal{K}}$ that satisfies properties 1. to 5.

By Condition 1. honest speech acts are preferred over the empty action, by Condition 2. the empty action is preferred over insincere speech acts, and by Condition 3. requesting speech acts are preferred over the empty action. Further, the relation is reflexive, transitive, (Conditions 4. and 5.) and minimal with respect to Conditions 1. to 5. (Condition 6.). In the following we focus on the consideration of the secrecy preference relation $\preceq^{s}_{(\mathcal{K},\xi)}$ and the *informativity relation* $\preceq^{a}_{\mathcal{K}}$. For the aggregation we use the lexicographic aggregation operator with the order giving priority to secrecy over informativity, i. e.,

$$f^{\mathsf{lex}}_{agg}((\preceq^{s}_{(\mathcal{K},\xi)}, \preceq^{a}_{\mathcal{K}})).$$

Hence, we know that $\preceq_{(\mathcal{K},\xi)}$ satisfies *Safe Preference* and that preference is given to honest answers, if secrecy is not violated by them.

Agents normally cannot consider all possible actions Act and restrict their evaluation to a small subset of all actions that is dependent on the current agent state of the agent $\mathsf{Act}'_{(\mathcal{K},\xi)} \subseteq \mathsf{Act}$, e. g., those that might lead towards the satisfaction of one of their goals. As we showed, a secrecy preserving agent has to execute a sound action. This demands that the *considered subset of actions* $\mathsf{Act}'_{(\mathcal{K},\xi)}$ has to be such that if there is a sound action in Act, then there is a sound action in $\mathsf{Act}'_{(\mathcal{K},\xi)}$.

Example 5.5.1. If *Emma* is asked if she intends to attend the *strike committee meeting* then she should consider lying or not answering as an option in order be able to preserve secrecy. ♢

We assume that an agent, when it considers to answer a query, always at least considers to answers with all possible truth values, and that it considers to not answer the query. In the following we consider two possible truth values for a formula $\hat{\phi}$, that it is *true* or *false*. These are expressed as $\mathsf{evals}_{\mathsf{bool}}(\hat{\phi}) = \{\phi, \neg\phi\}$. This means that an agent always considers to respond with a lie to a query since one of the truth values is not true; and it considers to refuse to answer by not acting. These two options resemble the two common *distortion strategies lying* and *refusing* that are considered in the literature on confidentiality in information systems, see [32] for an overview. Our agents may very well consider additional options to answer a query by answering with other formulae or by responding to a query with another query, or with an *inform* speech act.

An agent that chooses actions in accordance with a preference relation that satisfies sound preference, and that always considers a set of actions that contains a sound action is secrecy preserving.

Proposition 5.5.2. An agent with initial agent state $(\mathcal{K}^0, \xi) \in (\Lambda, \mathfrak{F})$ is secrecy preserving if for all $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ and all $p \in \mathsf{Per}$:

a) $\preceq_{(\mathcal{K}\circ p,\xi)}$ satisfies sound preference

b) if there is a sound action $\alpha \in \mathsf{Act}$, then there is a sound action $\alpha' \in \mathsf{Act}'_{(\mathcal{K},\xi)}$

Proof. See Appendix A.1.3, Page 244. □

It might not be possible for the agent to preserve secrecy, because there is no sound action or because it considers a set of options that does not contain the existing sound options.

If the agent chooses its actions according to a preference relation that does not satisfy sound or safe preference it might also prefer to perform a secrecy violating action over a non-secrecy violating, because the utility with respect to some goal out-ways the loss of utility due to the violation of secrecy. A motivating example for this scenario is the following.

Example 5.5.3. An information agent providing medical information about patients should not reveal confidential information about a patient without a good reason. This reason is, for instance, given if the information is crucial for the preservation of the respective patient's health. ♢

For these cases of, unavoidable or avoidable, violations of secrecy the $\preceq^{s}_{(\mathcal{K},\xi)}$ relation should not only distinguish between actions that violate a secret and those that do not. It should distinguish the severeness of the violation due to an action as fine grained as possible. This way it can support the agent's decision making in these cases as good as possible. In the following we develop a preference relation that uses the available information as good as possible to distinguish actions with respect to secrecy. For this, in Section 5.5.2, we formalize some relevant notions to capture the violation of secrets, concrete a concrete change operator and a family of belief operators to be able to illustrate the concepts. Then, we develop general principles for the classification of actions with respect to secrecy in Section 5.5.3 and an algorithm for the construction of a preference relation satisfying them it in Section 5.5.4.

5.5.2 *Classification of actions with respect to secrecy*

We want to evaluate the effects of a given finite set of actions Act' with respect to secrecy on the basis of an epistemic state $\mathcal{K}$. We consider the general case that the defending agent $\mathcal{D}$ has an uncertain world-view, *uncertainty of the world-view*, and an uncertain view on the uncertain world-view of $\mathcal{A}$, the *uncertainty of the agent-view*. The uncertainty of $\mathcal{D}$ about the world-view of $\mathcal{A}$ is represented by defining the view of $\mathcal{D}$ on $\mathcal{A}$ to consist of a set of world-views. Formally, $V_{\mathcal{A}}(\mathcal{K}_{\mathcal{D}})$ is a non-empty set of world-views. We focus on the evaluation of a set of actions for a given epistemic state. The effects of an action are determined by the change operator $\circ^a$ such that we have to consider, for each action $a \in \mathsf{Act}$, the resulting epistemic state $\mathcal{K} \circ^a a$. Further, to evaluate the effects with respect to secrecy we need to capture the information about the violation of secrets as good as possible. To this end we define *violation sets* of an action a with respect to an epistemic state $\mathcal{K}$, denoted by $\mathsf{vioAfter}(\mathcal{K}, a)$, which capture the violation of secrecy. They comprise all sets of secrets that are potentially violated in conjunction after the action has been performed in the given epistemic state.

Definition 5.5.4 (Violation Sets). Let V_W be a world-view and $\mathcal{S}$ a set of active secrets. We define the set of secrets in $\mathcal{S}$ that are violated for world-view V_W:

$$\mathsf{vio}(\mathcal{S}, V_W) = \{(\phi, \mathsf{Bel}, \mathcal{A}) \in \mathcal{S} \mid \phi \in \mathsf{Bel}(V_W)\}.$$

We use the notion of violated secrets with respect to a single world-view to define the set of *combinations of secrets* that are *potentially violated in $\mathcal{K}$ after action* a:

$$\mathsf{vioAfter}(\mathcal{K}, a) = \{\mathsf{vio}(\mathcal{S}(\mathcal{K} \circ^a a), V_W) \mid V_W \in V_{\mathcal{A}}(\mathcal{K} \circ^a a)\}.$$

As noted before, we assume that all considered sets of secrets are sets of *active secrets* as defined in (5.2.1) (Page 112). We define the comparison of actions on the basis of their violation sets for a given epistemic state in the following. To illustrate our ideas we consider this classification for a simple instantiation of the change operator $\circ$ and for a simplified set of actions in the following. The approach works in the same way for general sets of actions and other change operators.

We restrict our consideration to inform speech acts (cf. Section 3.1). Since here the sender is always the defending agent and the receiver is always the attacking agent we leave out the sender and receiver identifier of the speech act and in the secrets. The set of speech acts is represented as:

$$\mathsf{Act}_{\mathsf{inf}} =_{def} \{\mathit{inform}(\phi) \mid \phi \in \mathcal{L}_{BS}\}$$

For this set of actions we define simple instantiations of the change operator $\circ$. We assumed before (Page 115) that the effect of $\circ$ can be expressed by means of the component specific operators, such that

$$\begin{aligned} V_W(\mathcal{K} \circ^a a) &= V_W(\mathcal{K}) *_{\mathcal{L}_W} t^a_W(\iota), \\ V_{\mathcal{A}}(\mathcal{K} \circ^a a) &= V_{\mathcal{A}}(\mathcal{K}) *_{\mathcal{L}_V} t^a_V(\iota) \text{ and} \\ \mathcal{S}(\mathcal{K} \circ^a a) &= \mathcal{S}(\mathcal{K}) \end{aligned}$$

For $\mathsf{Act}_{\mathsf{inf}}$ we assume a change operator $\circ^a_{\mathsf{inf}}$ such that

$$\begin{aligned} V_W(\mathcal{K} \circ^a_{\mathsf{inf}} \mathit{inform}(\phi)) &= V_W(\mathcal{K}), \\ V_{\mathcal{A}}(\mathcal{K} \circ^a_{\mathsf{inf}} \mathit{inform}(\phi)) &= \oplus(V_{\mathcal{A}}(\mathcal{K}), \mathit{inform}(\phi)), \\ \mathcal{S}(\mathcal{K} \circ^a_{\mathsf{inf}} \mathit{inform}(\phi)) &= \mathcal{S}(\mathcal{K}). \end{aligned}$$

That is, we reduce the change operator to a simple expansion operation of the agent-view. This operator considers the effects of an action for all world-views in a set of world views $\mathcal{W}$, which results in a set of *hypothetically evolved world-views*. These are determined by the *agent-view append operator* $\oplus$ as follows:

$$\oplus(\mathcal{W}, \mathit{inform}(\phi)) = \{V_W \hat{+} \phi \mid V_W \in \mathcal{W}\}. \tag{5.5.6}$$

To specify the *world-view append operator* $\hat{+}$ we have to define the language of world-view we consider here.

A world-view is represented as a sequence $\langle B; \phi_1, \ldots, \phi_n \rangle$ with two parts: the *background knowledge* B as a set of sentences from a belief base language $\mathcal{L}_E$ and *observations* $\phi_1, \ldots, \phi_n$ from the corresponding belief set language $\phi_i \in \mathcal{L}_{BS}$. In this sense, agent $\mathcal{D}$'s agent-view on $\mathcal{A}$, denoted $V_{\mathcal{A}}(\mathcal{K}_{\mathcal{D}})$, is a non-empty set $\mathcal{W} \subseteq \mathcal{P}(\mathcal{L}_{BB}) \times \mathcal{L}_{BS}{}^*$ of world-views. We also write $V(\mathcal{K})$ instead of $V_{\mathcal{A}}(\mathcal{K}_{\mathcal{D}})$ in the following. We call this set $\mathcal{D}$'s *assumed world-views of* $\mathcal{A}$. This way, $\mathcal{D}$ assumes that each $V_W \in \mathcal{W}$ could be held by $\mathcal{A}$, but that each $V_W \in (\mathcal{P}(\mathcal{L}_{BB}) \times \mathcal{L}_{BS}{}^*) \setminus \mathcal{W}$ is *not* held by $\mathcal{A}$. In this set, agent $\mathcal{D}$ might have incorporated a priori assumptions or observations of $\mathcal{A}$ as having been perceived by $\mathcal{D}$.

We consider an expansion operator that appends the informational content of the speech act, the formula $\phi \in \mathcal{L}_{BS}$, to a sequence of formulae. The belief operator then computes a consistent belief set from this sequence. This approach to change operations is called *immediate revision* in the literature [106]. The following operator which is used by $\mathcal{D}$ to anticipate the effect of its actions on the world-view of $\mathcal{A}$:

$$\hat{+} : \mathcal{P}(\mathcal{L}_{BB}) \times \mathcal{L}_{BS}{}^* \times \mathsf{Act}_{\mathsf{inf}} \to \mathcal{P}(\mathcal{L}_{BB}) \times \mathcal{L}_{BS}{}^* \text{ with}$$

$$\langle B; \phi_1, \ldots, \phi_n \rangle \hat{+} \mathit{inform}(\phi) =_{\mathit{def}} \langle B; \phi_1, \ldots, \phi_n, \phi \rangle. \tag{5.5.7}$$

Since here we consider this operator only in the context of $\mathcal{D}$ simulating changes to the world-view of $\mathcal{A}$, it takes the inform speech acts as input, which are actions in this context. That is, we need only the change operator for actions $\circ^a$, and not the $\circ$ operator for percepts. For clarity of presentation of our main ideas, we assume that $B \cup \{\phi_1, \ldots, \phi_n\}$ is consistent in the following. We assume that the belief operators of the secrets are taken from a belief operator family $(\Xi, \preceq_{\mathsf{bel}})$ to be given that satisfies *Consistency*, *Suprality* and *Credulity*$_{\preceq_{\mathsf{bel}}}$ cf. Section 3.6.

Example 5.5.5. For the illustration of this classification we consider a propositional instantiation of the belief base and belief set language as the knowledge representation formalism as shown in Propositional Instance 1 (Page 63). To adapt the propositional belief operator family $(\Xi^{\mathsf{prop}}, \preceq_{\mathsf{bel}}^{\mathsf{prop}})$ to the sequence based world-views we define

$$\mathsf{Mod}(\langle B; \phi_1, \dots, \phi_n \rangle) =_{def} \mathsf{Mod}(B \cup \{\phi_1, \dots, \phi_n\})$$

such that the belief operators are defined as follows

$$\mathsf{Bel}_p(\langle B; \phi_1, \dots, \phi_n \rangle) =_{def} \{\phi \in \mathcal{L} \mid r(\phi, \langle B; \phi_1, \dots, \phi_n \rangle) \geqslant p\} \quad (5.5.8)$$

with

$$r(\phi, \langle B; \phi_1, \dots, \phi_n \rangle) =_{def} \frac{|\{\mathsf{Mod}(\phi) \cap \mathsf{Mod}(B \cup \{\phi_1, \dots, \phi_n\})\}|}{|\mathsf{Mod}(B \cup \{\phi_1, \dots, \phi_n\})|}.$$

Each operator calculates the agent's certainty of the truth of a formula $\phi \in \mathcal{L}_{At}^{\mathsf{prop}}$ as the ratio of its models. The operator can be seen as accepting every formula as true that holds in at least $p \cdot 100$ percent of those models (given by the model operator Mod)

We consider that agent $\mathcal{A}$ reasons about a particular person which can have several properties that are represented by the following set of propositions $At = \{p_1, \dots, p_4, s_1, s_2\}$. Considering $\mathcal{A}$'s relationship to the person, agent $\mathcal{D}$ assumes that $\mathcal{A}$ knows whether the person has properties p_1 and p_2, whereas $\mathcal{D}$ does not know this about the person. Agent $\mathcal{D}$ assumes that $\mathcal{A}$ has the following background knowledge:

$$B = \{p_1 \wedge p_2 \wedge p_4 \Rightarrow s_1,\ p_2 \wedge p_3 \Rightarrow s_1,\ p_1 \wedge p_3 \Rightarrow s_2\}. \quad (5.5.9)$$

The background knowledge B expresses that if the person has properties p_1, p_2 and p_4, then it also has property s_1; if it has properties p_2 and p_3, then it also has property s_1; and if it has properties p_1 and p_3, then it has property s_2.

Thus, $\mathcal{D}$ assumes that the following world-views could be held by $\mathcal{A}$:

$$V_{W1} = (B; p_1, p_2), \qquad V_{W2} = (B; p_1, \neg p_2),$$
$$V_{W3} = (B; \neg p_1, p_2), \qquad V_{W4} = (B; \neg p_1, \neg p_2).$$

Agent $\mathcal{D}$ considers the properties s_1 and s_2 as secret formulae and has the following secrets towards $\mathcal{A}$:

$$\mathcal{S} = \{(s_1, \mathsf{Bel}_{0.7}),\ (s_2, \mathsf{Bel}_{0.6})\}.$$

Hence, $\mathcal{D}$ aims to protect property s_2 towards a more credulous reasoning behavior than property s_1, since $\mathsf{Bel}_{0.7} \preceq_{\mathsf{bel}} \mathsf{Bel}_{0.6}$. ◇

We can now simplify the definition of violation sets for the focused setting we consider here. Let $\mathcal{S} \subseteq \mathcal{L}_{\mathcal{S}}$ be a set of secrets, $V_W \in \mathcal{P}(\mathcal{L}_{BB}) \times \mathcal{L}_{BS}{}^*$ a world-view and $a \in \mathsf{Act}_{\mathsf{inf}}$ an inform speech act.

The secrets in $\mathcal{S}$ that are violated in world-view V_W are defined as the set

$$\mathsf{vio}(\mathcal{S}, V_W) = \{(\phi, \mathsf{Bel}) \in \mathcal{S} \mid \phi \in \mathsf{Bel}(V_W)\}.$$

As in the general case, we use this notion of violated secrets with respect to a single world-view to define the set of *combinations of secrets* from $\mathcal{S}$ that are *potentially violated with respect to a set of assumed world-views* $\mathcal{W}$ *after action* a:

$$\mathsf{vioAfter}(\mathcal{S}, \mathcal{W}, a) = \{\mathsf{vio}(\mathcal{S}, V_W) \mid V_W \in \oplus(\mathcal{W}, a)\}.$$

In words, each of these combinations, i. e., elements of the violation set $\mathsf{vioAfter}(\mathcal{S}, \mathcal{W}, a)$ is a set of secrets that are jointly violated in the same world-view $W \in \mathcal{W}$. In the context of an epistemic state $\mathcal{K}$, we often use the notation

$$\mathsf{vio}(\mathcal{K}, V_W) \text{ for } \mathsf{vio}(\mathcal{S}(\mathcal{K}), V_W)$$

and

$$\mathsf{vioAfter}(\mathcal{K}, a) \text{ for } \mathsf{vioAfter}(\mathcal{S}(\mathcal{K}), \mathcal{V}(\mathcal{K}), a).$$

We say that $\mathcal{D}$ *considers a secret potentially violated after action* a if the secret occurs in at least one set in $\mathsf{vioAfter}(\mathcal{K}, a)$.

Example 5.5.6. Agent $\mathcal{D}$ wants to decide whether to reveal that the person has property p_3, or to reveal less by saying $p_1 \vee p_2 \vee p_3$, or to intentionally lie to $\mathcal{A}$ about p_3 by saying $\neg p_3$. In the following we determine the violation sets for the three actions.

The assumed world-views of agent $\mathcal{A}$ are given by Example 5.5.5 (Page 137). It thus considers one of the options

$$\mathsf{Act}'_{\mathsf{inf}} = \{\mathit{inform}(p_3), \mathit{inform}(p_1 \vee p_2 \vee p_3), \mathit{inform}(\neg p_3)\}$$

to inform $\mathcal{A}$, referred to as a_1, a_2 and a_3, respectively.

Agent $\mathcal{D}$ evaluates the effect of each of its considered actions on the respective world-views, as defined in (5.5.6) as

$$\oplus(\mathcal{W}, \mathit{inform}(\phi)) = \{V_W \hat{+} \phi \mid V_W \in \mathcal{W}\}.$$

For the action $a_2 = \mathit{inform}(p_1 \vee p_2 \vee p_3)$ and the world-view $V_{W1} = \langle B; p_1, p_2 \rangle$ the used $\hat{+}$ operation, as defined in (5.5.7) (Page 136) results in:

$$V_{W1} \hat{+} a_2 = \langle B; p_1, p_2, p_1 \vee p_2 \vee p_3 \rangle$$

The belief operator family is the propositional one given in Example 5.5.5 (Page 137). We want to determine which of the secrets are violated in the hypothetically evolved world-views. To this end we compute the ratios $r(s, V_W + a)$ for all combinations of a secret formula s, a world-view V_W and an action a. The different belief operators only differ in their threshold value p.

Exemplarily we describe the computation of the ratio for the secret formula s_1 in the potential evolution of the world-view V_{W1} by action a_2. We calculate the ratio

$$r(s_1, V_{W1} \hat{+} a_2) = \frac{|\mathsf{Mod}(s_1) \cap \mathsf{Mod}(V_{W1} \hat{+} a_2)|}{|\mathsf{Mod}(V_{W1} \hat{+} a_2)|}.$$

The models of the potential evolution of the world-view V_{W1} by action a_2 is given in the following. We represent a model of a set of formulae by the concatenation of all propositions and denoting negative truth values by a line over the proposition.

$$\begin{aligned}
&\mathsf{Mod}(\langle B; p_1, p_2, p_1 \vee p_2 \vee p_3 \rangle) = \\
&\mathsf{Mod}(B \cup \{p_1, p_2, p_1 \vee p_2 \vee p_3\}) = \\
&\qquad \{\, p_1\, p_2\, p_3\, p_4\, s_1\, s_2,\ p_1\, p_2\, p_3\, \overline{p_4}\, s_1\, s_2, \\
&\qquad\ \ p_1\, p_2\, \overline{p_3}\, p_4\, s_1\, s_2,\ p_1\, p_2\, \overline{p_3} p_4\, s_1\, \overline{s_2}, \\
&\qquad\ \ p_1\, p_2\, \overline{p_3}\, \overline{p_4}\, s_1\, s_2,\ p_1\, p_2\, \overline{p_3}\, \overline{p_4}\, \overline{s_1}\, s_2, \\
&\qquad\ \ p_1\, p_2\, \overline{p_3}\, \overline{p_4}\, s_1\, \overline{s_2},\ p_1\, p_2\, \overline{p_3}\, \overline{p_4}\, \overline{s_1}\, \overline{s_2} \,\}
\end{aligned}$$

Now we consider those of the models listed above that are also models of the secret formula s_1:

$$\begin{aligned}
&\mathsf{Mod}(B \cup \{p_1, p_2, p_1 \vee p_2 \vee p_3\}) \cap \mathsf{Mod}(s_1) = \\
&\qquad \{\, p_1\, p_2\, p_3\, p_4\, s_1\, s_2,\ p_1\, p_2\, p_3\, \overline{p_4}\, s_1\, s_2, \\
&\qquad\ \ p_1\, p_2\, \overline{p_3}\, p_4\, s_1\, s_2,\ p_1\, p_2\, \overline{p_3} p_4\, s_1\, \overline{s_2}, \\
&\qquad\ \ p_1\, p_2\, \overline{p_3}\, \overline{p_4}\, s_1\, s_2,\ p_1\, p_2\, \overline{p_3}\, \overline{p_4}\, s_1\, \overline{s_2} \,\}.
\end{aligned}$$

We can see that the ratio is $r(s_1, V_{W1} \hat{+} a_2) = \frac{6}{8} = \frac{3}{4} = 0.75$. Consequently the secret $(s_1, \mathsf{Bel}_{0.7})$ is violated since $r(s_1, V_{W1} \hat{+} a_2) = 0.75 \geqslant 0.7$ such that $s_1 \in \mathsf{Bel}_{0.7}(V_{W1} \hat{+} a_2)$.

The ratios of all the combinations of actions a_1 and a_2 with all world-views are given in the following table. The effects of action a_3 are not shown in the table since after that action no secret is potentially violated. We can find the result we just calculated in the first cell. The gray background of a cell indicates a violation of the respective secret.

	$r(s_1, V_{Wi})$	$r(s_2, V_{Wi})$
$V_{W1}\hat{+}a_2$	0.75	0.625
$V_{W2}\hat{+}a_2$	0.5	$0.\overline{6}$
$V_{W3}\hat{+}a_2$	$0.\overline{6}$	0.5
$V_{W4}\hat{+}a_2$	0.5	0.5

	$r(s_1, V_{Wi})$	$r(s_2, V_{Wi})$
$V_{W1}\hat{+}a_1$	1	1
$V_{W2}\hat{+}a_1$	1	0.5
$V_{W3}\hat{+}a_1$	0.5	1
$V_{W4}\hat{+}a_1$	0.5	0.5

Given the complete table we can easily determine the combinations of secrets potentially violated after each action resulting in the following violation sets:

$$
\begin{aligned}
&\mathsf{vioAfter}(\mathcal{K}_\mathcal{D}, a_1) = \mathsf{vioAfter}(\mathcal{K}_\mathcal{D}, a_2) = \{\\
&\quad \{(s_2, \mathsf{Bel}_{0.6}), (s_1, \mathsf{Bel}_{0.7})\},\\
&\quad \{(s_2, \mathsf{Bel}_{0.6})\}, \{(s_1, \mathsf{Bel}_{0.7})\},\\
&\quad \emptyset\}
\end{aligned}
$$

and $\mathsf{vioAfter}(\mathcal{K}_\mathcal{D}, a_3) = \{\emptyset\}$. ◇

Based on the notions we just introduced, we consider the classification of actions by natural numbers of a given set of actions with respect to secrecy. Formally, a *classification* is a function $\mathsf{cl} : \mathsf{Act}'_{\mathsf{inf}} \to \mathbb{N}_0$ such that $\mathsf{cl}^{-1}(0) \neq \emptyset$. The lower the classification rank of an action, the lesser is the potential violation of secrecy by this action. We denote the set of all classifications of arbitrary finite subsets $\mathsf{Act}'_{\mathsf{inf}}$ of actions by Cl. Given such a classification cl of $\mathsf{Act}'_{\mathsf{inf}}$ the canonical preference relation on $\mathsf{Act}'_{\mathsf{inf}}$ based on cl is given by

$$\preceq^s_{\mathsf{cl}} =_{def} \{(a, a') \mid a, a' \in \mathsf{Act}'_{\mathsf{inf}} \text{ and } \mathsf{cl}(a) \geqslant \mathsf{cl}(a')\}. \qquad (5.5.10)$$

In the next section we develop a set of principles that describes how actions should be classified.

5.5.3 *Principles for the classification of actions*

In this section, we define how a defending agent $\mathcal{D}$ should classify a set of actions with respect to secrecy. We introduce several principles to this end. Before we do so, we introduce the notion of set-inclusion maximal sets of a set X of sets which we use for several definitions, it is denoted by

$$\max_{\subseteq} X := \{S \in X \mid \text{ there is no } S' \in X \text{ such that } S \subset S'\}.$$

In the following we first give the intuition of the principles and then the formalization.

Principle I.1 (*Avoid potential violations*) intuitively expresses that the agent, $\mathcal{D}$, desires that as few secrets as possible are violated in its assumed world-views. Hence, an action b should be classified higher (worse) than another action a given an epistemic state $\mathcal{K}$ if after the former action more secrets are potentially violated in combination. Given the definition of violation sets in Definition 5.5.4, we formalize this comparison by the $\sqsupset$ relation that is defined as follows:

$$\mathsf{vioAfter}(\mathcal{K}, b) \sqsupset \mathsf{vioAfter}(\mathcal{K}, a) \text{ if and only if} \tag{5.5.11}$$

1. for all $S_a \in \mathsf{vioAfter}(\mathcal{K}, a)$ there exists $S_b \in \mathsf{vioAfter}(\mathcal{K}, b)$ such that $S_a \subseteq S_b$ and
2. there exists some $S_a \in \max_{\subseteq} \mathsf{vioAfter}(\mathcal{K}, a)$ such that there exists some $S_b \in \mathsf{vioAfter}(\mathcal{K}, b)$ with $S_a \subset S_b$.

Example 5.5.7. Considering Example 5.5.6, it holds that

$$\mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_1) \sqsupset \mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_3) \text{ and}$$

$$\mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_2) \sqsupset \mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_3),$$

but $\mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_1)$ and $\mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_2)$ are not ordered by $\sqsupset$. ◇

The $\sqsupset$ relation does also not relate two actions if it lacks the information in $\mathcal{K}$ to compare combinations of secrets potentially violated after these actions. This is the case if in $\mathcal{K}$ for neither of two actions a and b the first condition for the relation $\sqsupset$ is satisfied. In this case, there are secrets potentially violated after a in combination that are not violated in combination after action b, and there is another set of secrets that is potentially violated after b in combination, but not after action a. In such a case, it cannot be decided which action's execution is better.

Example 5.5.8. The following pairs of violation sets, i. e., sets of sets of secrets that are violated in combination, are examples that are not ordered by $\sqsupset$:

1. $\{\{(s_1, \mathsf{Bel}_{0.7})\}, \emptyset\}$ and $\{\{(s_2, \mathsf{Bel}_{0.6})\}, \emptyset\}$
2. $\{\{(s_1, \mathsf{Bel}_{0.7})\}, \emptyset\}$ and $\{\{(s_2, \mathsf{Bel}_{0.6}), (s_3, \mathsf{Bel}_{0.5})\}, \emptyset\}$.

In Case 1. the violations sets cannot be ordered even though the belief operators of the secret $(s_1, \mathsf{Bel}_{0.7})$ and $(s_2, \mathsf{Bel}_{0.6})$ can be ordered since the secret formulae are different and violations with respect to different formulae cannot be compared. Formally, the condition 1. of Equation (5.5.11) is violated since $\{(s_1, \mathsf{Bel}_{0.7})\} \not\subseteq \{(s_2, \mathsf{Bel}_{0.6})\}$ and $\{(s_1, \mathsf{Bel}_{0.7})\} \not\subseteq \emptyset$.

In Case 2. the violation sets cannot be compared even though in the first one only one secret is maximally violated and in the second one two secrets are maximally violated in combination. This is due to the fact that there is a secret in each of the violation sets that is not contained in the other and that violations with respect to different formulae cannot be compared. The sole number of violations is not sufficient to order violation sets. Formally, the condition 1. of Equation (5.5.11) is violated since $\{(s_1, \mathsf{Bel}_{0.7})\} \not\subseteq \{(s_2, \mathsf{Bel}_{0.6}), (s_3, \mathsf{Bel}_{0.5})\}\}$ and $\{(s_1, \mathsf{Bel}_{0.7})\} \not\subseteq \emptyset$, and $\{(s_2, \mathsf{Bel}_{0.6})\} \not\subseteq \{(s_1, \mathsf{Bel}_{0.7})\}$. ◇

Principle I.2 (*Confine degrees of violations*) addresses that the case that secrets might be violated to different degrees. Confining the degree of violation means that, if a secret formula could not be protected against inferences with Bel as desired, it should at least be protected against inferences with operators more skeptical than Bel. To this end, we consider actions with the same effect concerning combinations of potentially violated secrets in an epistemic state $\mathcal{K}$, i. e., actions $a, b \in \mathsf{Act}'_{\mathsf{inf}}$ such that

$$\mathsf{vioAfter}(\mathcal{K}, a) \sim \mathsf{vioAfter}(\mathcal{K}, b) \text{ defined as} \\ \max_{\subseteq} \mathsf{vioAfter}(\mathcal{K}, a) = \max_{\subseteq} \mathsf{vioAfter}(\mathcal{K}, b). \quad (5.5.12)$$

Such pairs of actions are incomparable with respect to the $\sqsupset$ relation. But they can be compared with respect to the degrees of the violations of secrets as formalized in the following relation.

Definition 5.5.9. Let $\mathcal{K}$ be an epistemic state and $a, b \in \mathsf{Act}'_{\mathsf{inf}}$ such that $\mathsf{vioAfter}(\mathcal{K}, a) \sim \mathsf{vioAfter}(\mathcal{K}, b)$. *Action* a *has a lesser degree of potential violation than* b, written as $a \prec_{vio} b$, if the following two conditions hold:

1. There exists $S \in \max_{\subseteq} \mathsf{vioAfter}(\mathcal{K}, b)$ such that there exist $(\phi, \mathsf{Bel}_1) \in S$ and $\mathsf{Bel}'_1 \in \Xi$ with $\mathsf{Bel}'_1 \prec_{\mathsf{bel}} \mathsf{Bel}_1$ such that the secret (ϕ, Bel'_1) is potentially violated after action b in $\mathcal{K}$, but not after action a.

2. There are no $(\phi, \mathsf{Bel}_2) \in S$ and $\mathsf{Bel}'_2 \in \Xi$ with $\mathsf{Bel}'_2 \prec_{\mathsf{bel}} \mathsf{Bel}_2$ such that the secret (ϕ, Bel'_2) is potentially violated after action a in $\mathcal{K}$, but not after action b.

Example 5.5.10. The table in Example 5.5.6 shows that for each row for action a_2 in which there are secrets violated, there exists a row for action a_1 in which the same secrets are violated but with strictly higher ratios. This does not hold the other way round. From this follows that $a_2 \prec_{vio} a_1$, but not $a_1 \prec_{vio} a_2$.

Formally, this is the case since

$$\mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_1) \sim \mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_2)$$

holds such that the precondition of Definition 5.5.9 is satisfied. There is just one inclusion maximal violation set such that

$$\begin{aligned}\max_{\subseteq} \mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_1) &= \max_{\subseteq} \mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, \mathcal{D}, a_2)\\ &= \{(s_2, \mathsf{Bel}_{0.6}), (s_1, \mathsf{Bel}_{0.7})\}.\end{aligned}$$

For the secret $(s_2, \mathsf{Bel}_{0.6})$ in $\{(s_2, \mathsf{Bel}_{0.6}), (s_1, \mathsf{Bel}_{0.7})\}$ we obtain by replacing the operator $\mathsf{Bel}_{0.6}$ by the operator $\mathsf{Bel}_{0.7}$ that

$$\begin{aligned}&\mathsf{vioAfter}(\{(s_2, \mathsf{Bel}_{0.7})\}, V(\mathcal{K}), a_1) = \{\{(s_2, \mathsf{Bel}_{0.7})\}\} \text{ and}\\ &\mathsf{vioAfter}(\{(s_2, \mathsf{Bel}_{0.7})\}, V(\mathcal{K}), a_2) = \{\emptyset\}.\end{aligned}$$

Hence, the Condition 1. is satisfied for the consideration of $a_2 \prec_{vio} a_1$. For the other secret $(s_1, \mathsf{Bel}_{0.7})$ in $\{(s_2, \mathsf{Bel}_{0.6}), (s_1, \mathsf{Bel}_{0.7})\}$ we obtain by replacing the operator $\mathsf{Bel}_{0.7}$ by the operator $\mathsf{Bel}_{0.8}$ that

$$\begin{aligned}&\mathsf{vioAfter}(\{(s_1, \mathsf{Bel}_{0.8})\}, V(\mathcal{K}), a_1) = \{\{(s_1, \mathsf{Bel}_{0.8})\}\} \text{ and}\\ &\mathsf{vioAfter}(\{(s_1, \mathsf{Bel}_{0.8})\}, V(\mathcal{K}), a_2) = \{\emptyset\}.\end{aligned}$$

Thus, there are no $(\phi, \mathsf{Bel}_2) \in S$ and $\mathsf{Bel}_2' \in \Xi$ with $\mathsf{Bel}_2' \prec_{\mathsf{bel}} \mathsf{Bel}_2$ such that (ϕ, Bel_2') is potentially violated after action a_2 in $\mathcal{K}$, but not after action a_1. Hence, the Condition 2. is satisfied for the consideration of $a_2 \prec_{vio} a_1$ and the Condition 1. is falsified for the consideration of $a_1 \prec_{vio} a_2$. $\diamondsuit$

If action a has a lesser degree of potential violation than action b it should be classified lower than the latter. However, the relation $\prec_{vio}$ is not acyclic such that $a \prec_{vio} b$ can be part of a cycle, i. e.,

there are actions $a_1, \ldots, a_n \in \mathsf{Act}_{\mathsf{inf}}'$ such that
$a_1 = b$, $a_n = a$ and $a_i \prec_{vio} a_{i+1}$ for all $i \in \{1, \ldots, n-1\}$.

In this case there is no justification to classify a lower than b and we call a and b *conflicting*.

Principle I.2 demands that if a has a lesser degree of potential violation than b and a and b are not conflicting, then a should be classified lower than b. If $a \prec_{vio} b$ and a and b are conflicting, then the principle accounts for this by requiring that a should be classified at least as low as b.

Principle II (*Minimize classification*) requires that the classification should be as little restrictive as possible with regard to $\mathcal{D}$'s possible other desires such as cooperative information sharing. The lower the classification rank of an action the less $\mathcal{D}$ is admonished to refrain from executing that action. In particular, with respect to secrecy no restriction is posed on all actions with a rank of 0.

Having described our core set of principles we now formalize these and describe a secrecy reasoner that outputs a classification of actions in accordance with the principles.

Definition 5.5.11 (Secrecy Reasoner For Action Classification). A *secrecy reasoner* is a function $\mathsf{sr} : \mathcal{P}_{\mathsf{fin}}(\mathsf{act}) \times \mathcal{L}_{ES} \to Cl$ that is parameterized with a family $(\Xi, \preceq_{\mathsf{bel}})$ of belief operators satisfying credulity. Hereby, $\mathcal{P}_{\mathsf{fin}}(S)$ denotes the set of finite subsets of S. Thus, a secrecy reasoner takes a finite subset $\mathsf{Act}'_{\mathsf{inf}}$ of $\mathsf{Act}_{\mathsf{inf}}$ and an epistemic state $\mathcal{K}$ as input, and outputs a classification cl of $\mathsf{Act}'_{\mathsf{inf}}$. Moreover, the function sr has to fulfill the following principles:

PRINCIPLE I.1: AVOID POTENTIAL VIOLATIONS

Let $a, b \in \mathsf{Act}'_{\mathsf{inf}}$ be actions and $\mathcal{K}$ an epistemic state such that

$$\mathsf{vioAfter}(\mathcal{K}, b) \sqsupset \mathsf{vioAfter}(\mathcal{K}, a).$$

Further, let $\mathsf{cl} = \mathsf{sr}(\mathsf{Act}'_{\mathsf{inf}}, \mathcal{K})$ be the reasoner's classification of $\mathsf{Act}'_{\mathsf{inf}}$ in $\mathcal{K}$. Then, it holds that $\mathsf{cl}(b) > \mathsf{cl}(a)$.

PRINCIPLE I.2: CONFINE DEGREES OF VIOLATIONS

Let $\mathcal{K}$ be an epistemic state and $a, b \in \mathsf{Act}'_{\mathsf{inf}}$ such that $a \prec_{vio} b$, then:

1. *Conflict Free*: If there do not exist actions $a_1, \ldots, a_n \in \mathsf{Act}'_{\mathsf{inf}}$ such that $a_1 = b$, $a_n = a$ and $a_i \prec_{vio} a_{i+1}$ for all $i \in \{1, \ldots, n-1\}$, then it holds that $\mathsf{cl}(b) > \mathsf{cl}(a)$ with $\mathsf{cl} = \mathsf{sr}(\mathsf{Act}'_{\mathsf{inf}}, \mathcal{K})$.
2. *Conflicting*: Otherwise, it holds $\mathsf{cl}(b) \geqslant \mathsf{cl}(a)$.

PRINCIPLE II: MINIMIZE CLASSIFICATION

Given $(\Xi, \preceq_{\mathsf{bel}})$ as parameter, let sr' be another function

$$\mathsf{sr}' : \mathcal{P}_{\mathsf{fin}}(\mathsf{Act}_{\mathsf{inf}}) \times \mathcal{L}_{ES} \to Cl$$

fulfilling Principles I.1 and I.2

Then, for all $\mathsf{Act}'_{\mathsf{inf}} \subset_{\mathsf{fin}} \mathsf{Act}_{\mathsf{inf}}$, for all $\mathcal{K} \in \mathcal{L}_{ES}$ and for all $a \in \mathsf{Act}'_{\mathsf{inf}}$ it holds $\mathsf{cl}'(a) \geqslant \mathsf{cl}(a)$ with $\mathsf{cl}' = \mathsf{sr}'(\mathsf{Act}'_{\mathsf{inf}}, \mathcal{K})$ and $\mathsf{cl} = \mathsf{sr}(\mathsf{Act}'_{\mathsf{inf}}, \mathcal{K})$.

Beside the core Principles I.1, I.2 and II we define two additional principles that express interesting and desirable properties for a secrecy reasoner. These principles can be shown to be consequences of the core principles. Both principles consider the cautiousness of a secrecy reasoner in classifying an action with least violation (classification rank 0). Again, we first describe the intuitive ideas of the principles and then formalize these.

Principle III (*Be cautious towards credulous reasoners*) follows the intuition that the more credulous $\mathcal{A}$ is assumed to reason about a secret formula, the easier that sentence might be inferred because $\mathcal{A}$ may accept more propositions as true and believe them. Thus, the more credulous the belief operator is in a secrecy constraint, the more cautious agent $\mathcal{D}$ has to be while acting.

Principle IV (*Be more cautious the more uncertain*) bases on the general idea that being uncertain leads to cautious behavior. Here, the more world-views could be held by $\mathcal{A}$ according to the assumed world-views, the more uncertain $\mathcal{D}$ is about $\mathcal{A}$'s actual situation.

PRINCIPLE III: BE CAUTIOUS TOWARDS CREDULOUS REASONERS

Let $\mathcal{K}$ and $\mathcal{K}'$ be epistemic states with equal components possibly except for the sets of secrets that are of the following form:

$$\mathcal{S}(\mathcal{K}) = \{(\phi_1, \mathsf{Bel}_1), \dots (\phi_n, \mathsf{Bel}_n)\},$$

$$\mathcal{S}(\mathcal{K}') = \{(\phi_1, \mathsf{Bel}_1{}'), \dots (\phi_n, \mathsf{Bel}_n{}')\}$$

with $\mathsf{Bel}_i{}' \preceq_{\mathsf{bel}} \mathsf{Bel}_i$ for all $i \in \{1, \dots, n\}$.

If there exist actions $a, b \in \mathsf{Act}'_{\mathsf{inf}}$ such that

$$\mathsf{vioAfter}(\mathcal{K}, a) = \{\emptyset\} \text{ and } \mathsf{vioAfter}(\mathcal{K}', b) = \{\emptyset\},$$

then for the classifications $\mathsf{cl} = \mathsf{sr}(\mathsf{Act}'_{\mathsf{inf}}, \mathcal{K})$ and $\mathsf{cl}' = \mathsf{sr}(\mathsf{Act}'_{\mathsf{inf}}, \mathcal{K}')$ it holds that

$$\{a \in \mathsf{Act}'_{\mathsf{inf}} \mid \mathsf{cl}(a) = 0\} \subseteq \{a \in \mathsf{Act}'_{\mathsf{inf}} \mid \mathsf{cl}'(a) = 0\}.$$

In this formalization two sets of secrets $\mathcal{S}(\mathcal{K})$ and $\mathcal{S}(\mathcal{K}')$ are considered. It is assumed that both are identical up to the belief operators in the secrets. It is further assumed that the belief operator with respect to each secret formula in $\mathcal{S}(\mathcal{K})$ is at least as credulous as the belief operator with respect to the same formula in $\mathcal{S}(\mathcal{K}')$. Then the case is considered in which there exists an action for each set of secrets that does not violate any secret. In this case all actions with classification rank 0 do not violate any secret. In this case we can compare these sets of actions with rank 0 for different epistemic states on the basis of set inclusion. If there are no actions that do not violate any secret, actions with rank 0 with respect to different epistemic states might violate secrecy to very different degrees and cannot be related on the basis of set inclusion to make a statement about the cautiousness of the agent.

For the case of the existence of secrecy preserving actions, it is then demanded that all actions that are classified with rank 0 with respect to $\mathcal{S}(\mathcal{K})$ are also classified with rank 0 with respect to $\mathcal{S}(\mathcal{K}')$. The more actions are classified with rank 0 with respect to secrecy, the less restricted is the agent in its choice of actions.

PRINCIPLE IV: BE MORE CAUTIOUS THE MORE UNCERTAIN

Let $\mathcal{K}$ and $\mathcal{K}'$ be epistemic states with equal components except for $V(\mathcal{K}) \supseteq V(\mathcal{K}')$. If there exist actions $a, b \in \mathsf{Act}'_{\mathsf{inf}}$ such that

$$\mathsf{vioAfter}(\mathcal{K}, a) = \{\emptyset\} \text{ and } \mathsf{vioAfter}(\mathcal{K}', b) = \{\emptyset\},$$

then for the classifications $\mathsf{cl} = \mathsf{sr}(\mathsf{Act}'_{\mathsf{inf}}, \mathcal{K})$ and $\mathsf{cl}' = \mathsf{sr}(\mathsf{Act}'_{\mathsf{inf}}, \mathcal{K}')$ it holds that

$$\{a \in \mathsf{Act}'_{\mathsf{inf}} \mid \mathsf{cl}(a) = 0\} \subseteq \{a \in \mathsf{Act}'_{\mathsf{inf}} \mid \mathsf{cl}'(a) = 0\}.$$

If $V(\mathcal{K}) \supseteq V(\mathcal{K}')$ then an agent with epistemic state $\mathcal{K}'$ is at least as uncertain about the world-view of $\mathcal{A}$ than an agent with epistemic state $\mathcal{K}$. In this setting, as for Principle III, the case is considered in which there exists an action for each set of secrets that does not violate any secret. The principle formalizes that in this case the set of actions that are classified with rank 0 with respect to $\mathcal{S}(\mathcal{K})$ are also classified with rank 0 with respect to $\mathcal{S}(\mathcal{K}')$. As said above, the more actions are classified with rank 0 with respect to secrecy, the less restricted is the agent in its choice of actions.

The next proposition shows that the Principles III and IV are consequences of the core principles.

Proposition 5.5.12. Any function $\mathsf{sr} : \mathcal{P}_{\mathsf{fin}}(\mathsf{Act}) \times \mathcal{L}_{ES} \to Cl$ that satisfies Principles I.1, I.2 and II also satisfies Principle III and Principle IV.

Proof. See Appendix A.1.3, Page 244. □

After this declarative description of the classification task we turn to a constructive approach in the next section.

5.5.4 *Algorithm for the classification of actions*

In this section we present an algorithm, in Procedure 5.5.1, that implements a secrecy reasoner as defined in Definition 5.5.11. That is, it implements a function $\mathsf{sr} : \mathcal{P}_{\mathsf{fin}}(\mathsf{act}) \times \mathcal{L}_{ES} \to Cl$ being parameterized with a family $(\Xi, \preceq_{\mathsf{bel}})$ of belief operators with credulity order that takes as input a finite subset $\mathsf{Act}'_{\mathsf{inf}}$ of $\mathsf{Act}_{\mathsf{inf}}$ and an epistemic state $\mathcal{K}$ and outputs a classification cl of $\mathsf{Act}'_{\mathsf{inf}}$.

The main idea of the algorithm is to keep track of all not yet classified actions, denoted by $\mathsf{unclass}$, while it iteratively assigns the currently considered (classification) rank to the actions of the input set of actions $\mathsf{Act}'_{\mathsf{inf}}$. To this end, starting with a classification rank of 0, all actions for which there is no reason not to classify them with the current rank are assigned the current rank as their classification. Intuitively, a reason not to classify an action a with the current rank is given if more secrets are potentially violated in combination after a than after another unclassified action b (Principle I.1), or another unclassified action b' has a lesser degree of violation than a, without a conflict (Principle I.2).

In addition to the definitions already introduced, the algorithm makes use of auxiliary sets to test the preconditions of Principle I.2, which we define in the following. For a given epistemic state $\mathcal{K}$ the algorithm determines the set of equivalence classes with respect to the equivalence relation $\sim$ defined in (5.5.12) for an inspected set of actions A; this set is denoted by $A/{\sim}$. Only pairs taken from the same equivalence class might satisfy the precondition of Principle I.2 by definition. Further, the algorithm has to be able to test *conflict freeness* of a pair of actions a, b with $a \prec_{vio} b$. For this, all (maximal) conflict sets $\mathsf{conflictSets}(A, \mathcal{K})$ are computed for a given equivalence class A. Formally:

$$\begin{aligned}\mathsf{conflictSets}(A,\mathcal{K}) = \max\nolimits_{\subseteq} \{A' \subseteq A \mid{} & \text{for all } a, b \in A',\ a \neq b \text{ exist} \\ & a_1, \dots, a_n \in A' \text{ such that } a_1 = a,\ a_n = b \\ & \text{and for all } i \in \{1, \dots, n-1\}\text{: } a_i \prec_{vio} a_{i+1}.\}\end{aligned}$$

Note, that for any pair of actions $\{a, b\} \subseteq A'$ such that

$$A' \in \mathsf{conflictSets}(A, \mathcal{K})$$

for some A and $\mathcal{K}$ with $a \prec_{vio} b$ the condition *conflict free* of Principle I.2 is violated.

Lemma 5.5.13. Given some set of actions A and an epistemic state $\mathcal{K}$. The set $\mathsf{conflictSets}(A, \mathcal{K})$ is a partition of A, i. e.

1. $\bigcup \mathsf{conflictSets}(A, \mathcal{K}) = A$ and
2. for all $CS, CS' \in \mathsf{conflictSets}(A, \mathcal{K})$, $CS \neq CS'$ it holds that $CS \cap CS' = \emptyset$.

Proof. See Appendix A.1.3, Page 247. □

The algorithm presented in Procedure 5.5.1 consists of two nested repeat-until loops. The outer one determines in each execution the set of not yet classified actions for which Principle I.1 does not give a reason not to classify them with the currently considered classification rank. The inner one determines in each execution the set of actions out of the selected actions from the outer loop for which Principle I.2 does not give a reason not to classify them with the currently considered classification rank and classifies them.

The outer loop first determines the subset of currently unclassified actions best whose effects are not worse concerning combinations of potentially violated secrets than the effects of other unclassified actions (Principle I.1). Then, it constructs the auxiliary sets to check the precondition for Principle I.2. In particular, it creates a partitioning eqbest of the set best consisting of the equivalence classes wrt. $\sim$. Then, an array $\mathsf{rank}[]$ is created that holds for each equivalence class $A \in \mathsf{eqbest}$ its currently considered classification rank. Each $\mathsf{rank}[A]$ is initialized in Line 9 by the minimal classification rank for which Principle I.1 does not give a reason to classify any action of A higher than any action that is already classified. In the for-loop from Line 11

Procedure 5.5.1 Secrecy Reasoner

```
Input: $\mathsf{Act}'_{\mathsf{inf}}, \mathcal{K}, (\Xi, \preceq_{\mathsf{bel}})$
Output: Array cl of classification ranks for actions $a \in \mathsf{Act}'_{\mathsf{inf}}$
1: unclass := $\mathsf{Act}'_{\mathsf{inf}}$
2: for each $a \in \mathsf{Act}'_{\mathsf{inf}}$ do
3:     cl[a] := 0
4: end for
5: repeat
6:     best := {$a \in$ unclass | there is no $b \in$ unclass
                                such that vioAfter($\mathcal{K}, a$) $\sqsupset$ vioAfter($\mathcal{K}, b$)}
7:     eqbest := best/$\sim$
8:     for each $A \in$ eqbest do
9:         rank[A] := $\max^1${ cl[a] | $a \in \mathsf{Act}'_{\mathsf{inf}} \setminus$ unclass and          (*)
                             there is $b \in A$ such that
                             vioAfter($\mathcal{K}, b$) $\sqsupset$ vioAfter($\mathcal{K}, a$)} + 1
10:    end for
11:    for each $A \in$ eqbest do
12:        conflictSets := conflictSets($A, \mathcal{K}$)
13:        repeat
14:            classSets := $\emptyset$
15:            for each $CS \in$ conflictSets do
16:                if there is no $CS' \in$ conflictSets with $CS' \neq CS$ such that
                   $a' \in CS'$ and $a \in CS$ exist with $a' \prec_{vio} a$ then
17:                    cl$a$ := rank[A] for all $a \in CS$
18:                    classSets := classSets $\cup$ {CS}
19:                end if
20:            end for
21:            conflictSets := conflictSets $\setminus$ classSets
22:            rank[A] := rank[A] + 1
23:        until conflictSets = $\emptyset$
24:    end for
25:    unclass := unclass $\setminus$ best
26: until unclass = $\emptyset$
```

(*) We define $\max(\emptyset) := -1$, which is needed in the first iteration only.

to Line 24 each current equivalence class A is first partitioned into its conflict sets.

The for-loop from Line 15 to Line 20 is intended to determine all actions of the current equivalence class for which no other action in the same equivalence class exists such that this pair satisfies the precondition of Principle I.2. This is done by comparing the conflict sets and either classifying all elements of the conflict set or none. This way all elements of a conflict set are classified with the same classification rank. The classified conflict sets of the currently considered equivalence class are stored in classSets and removed from the current set of conflict sets after the termination of the for-loop over all conflict sets in Line 21.

If all actions in best are classified the condition in Line 23 is true and they are removed from the set of unclassified actions in Line 25. If all input actions are classified the condition in Line 26 is true and the algorithm terminates.

Example 5.5.14. We consider the execution of Procedure 5.5.1 for the running example, continuing Example 5.5.10.

Iteration 1:
Initially we have $\mathsf{unclass} := \{a_1, a_2, a_3\}$ in Line 1. As shown in Example 5.5.6 it holds that

$$\mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_1) = \mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_2) \text{ and}$$

$$\mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_1) \sqsupset \mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_3)$$

such that $\mathtt{best} := \{a_3\}$ and therefore also $\mathsf{eqbest} = \{\{a_3\}\}$.

Since $\mathsf{Act}'_{\mathsf{inf}} \setminus \mathsf{unclass} = \emptyset$ it holds for all A that $\mathsf{rank}[A] = 0$. In Line 12 we get $\mathsf{conflictSets} = \{\{a_3\}\}$. The conflict set $\{a_3\}$ trivially satisfies the condition in Line 16 such that $\mathsf{cl}a_3 = 0$. Then $\{a_3\}$ is removed from conflictSets such that $\mathit{conflictsSets} = \emptyset$. In Line 22 $\mathit{rank}[\{a_3\}] := 1$, which does not have any effect in this special case, and the inner repeat-until loop terminates.

Iteration 2:
For the second execution of the outer repeat-until loop $\mathsf{unclass} = \{a_1, a_2\}$. Since

$$\mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_1) = \mathsf{vioAfter}(\mathcal{K}_{\mathcal{D}}, a_2)$$

it follows that $\mathsf{best} := \{a_1, a_2\}$ and $\mathsf{eqbest} = \{\{a_1, a_2\}\}$. Then, in Line 9 we get $\mathit{rank}[\{a_1, a_2\}] := 1$.

We already showed in Example 5.5.10 that it holds that $a_2 \prec_{vio} a_1$ and that it does not hold that $a_1 \prec_{vio} a_2$. This means that a_1 and a_2 are not in a cyclic dependency such that they are in different conflict sets. Therefore $\mathsf{conflictSets} = \{\{a_1\},\{a_2\}\}$ in Line 12. The condition in Line 16 is satisfied for $\{a_2\}$ but not for $\{a_1\}$ such that action a_2 is classified with $\mathsf{cl}a_2 := 1$ in Line 17. The set $\{a_2\}$ is added to classSets in Line 18, and the set classSets is removed from conflictSets in Line 21. Hence, $\mathsf{conflictSets} := \{\{a_1\}\}$ in Line 21. In Line 22 the currently considered rank is increased such that $\mathit{rank}[\{a_1, a_2\}] := 2$.

The next execution of the inner repeat-until loop begins. The only remaining conflict set $\{a_1\}$ trivially satisfies the condition of Line 16 such that $\mathsf{cl}a_1 = 2$ and $\mathsf{conflictSets} := \emptyset$ in Line 21. The inner repeat-until loop terminates. In Line 25 $\mathsf{unclass} := \emptyset$ so that the outer repeat-until loop and thus the algorithm terminates. The output classification is

$$\mathsf{cl}a_1 = 2,\ \mathsf{cl}a_2 = 1,\ \mathsf{cl}a_3 = 0. \qquad \diamondsuit$$

We can show that the proposed algorithm terminates with a complete classification.

Proposition 5.5.15. If for all elementary operations of Procedure 5.5.1 algorithms that always terminate are given, then the algorithm always terminates and returns a complete classification.

Proof. See Appendix A.1.3, Page 247. □

Further, we can show that the secrecy reasoner implemented by our proposed algorithm satisfies the Principles I.1, I.2 and II, and thereby also Principles III and IV.

Proposition 5.5.16. Procedure 5.5.1 satisfies the Principles I.1, I.2 and II, as given in Definition 5.5.11.

Proof. See Appendix A.1.3, Page 250. □

In this and the previous two sections we developed declarative principles and an algorithm for the classification of actions with respect to secrecy under incomplete information. This way, we can generate a preference relation on actions with respect to secrecy, which is then used in combination with other preference relations that express the preferences on actions with respect to e. g., the utility with respect to the current goal of the agent. If the aggregation function of the preference relations is the lexicographic one we used before, then the agent violates secrecy only as little as possible from its perspective. Other aggregation operators might enable the agent to weigh the violation of secrecy of an action against the utility of it with respect to other goals such that it decides to violate secrets.

In the next section we develop secrecy aware agents based on the BDI model we developed in Section 3.5. The resulting agents then make use of the results of this and the previous two sections.

5.6 SECRECY BDI$^+$ AGENTS

In this section we combine the *basic BDI$^+$ agent model* as defined in Section 3.5.4 on Page 56 with the *secrecy agent model* presented in Section 5.2 and instantiate the resulting *basic BDI$^+$ secrecy agent model* by ASP as the knowledge representation formalism. Using ASP we model the background knowledge and instantiate the functional components of the communicating agents to adequately process and reason on the basis of exchanged speech acts. Hereby we include the representation of the meta-information of speech acts and model meta-inferences of agents based thereupon. The interaction of beliefs, motives and know-how is also modeled by use of ASP. Finally, we use these ASP representations to completely model the SCM Scenario that we used throughout this thesis.

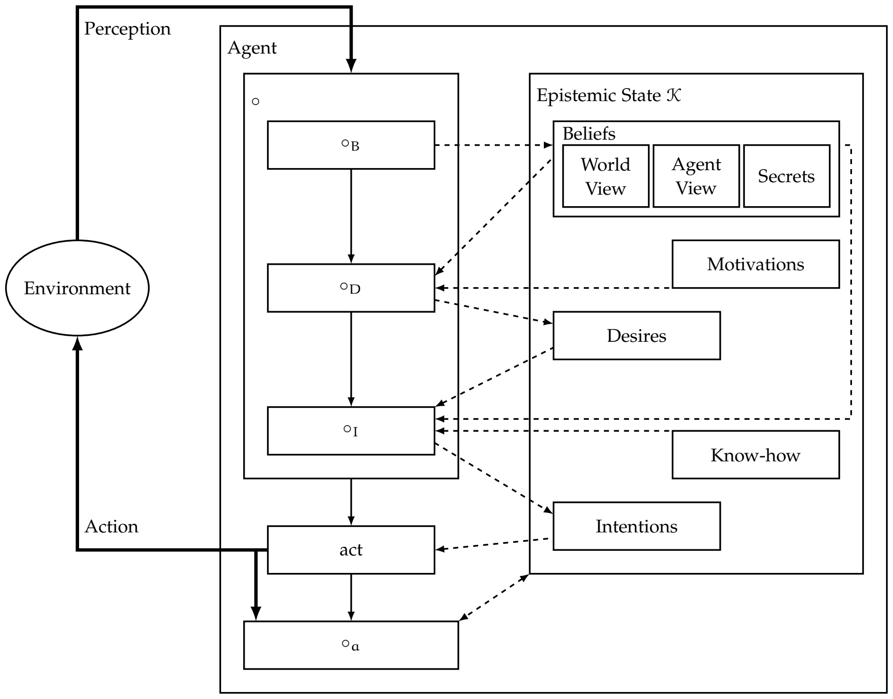


Figure 5.6.1: BDI$^+$ secrecy agent model and agent cycle

A basic *BDI$^+$ secrecy agent state* is a tuple $(\mathcal{K}^s_{\mathsf{BDI}^+,b}, \xi^s_{\mathsf{BDI}^+,b})$. It is a combination of the *basic BDI$^+$ state* presented in Section 3.5.3 and the *secrecy state* presented in Section 5.2. This means that the agent has all BDI$^+$ components, and its beliefs component consists of the secrecy components, i. e., world-view, agent-views, and the set of secrets. The BDI$^+$ secrecy agent model is illustrated in Figure 5.6.1 (Page 151), and formalized in the following. The epistemic state is a BDI$^+$ epistemic state

$$\mathcal{K}^s_{\mathsf{BDI}^+,b} = \langle \mathfrak{B}^s, \mathfrak{D}, \mathfrak{I}, \mathfrak{M}_b, \mathfrak{KH}_b \rangle \tag{5.6.1}$$

whose beliefs component contains the secrecy components required for a secrecy setting. That is, the agent's beliefs are of the form $\mathfrak{B}^s = \langle V_{\mathcal{D},W}, \mathcal{V}, \mathcal{S} \rangle$ with the world-view of $\mathcal{D}$, $V_{\mathcal{D},W} \subseteq \mathcal{L}_W$, a set of views on the world-views of all other agents, agent-views $\mathcal{V}$, and a set of secrets $\mathcal{S}$. Agent views are of the form $V_{\mathcal{D},\mathcal{X}}$, denoting that it is the view of agent $\mathcal{D}$ on the world-view of agent $\mathcal{X}$. The set of agent-views is then given by

$$\mathcal{V} = \{V_{\mathcal{D},\mathcal{X}} \in \mathcal{L}_V \mid \mathcal{X} \in \mathsf{Ag} \setminus \{\mathcal{D}\}\}. \tag{5.6.2}$$

In the two agent scenario we use for the presentation of our approach here, the set of agent-views has only one entry, the view of the defender on the other agent, the attacker. For this singleton set $\{V_{\mathcal{D},\mathcal{A}}\}$ we also write $V_{\mathcal{D},\mathcal{A}}$ in the following, by slight abuse of notation. The set of secrets is given by $\mathcal{S} \subseteq \mathcal{L}_S$.

We define the language of basic BDI$^+$ secrecy epistemic states as $\mathcal{L}^s_{\mathsf{BDI}^+,b}$. A functional basic BDI$^+$ component $\xi^s_{\mathsf{BDI}^+,b} = (\circ^s, \circ^{a,s}, \mathsf{act})$

consists of change operators $\circ^{s}$ and $\circ^{a,s}$, and an execute operator act of the types

$$\circ^{s} : \mathcal{L}^{s}_{\mathsf{BDI}^{+},b} \times \mathsf{Per} \to \mathcal{L}^{s}_{\mathsf{BDI}^{+},b},$$

$$\circ^{a,s} : \mathcal{L}^{s}_{\mathsf{BDI}^{+},b} \times \mathsf{Act} \to \mathcal{L}^{s}_{\mathsf{BDI}^{+},b},$$

$$\mathsf{act} : \mathcal{L}^{s}_{\mathsf{BDI}^{+},b} \to \mathsf{Act}.$$

In Section 3.1 we already defined the speech acts for the base language of literals, which is the base language in our ASP setting as well. Hence, we use the same speech acts as defined in Definition 3.1.2 (Page 38) that define the possible actions and percepts of an agent $\mathcal{X} \in \mathsf{Ag}$, these are:

$$\mathsf{Act}_{\mathcal{X}} = \{\langle \mathcal{X}, \mathcal{X}_r, \mathsf{type}, \mathsf{L}\rangle \mid \mathcal{X}_r \in \mathsf{Ag} \setminus \{\mathcal{X}\}, \mathsf{type} \in \mathsf{types}, \mathsf{L} \in \mathsf{Lit}\} \cup \epsilon. \quad (5.6.3)$$

$$\mathsf{Per}_{\mathcal{X}} = \{\langle \mathcal{X}_s, \mathcal{X}, \mathsf{type}, \mathsf{L}\rangle \mid \mathsf{type} \in \mathsf{types}, \mathsf{L} \in \mathsf{Lit}\} \cup p_{\epsilon}$$

with $\mathsf{type} \in \mathsf{types}_I \cup \mathsf{types}_R$ whereby $\mathsf{types}_I = \{\mathsf{inform}, \mathsf{answer}\}$ is the set of informative speech acts and $\mathsf{types}_R = \{\mathsf{query}\}$ the set of requesting speech acts. The symbol ϵ denotes the empty action and p_{ϵ} denotes the empty percept. The set of all speech acts as defined in (3.1.1) (Page 39) is:

$$\Sigma = \{\langle \mathcal{X}_s, \mathcal{X}_r, \mathsf{type}, \mathsf{L}\rangle \mid \mathcal{X}_s, \mathcal{X}_r \in \mathsf{Ag}, \mathsf{type} \in \mathsf{types}, \mathsf{L} \in \mathsf{Lit}\} \cup \{\epsilon, p_{\epsilon}\}.$$

Example 5.6.1. We recall our strike committee meeting example for speech acts in Section 3.1. In this, the agent *Beatriz* sends a speech act

$$\langle \mathit{Beatriz}, \mathit{Emma}, \mathsf{query}, \mathsf{attend\_scm}\rangle$$

to agent *Emma*. The answer of *Emma* is

$$\langle \mathit{Emma}, \mathit{Beatriz}, \mathsf{answer}, \neg\mathsf{attend\_scm}\rangle. \qquad \diamondsuit$$

An *ASP basic BDI secrecy agent state* $(\mathcal{K}^{b,s,asp}_{\mathsf{BDI}^{+}}, \xi^{b,s,asp}_{\mathsf{BDI}^{+}})$ is a *basic BDI secrecy agent state* that is based on ASP. In the following we consider $\mathit{ASPbasicBDIsecrecyagents}$, hence we leave out the indices of the components of an agent to ease readability. The agent-view and world-view languages are the same, the one of extended logic programs:

$$\mathcal{L}_V = \mathcal{L}_W = \mathcal{L}^{\mathsf{asp}}_{At}.$$

The secrets have literals as formulae and an ASP belief operator from the family $(\Xi^{\mathsf{asp}}, \preceq^{\mathsf{asp}}_{\mathsf{bel}})$, defined in ASP Instance 6 (Page 66) as

$$\Xi^{\mathsf{asp}} = \{\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}\},$$

$$\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}(P) = \cup AS(P) \text{ and } \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}(P) = \cap AS(P), \text{ and}$$

$$\preceq^{\mathsf{asp}}_{\mathsf{bel}} = \{(\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}), (\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}), (\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}})\}.$$

In the following we determine the sets of answer sets $AS(\mathsf{P})$ of a program P as an intermediate step to determine the belief set for one of the belief operators. If there is just one answer set, then both operators have the same result. The belief set language is $\mathcal{L}_{BS} = \mathsf{Lit}$. The possible secrets for agent $\mathcal{X}$ are then

$$\mathcal{L}_S = \mathcal{P}(\{(\mathsf{L}, \mathsf{Bel}, \mathcal{X}) \mid \mathsf{L} \in \mathsf{Lit}, \mathsf{Bel} \in \Xi^{\mathsf{asp}}, \mathcal{X} \in \mathsf{Ag} \setminus \mathcal{D}\}).$$

Further, the desires are of the form

$$\mathcal{L}_{\mathfrak{D}} \subseteq \mathcal{P}(\{(\mathsf{L}, \mu) \mid \mathsf{L} \in \mathsf{Lit}, \mu \in [0,1]\})$$

and the intentions are literals $\mathcal{L}_{\mathfrak{I}} \subseteq \mathcal{P}(\mathsf{Lit})$. The sets of percepts and actions are given by $\mathsf{Per}_{\mathcal{D}}$ and $\mathsf{Act}_{\mathcal{D}}$, respectively.

That is, in the ASP setting views are represented by extended logic programs, and desires and intentions are represented by literals. Note that by fixing these languages also the languages of the motivational structures $\mathfrak{M}$, Section 3.5.1, and the set of possible know-how bases $\mathfrak{KH}$, Section 3.5.2, are defined, since they are defined on the basis of the languages we just defined. In particular, a belief-desire coupling (D, Φ, μ) comprises a desire (D, μ) and a set of logical formulae $\Phi \subseteq \mathcal{L}_{BS}$. A know-how statement

$$(I, (s_1, \ldots, s_n), \{c_1, \ldots, c_m\}) \tag{5.6.4}$$

comprises a target goal I, a set of subgoals $(s_1, \ldots, s_n)$ and a set of conditions $\{c_1, \ldots, c_m\} \subseteq \mathcal{L}_{BS}$.

We illustrate the ASP instantiation by means of our *strike committee meeting* scenario in the following example.

Example 5.6.2. We start to model the *scm* scenario and concretize the formulation of the multiagent system we started in Example 3.3.2 (Page 43). The system contains the agent $\mathcal{E}$ for *Emma* and the agent $\mathcal{B}$ for *Beatriz* such that

$$\mathsf{Ag} = \{\mathcal{E}, \mathcal{B}\}.$$

The initial epistemic states are $\mathcal{K}^0_{\mathcal{E}}$ and $\mathcal{K}^0_{\mathcal{B}}$. We consider $\mathcal{K}^0_{\mathcal{E}}$ in more detail, it is:

$$\mathcal{K}^0_{\mathcal{E}} = \langle \{V^0_{\mathcal{E},W}, \mathcal{V}^0_{\mathcal{E},\mathcal{B}}, \mathcal{S}^0_{\mathcal{E}}\}, \mathfrak{D}^0_{\mathcal{E}}, \mathfrak{I}^0_{\mathcal{E}}, \mathfrak{M}_{\mathcal{E}}, \mathfrak{KH}_{\mathcal{E}} \rangle.$$

The superscript 0 is used to emphasize that the component is the initial version, it represents the components at time 0. The motivation and know-how components are static and therefore do not have this superscript.

Agent $\mathcal{E}$ assumes that $\mathcal{B}$ has the background knowledge $\mathsf{P}^{bg}_{\mathcal{B}} \subseteq V^0_{\mathcal{E},\mathcal{B}}$:

Predicate	Meaning
attend_work	*Emma* has to work the next day.
day_off	*Emma* gets the day_off by *Beatriz*.
attend_scm	*Emma* intends to attend the *strike committee meeting* the next day.
blacklist	*Emma* is on the blacklist of *Beatriz*.

Table 5.6.1: Representation of meta-information

$$
\begin{aligned}
P^{bg}_{\mathcal{B}} = \{ \quad & r_1 : \mathsf{attend\_work} && \leftarrow \mathsf{not}\ \neg\mathsf{attend\_work}.\\
& r_2 : \neg\mathsf{attend\_work} && \leftarrow \mathsf{day\_off}.\\
& r_3 : \mathsf{day\_off} && \leftarrow \neg\mathsf{attend\_scm}.\\
& r_4 : \mathsf{day\_off} && \leftarrow \mathsf{attend\_scm}.\\
& r_5 : \mathsf{blacklist} && \leftarrow \mathsf{attend\_scm}.\\
& r_6 : \mathsf{blacklist} && \leftarrow \mathsf{not}\ \mathsf{day\_off}, \neg\mathsf{attend\_work}.\}
\end{aligned}
$$

Table 5.6.1 lists the predicates of this program and their respective meanings. The program encodes that *Emma* normally goes to work (r_1) and does not go to work if she has a day off (r_2). She gets the day off only if she does not attend the *strike committee meeting* (r_3–r_4). If she attends the scm she is put on the blacklist (r_5). If she is absent without having the day_off she is also put on the blacklist (r_6). The set of answer sets of this program is $AS(P^{bg}_{\mathcal{B}}) = \{\{\mathsf{attend\_work}\}\}$.

Agent $\mathcal{E}$ has the following background knowledge $P^{bg}_{\mathcal{E}} \subseteq V^0_{\mathcal{E},W}$:

$$
\begin{aligned}
P^{bg}_{\mathcal{E}} = \{ \quad & r_1 : \mathsf{attend\_work} && \leftarrow \mathsf{not}\ \neg\mathsf{attend\_work}.\\
& r_2 : \neg\mathsf{attend\_work} && \leftarrow \mathsf{day\_off}.\\
& r_3 : \mathsf{attend\_scm}.\}
\end{aligned}
$$

The set of answer sets of this program is

$$AS(P_{\mathcal{E}}) = \{\{\mathsf{attend\_work},\ \mathsf{attend\_scm}\}\}.$$

The secret of *Emma* is given as

$$\mathcal{S}^0_{\mathcal{E}} = \{(\mathsf{attend\_scm}, \mathsf{Bel}^{asp}_{skep}, \mathcal{B})\}.$$

The initial sets of desires and intentions of agent $\mathcal{E}$ are empty

$$\mathfrak{D}^0_{\mathcal{E}} = \emptyset,\ \ \mathfrak{I}^0_{\mathcal{E}} = \emptyset.$$

We define the know-how base and the motives, here represented by motive-desire couplings, of the agents later on. We can note, that given the answer set of $\mathcal{E}$'s view on the world-view of $\mathcal{B}$ it holds that $\mathsf{attend\_scm} \notin \mathsf{Bel}^{asp}_{skep}(V_{\mathcal{E},\mathcal{B}})$ such that $\mathcal{K}_{\mathcal{E}}$ is safe. ♢

In the next sections we instantiate the functional component for ASP. First, we define the belief change operators, then the desire operator, and then the intention operator. After that, we define the complete functional component and we present the knowledge representation and attacker modeling for communicating ASP agents. Then, we show how all components play together in our model by showing that we can completely model our *strike committee meeting* example.

5.6.1 *ASP BDI Belief Change*

As defined in Section 3.5.3 (Page 54) the basic BDI change operator is realized by the sub-operators as illustrated in Figure 5.6.1 (Page 151). Formally the change operator for percepts is given as

$$\begin{aligned}&\langle \mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathfrak{M}_b, \mathfrak{K}\mathfrak{H}_b\rangle \circ^b_{\mathsf{BDI}^+} p = \\ &\quad\langle \mathfrak{B} \circ^{\mathsf{asp}}_{\mathfrak{B}} p, \\ &\qquad \mathfrak{D} \circ^b_{\mathfrak{D}} (\mathfrak{B} \circ^{\mathsf{asp}}_{\mathfrak{B}} p), \\ &\qquad\quad \circ^b_{\mathfrak{I}}(\mathfrak{I}, \mathfrak{B} \circ^{\mathsf{asp}}_{\mathfrak{B}} p, \mathfrak{D} \circ_{\mathfrak{D}} (\mathfrak{B} \circ^{\mathsf{asp}}_{\mathfrak{B}} p)), \\ &\qquad\qquad \mathfrak{M}_b, \mathfrak{K}\mathfrak{H}_b\rangle\end{aligned}$$

and the change operator for actions is given as

$$\langle \mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathfrak{M}_b, \mathfrak{K}\mathfrak{H}_b\rangle \circ^b_{a,\mathsf{BDI}^+} a = \langle \mathfrak{B} \circ^{\mathsf{asp}}_{a,\mathfrak{B}} a, \mathfrak{D}, \mathfrak{I}, \mathfrak{M}_b, \mathfrak{K}\mathfrak{H}_b\rangle.$$

The sub-operations $\circ^b_{\mathfrak{D}}$ and $\circ^b_{\mathfrak{I}}$ have already been defined in Section 3.5.4 (Page 56). We define the changes of the $\circ^{\mathsf{asp}}_{\mathfrak{B}}$ operator to the components of the beliefs component, in accordance with Equations (5.3.1) to (5.3.6) (Page 115). We realize the interpretation functions t_W, t^a_W, t_V and t^a_V for our ASP instance by a single interpretation function with three arguments

$$t^{\mathsf{asp}} : \mathsf{Ag} \times \mathsf{Ag} \times \Sigma \to \mathcal{L}^{\mathsf{asp}}_{At}.$$

It takes two agent identifiers from Ag and a speech act from Σ as input and returns an extended logic program. The extended logic program represents the interpretation of the speech act from the perspective of the view of the first agent on the world-view of the second agent. If the agent identifiers are identical the program represents the speech act for the world-view of that agent. Hence it is

$$\begin{aligned}t_W(p) &= t^{\mathsf{asp}}(\mathcal{D}, \mathcal{D}, p), & t^a_W(a) &= t^{\mathsf{asp}}(\mathcal{D}, \mathcal{D}, a), \\ t_V(p) &= t^{\mathsf{asp}}(\mathcal{D}, \mathcal{A}, p) \text{ and } & t^a_V(a) &= t^{\mathsf{asp}}(\mathcal{D}, \mathcal{A}, a).\end{aligned} \tag{5.6.5}$$

We realize the change operators of the agent by sub-operations, such that

$$\begin{aligned}(V_{\mathcal{D},W}, V_{\mathcal{D},\mathcal{A}}, \mathcal{S}) \circ^{\mathsf{asp}}_{\mathfrak{B}} p = \langle &V_{\mathcal{D},W} *^{\mathsf{asp}} t^{\mathsf{asp}}(\mathcal{D}, \mathcal{D}, p), \\ &V_{\mathcal{D},\mathcal{A}} *^{\mathsf{asp}} t^{\mathsf{asp}}(\mathcal{D}, \mathcal{A}, p), \\ &*_{\mathcal{S}}(\mathcal{S}, V_{\mathcal{D},W} *^{\mathsf{asp}} t^{\mathsf{asp}}(\mathcal{D}, \mathcal{D}, p), \\ &\qquad V_{\mathcal{D},\mathcal{A}} *^{\mathsf{asp}} t^{\mathsf{asp}}(\mathcal{D}, \mathcal{A}, p))\rangle\end{aligned} \tag{5.6.6}$$

$$\begin{aligned}(V_{\mathcal{D},\mathcal{W}}, V_{\mathcal{D},\mathcal{A}}, \mathcal{S}) \circ_{\mathfrak{B}}^{a,\mathsf{asp}} a = \langle & V_{\mathcal{D},\mathcal{W}} *^{\mathsf{asp}} t^{\mathsf{asp}}(\mathcal{D}, \mathcal{D}, a), \\ & V_{\mathcal{D},\mathcal{A}} *^{\mathsf{asp}} t^{\mathsf{asp}}(\mathcal{D}, \mathcal{A}, a), \\ & \mathcal{S}\rangle.\end{aligned}$$

The $*_{\mathcal{S}}$ operator is the operator defined in Equation 5.3.9 (Page 118). The operator $*^{\mathsf{asp}}$ is the multiple ASP base revision operator as presented in Section 4.3, Definition 4.3.22 (Page 96). Having defined the $*^{\mathsf{asp}}$ and the $*_{\mathcal{S}}$ operators, the interpretation function t^{asp} is the only operator that has to be instantiated to complete the instantiation of the belief change operators $\circ_{\mathfrak{B}}^{\mathsf{asp}}$ and $\circ_{\mathfrak{B}}^{a,\mathsf{asp}}$.

Before we come to the definition of t^{asp}, we elaborate an ASP representation of the information conveyed by speech acts. In accordance with (P1), it represents information on two levels: on the one hand the actual information, that is the informational content of the speech act; on the other hand the meta-information about the speech act that has been performed, which includes especially the information about the sender and the type of speech act and information revealed by these parameters. We distinguish these two levels of information available to the agent. The first is the information about the actual state of affairs, the *information-level*, and the second is the information about the communication with other agents, *meta-information-level*.

Example 5.6.3. The speech act

$$\langle \mathcal{E}, \mathcal{B}, \mathsf{answer}, \neg \mathsf{attend\_scm} \rangle$$

contains the information $\neg\mathsf{attend\_scm}$ expressing that $\mathcal{E}$ does not intend to attend the *strike committee meeting*. The meta-information of the speech act is the information about the action itself. That is, that $\mathcal{E}$ sent an answer to $\mathcal{B}$ with the answer value $\neg\mathsf{attend\_scm}$. The time at which the agent executed or perceived the speech also belongs to the meta-information of the speech act. It is not explicitly represented in the speech act, but the agent is aware of the time when it registers a percept and when it executes an action. ◇

Both levels have to be represented, evaluated and changed adequately. For the representation we use a modular ASP approach in which the representation of the information level and the meta-information level are formalized by ASP programs. Additional rules connect both levels and enable the agent to reason on the meta-information level, with effects on the information level. The representation of the *information-level* is straightforward since the base language are literals, which can be directly transformed into facts in a program. For the representation of the *meta-information-level* we introduce new literals that can be used to represent speech acts and the communication history. We denote the atoms for the representation of the meta-information level, the *meta-information level alphabet*, by $\mathsf{At}_{\mathsf{meta}}$ and the literals by

$\mathsf{Lit}_{\mathsf{meta}}$. We denote the set of all other literals as the *information level alphabet* $\mathsf{At}_{\mathsf{inf}} =_{def} \mathsf{At} \setminus \mathsf{At}_{\mathsf{meta}}$ and the literals by $\mathsf{Lit}_{\mathsf{inf}}$.

The interpretation function outputs the information contained in the considered ASP speech act on both levels. We define ASP literals representing that a speech act of type type has been received from some sender $\mathcal{X}_s$ with informational content L at time t. The time is assumed to be given in form of increasing natural numbers for each agent cycle in the multiagent system, as we introduced in Section 3.3. We use the literal $\mathsf{time}(t)$ to express that t is a point in time, it is added with the current time t to the epistemic component of the agent in each cycle. Only one agent cycle is executed at the same time such that the time stamp of each executed speech act is unique. We represent the, grounded, literal L as an argument of another literal, i. e., we *reify* the literals. For this we assume a function c that assigns a new and unique constant symbol to each grounded information level literal from $\mathrm{U}(\mathsf{Lit}_{\mathsf{inf}})$, e. g., we can use a literal L of the information level as an argument of the literal M of the meta-information level as $M(c(L))$.

This way the agent records the history of its interaction with other agents and takes this into consideration to draw inferences. We represent each speech act by a 4-ary literal $\mathsf{sa}(\cdot,\cdot,\cdot,\cdot)$ such that, each speech act of the form

$$\langle \mathcal{X}_s, \mathcal{X}_r, \mathsf{type}, L \rangle$$

is represented by the literal

$$\mathsf{sa}(\mathsf{type}, \mathcal{X}_s, \mathcal{X}_r, c(L), t).$$

Example 5.6.4. The first action in our SCM Example 3.3.2 (Page 43) at time 1,

$$\langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off} \rangle,$$

is thus represented by the literal

$$\mathsf{sa}(\mathsf{query}, \mathcal{E}, c(\mathsf{day\_off}), 1). \qquad \diamondsuit$$

In Table 5.6.2 we list all predicates for the representation of meta-information. There is a crucial difference of the semantics of the literal L depending on whether it is part of an informative or a requesting speech act. If it is part of an informative speech act, it is a statement that the literal L holds. If it is part of a requesting speech act, it is the request for the evaluation of the literal L, and no statement about the truth value of the literal is made. Hence, the interpretation of a speech act depends on whether it is a requesting or informative speech act. Further it depends on the fact if it is represented for the interpreting agent's world-view or for its view on the world-view of the other

Predicate	Meaning
$\mathsf{sa}(\mathsf{type}, \mathcal{X}_s, \mathcal{X}_r, c(L), t)$	A speech act of type type and information L was performed by $\mathcal{X}_s$, or received by $\mathcal{X}_r$, at time t
$\mathsf{refused}(\mathcal{D}, c(L))$	$\mathcal{D}$ refused to answer a query wrt. L
$\mathsf{time}(t)$	t is a time value
$\mathsf{at}(t)$	the current time is t
$\mathsf{has\_secret}(\mathcal{D}, c(L)$	The literal L is a secret formula of $\mathcal{D}$
$\mathsf{holds}(c(L))$	Literal L holds
$\mathsf{related}(c(L), c(L'))$	Literals L and L' are related
$\mathsf{open\_sens\_query}(\mathcal{A}, c(L))$	A query from $\mathcal{A}$ wrt. the sensitive formula L has not been answered
$\mathsf{open\_query}(\mathcal{A}, c(L))$	A query from $\mathcal{A}$ wrt. L has not been answered

Table 5.6.2: Representation of meta-information - At_{meta}

agent. And it depends on the fact if it is received or performed by the agent it is represented for.

The result of the interpretation function has three elements for percepts that are informative speech acts, i. e., $\mathsf{type} \in \mathsf{types}_I$. The first is the representation of the meta-information about the speech act, i. e., the predicate sa with the arguments representing the type, sender, literal and time of the speech act. The second element is the representation of the information-level, i. e., the literal contained in the speech act. The third element is the representation of the current time. For requesting speech acts, i. e., $\mathsf{type} \in \mathsf{types}_R$, the second element is left out, because a request is modeled not to contain information on the information-level. If the speech act under consideration is an action of $\mathcal{D}$ the second element is also left out for the world-view of $\mathcal{D}$. That is, $\mathcal{D}$ records that it sent the speech act, but does not need to revise its world-view by its information-level content. If agent $\mathcal{D}$ lies, it should know how it lied, but should not revise its own beliefs by its lie. If an agent is not among the receivers of a speech act, then only the representation of the current time is contained in the result of the interpretation. If the percept is the empty percept also only the representation of the current time is contained in the result of the interpretation.

The following definition of an interpretation function distinguishes these cases and generates the appropriate program for each of these.

Definition 5.6.5 (Interpretation Function). Let $\iota = \langle \mathcal{X}_s, \mathcal{X}_r, \mathsf{type}, \mathsf{L} \rangle$ and the current time be t. The interpretation function is defined as:

$$t^{asp}(\mathcal{X}, \mathcal{Y}, \iota) = \{\ \mathsf{sa}(\mathsf{type}, \mathcal{X}_s, \mathcal{X}_r, \mathsf{c}(\mathsf{L}), \mathsf{t}).,\ \mathsf{L}.,\ \mathsf{time}(\mathsf{t}).\} \qquad \text{if } \mathsf{type} \in \mathsf{types}_I,\ \mathcal{X} = \mathcal{Y} \text{ and } \mathcal{X} = \mathcal{X}_r$$

$$t^{asp}(\mathcal{X}, \mathcal{Y}, \iota) = \{\ \mathsf{sa}(\mathsf{type}, \mathcal{X}_s, \mathcal{X}_r, \mathsf{c}(\mathsf{L}), \mathsf{t}-1).,\ \mathsf{L}.,\ \mathsf{time}(\mathsf{t}-1).\} \qquad \text{if } \mathsf{type} \in \mathsf{types}_I \text{ and } \mathcal{X} \neq \mathcal{Y} \text{ and } \mathcal{Y} = \mathcal{X}_s$$

$$t^{asp}(\mathcal{X}, \mathcal{Y}, \iota) = \{\ \mathsf{sa}(\mathsf{type}, \mathcal{X}_s, \mathcal{X}_r, \mathsf{c}(\mathsf{L}), \mathsf{t}+1).,\ \mathsf{L}.,\ \mathsf{time}(\mathsf{t}+1).\} \qquad \text{if } \mathsf{type} \in \mathsf{types}_I \text{ and } \mathcal{X} \neq \mathcal{Y} \text{ and } \mathcal{X} = \mathcal{X}_s$$

$$t^{asp}(\mathcal{X}, \mathcal{Y}, \iota) = \{\ \mathsf{sa}(\mathsf{type}, \mathcal{X}_s, \mathcal{X}_r, \mathsf{c}(\mathsf{L}), \mathsf{t}).,\ \mathsf{time}(\mathsf{t}).\} \qquad \text{if } \mathcal{X} = \mathcal{Y} \text{ and } (\mathsf{type} \in \mathsf{types}_R \text{ or } (\mathsf{type} \in \mathsf{types}_I \text{ and } \mathcal{X} = \mathcal{X}_s)$$

$$t^{asp}(\mathcal{X}, \mathcal{Y}, \iota) = \{\ \mathsf{sa}(\mathsf{type}, \mathcal{X}_s, \mathcal{X}_r, \mathsf{c}(\mathsf{L}), \mathsf{t}-1).,\ \mathsf{time}(\mathsf{t}-1).\} \qquad \text{if } \mathsf{type} \in \mathsf{types}_R,\ \mathcal{X} \neq \mathcal{Y} \text{ and } \mathcal{Y} = \mathcal{X}_s$$

$$t^{asp}(\mathcal{X}, \mathcal{Y}, \iota) = \{\ \mathsf{sa}(\mathsf{type}, \mathcal{X}_s, \mathcal{X}_r, \mathsf{c}(\mathsf{L}), \mathsf{t}+1).,\ \mathsf{time}(\mathsf{t}+1).\} \qquad \text{if } \mathsf{type} \in \mathsf{types}_R,\ \mathcal{X} \neq \mathcal{Y} \text{ and } \mathcal{X} = \mathcal{X}_s$$

$$t^{asp}(\mathcal{X}, \mathcal{Y}, \iota) = \{\ \mathsf{time}(\mathsf{t}).\} \qquad \text{else}$$

We call $\mathcal{X}$ the *interpreting agent*, and $\mathcal{Y}$ the *other agent* in the following. The first case is the case of the interpretation of an informative speech act, $\mathsf{type} \in \mathsf{types}_I$, for the world-view of the interpreting agent, $\mathcal{X} = \mathcal{Y}$, as a percept, $\mathcal{X} = \mathcal{X}_r$. The result is a program comprising the representation of the meta-information about the speech act $\mathsf{sa}(\mathsf{type}, \mathcal{X}_s, \mathcal{X}_r, \mathsf{c}(\mathsf{L}), \mathsf{t}).$, the representation of the logical content of the speech act L., and the representation of the current time $\mathsf{time}(\mathsf{t}).$.

The second case is also for an informative speech act, but for the view on the world-view of the sender of the speech act, which is the other agent since $\mathcal{Y} = \mathcal{X}_r$. The difference to the first case is that the speech act has been sent at time $\mathsf{t}-1$ by the other agent such that it is represented with this time in the agent-view. The logical content of the speech act is also represented, which means that the receiving agent assumes that the sender believes the information it sent itself. Since the sender is assumed to use a skeptical belief operator, believing a literal L means that it is in all answer sets. This is the case if the fact L. is added to the view on the sender.

The third case is for an informative speech act that is sent by the agent and is represented for the agent-view on the other agent as a percept. The speech act is received, and represented, by the other agent at time $t+1$, and also represented with this time in the agent-view on the other agent.

The fourth case represents three different cases which all lead to the same representation of the speech act. All have in common that the representation is for the world-view of the interpreting agent $\mathcal{X}$, and that the logical information of the speech act is not contained in the representation. The logical information of a speech act is not represented if

a) it is a requesting speech act and the interpreting agent is receiver or sender: requests do not inform about, but query for the literal they contain.

b) it is an informative speech act and the interpreting agent is the sender of it: an agent only keeps a record of the meta-information of the actions it performed.

The fifth case is for a requesting speech act, sent by the other agent, represented for the view on the other agent. As in the previous case requesting speech acts are represented by their meta-information and the representation of the time. The represented time is the time when the speech act was sent.

The sixth case is for a requesting speech act sent by the agent, represented for the view on the other agent. The represented time is the time the other agent receives the speech act.

The interpretation function still has to be defined for the empty action and the empty percept. For the empty action ϵ it is

$$t^{asp}(\mathcal{X},\mathcal{Y},\epsilon) = \begin{cases} \{\mathsf{time(t).}\} & \text{if } \mathcal{X} = \mathcal{Y} \\ \{\mathsf{time(t+1).}\} & \text{if } \mathcal{X} \neq \mathcal{Y} \end{cases}$$

and for the empty percept p_ϵ it is

$$t^{asp}(\mathcal{X},\mathcal{Y},p_\epsilon) = \begin{cases} \{\mathsf{time(t).}\} & \text{if } \mathcal{X} = \mathcal{Y} \\ \{\mathsf{time(t-1).}\} & \text{if } \mathcal{X} \neq \mathcal{Y} \end{cases}$$

That is, for the empty action and the empty percept the agent only notes that the time has passed. As for before, for its view on the world-view of the other agent the agent records the time as the other agent records it. The next example illustrates the definition of the interpretation function.

Example 5.6.6. Consider the speech act $\langle \mathcal{E},\mathcal{B},\mathsf{query},\mathsf{day\_off}\rangle$ from Example 5.6.4, which is an action of $\mathcal{E}$. In the belief change process of $\mathcal{E}$ it is used as input for the interpretation function for its world-view:

$$t^a_W(\langle \mathcal{E},\mathcal{B},\mathsf{query},\mathsf{day\_off}\rangle) = t^{asp}(\mathcal{E},\mathcal{E},\langle \mathcal{E},\mathcal{B},\mathsf{query},\mathsf{day\_off}\rangle) = \\ \{\mathsf{sa(query},\mathcal{E},\mathcal{B},\mathsf{c(day\_off),1).},\ \mathsf{time(1).}\}.$$

The same speech act is also the input for the interpretation function for the agent-view on agent $\mathcal{B}$:

$$t_V^a(\langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off} \rangle) = t^{asp}(\mathcal{E}, \mathcal{B}, \langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off} \rangle) = \{\mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off}), 2)., \ \mathsf{time}(2).\}$$

Agent $\mathcal{B}$ receives the speech act as its percept in its next agent cycle, at time 2. The agent's interpretation function for its world-view is given as:

$$t_W(\langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off} \rangle) = t^{asp}(\mathcal{B}, \mathcal{B}, \langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off} \rangle) = \{\mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off}), 2)., \ \mathsf{time}(2).\}$$

And for the agent-view on agent $\mathcal{E}$ the interpretation function is given as:

$$t_V(\langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off} \rangle) = t^{asp}(\mathcal{B}, \mathcal{E}, \langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off} \rangle) = \{\mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off}), 1)., \ \mathsf{time}(1).\}. \qquad \diamondsuit$$

Up to this point we have defined all main components of the belief change operators $\circ_{\mathfrak{B}}^{asp}$ and $\circ_{\mathfrak{B}}^{a,asp}$, as defined in (5.6.6). These are the ASP revision operator $*^{asp}$, the change operator for secrets $*_{\mathcal{S}}$, and the interpretation function t^{asp}. We now turn to the definition of the desire operator in the next section.

5.6.2 *Secrecy desire operator*

The *secrecy desire operator* $\circ_{\mathfrak{D},s}$ uses the belief-desire couplings to determine the current set of desires in the light of the new beliefs just as the *basic desire operator* we defined in Equation (3.5.3) (Page 57). The only difference in the *secrecy* BDI$^+$ model is that the beliefs component of the agent is compound, it is $\mathfrak{B} = (V_{\mathcal{D},W}, V_{\mathcal{D},\mathcal{A}}, \mathcal{S})$. The conditions of the belief-desire couplings have to be satisfied by the world-view of the agent, instead of the beliefs component:

$$\circ_{\mathfrak{D},s}(\mathfrak{M}_b, (V_{\mathcal{D},W}, V_{\mathcal{D},\mathcal{A}}, \mathcal{S})) =_{def} \qquad (5.6.7)$$

$$\{(D, \mu) \mid (D, \Phi, \mu) \in \mathfrak{M}_b, \Phi \subseteq \mathsf{Bel}_{\mathcal{D}}(V_{\mathcal{D},W})\}$$

Similarly the intention operator has to be adapted for the secrecy agent model, which we do in the next section.

5.6.3 *Generation of Options*

We defined the intention operator in Procedure 3.5.1 (Page 60), independently of the knowledge representation formalism. For the knowledge representation specific parts of the intention operator we have to define the set of possible intentions, the know-how of the agent

Predicate	Meaning
$\mathsf{informed}(\mathcal{X}, \mathsf{c}(\mathsf{L}))$	Intention to inform $\mathcal{X}$ that L holds
$\mathsf{answered}(\mathcal{X}, \mathsf{c}(\mathsf{L}))$	Intention to send the answer that L holds to $\mathcal{X}$
$\mathsf{queried}(\mathcal{X}, \mathsf{c}(\mathsf{L}))$	Intention to send a query for L to $\mathcal{X}$
$\mathsf{sacted}(\mathsf{type}, \mathcal{X}_r, \mathsf{c}(\mathsf{L})))$	Intention to perform speech act of type type, receiver $\mathcal{X}_r$ and information L
not_acted	Intention to not perform any action

Table 5.6.3: Considered intentions and their intuitive meanings

and the evaluation function. Further we have to slightly adapt the *basic options operator* opt_b, defined in Equation (3.5.4) (Page 58), to the *secrecy* BDI^+ model.

The *basic options operator* uses a basic know-how base to determine the current options and is of the form

$$\mathsf{opt}_b : \mathcal{L}_{\mathfrak{B}} \times \mathcal{L}_{\mathfrak{I}} \times \mathcal{L}_{\mathfrak{KH},b} \to \mathcal{L}_{\mathfrak{I}}.$$

It determines all subgoals that are contained in a know-how statement for the agent's current intention, if the conditions of the respective know-how statement are satisfied. For the use of this operator with respect to the *secrecy* BDI^+ model with a compound beliefs component $\mathfrak{B} = (V_{\mathcal{D},W}, V_{\mathcal{D},\mathcal{A}}, \mathcal{S})$ the conditions of the know-how statements have to be satisfied by the world-view $V_{\mathcal{D},W}$:

$$\begin{aligned} \mathsf{opt}_s(\mathfrak{B}, \mathfrak{I}, \mathfrak{KH}_b) = \{ s \mid & (I, (s), \{c_1, \ldots, c_m\}) \in \mathfrak{KH}_b, \\ & I \in \mathfrak{I}, \{c_1, \ldots, c_m\} \subseteq \mathsf{Bel}_{\mathcal{D}}(V_{\mathcal{D},W})\}. \end{aligned} \tag{5.6.8}$$

Having defined the opt_s operator we continue with the definition of the set of possible intentions. An intention is of the form of a goal, which means that it characterizes a feature of a state that the agent aspires to make true. An example of such a property of a state is that $\mathsf{informed}(\mathcal{X}, \mathsf{c})$ holds in its belief set, meaning that it has informed agent $\mathcal{X}$ about the literal that is represented by the constant c. All intentions that we consider in the following are described in Table 5.6.3. Each intention can be instantiated for each agent identifier and for each constant symbol representing a grounded literal from $\mathsf{U}(\mathsf{Lit}_{\mathsf{inf}})$, cf. Section 2.4 (Page 25). This results in the following set of possible intentions of an agent:

$$\begin{aligned} \mathfrak{I}^b = \; & \{ \mathsf{answered}(\mathcal{X}, \mathsf{c}), \mathsf{informed}(\mathcal{X}, \mathsf{c}), \mathsf{queried}(\mathcal{X}, \mathsf{c}) \\ & \qquad \mid \mathcal{X} \in \mathsf{Ag} \text{ and } \mathsf{c} \in \bigcup \{ \mathsf{c}(\mathsf{L}) \mid \mathsf{L} \in \mathsf{U}(\mathsf{Lit}_{\mathsf{inf}}) \} \} \\ & \cup \{ \mathsf{sacted}(\mathsf{type}, \mathcal{X}, \mathsf{c}) \mid \mathsf{type} \in \mathsf{types}_I \cup \mathsf{types}_R, \\ & \qquad \mathcal{X} \in \mathsf{Ag} \text{ and } \mathsf{c} \in \bigcup \{ \mathsf{c}(\mathsf{L}) \mid \mathsf{L} \in \mathsf{U}(\mathsf{Lit}_{\mathsf{inf}}) \} \} \\ & \cup \{ \mathsf{not\_acted} \} \end{aligned}$$

The intentions of the type sacted and the intention not_acted are atomic intentions and the corresponding actions are given by the function α, introduced in Section 3.5.3 (Page 54), as follows (assuming that we model agent $\mathcal{D}$):

$$\alpha(\mathsf{sacted}(\mathsf{type}, \mathcal{X}, \mathsf{c}(\mathsf{L}))) = \langle \mathcal{D}, \mathcal{X}, \mathsf{type}, \mathsf{L} \rangle$$
$$\alpha(\mathsf{not\_acted}) = \epsilon$$

Example 5.6.7. In Equation (3.5.8) (Page 59) we defined the action function of our agents as

$$\mathsf{act}(\mathcal{K}) =_{def} \alpha(I).$$

We consider the first cycle of agent $\mathcal{E}$ in Example 3.3.2 (Page 43). The agent chooses to satisfy the atomic intention

$$\mathsf{sacted}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off})),$$

such that its action is given by

$$\alpha(\mathsf{sacted}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off}))) = \langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off} \rangle. \qquad \diamondsuit$$

After defining the possible intentions of an agent, we define the know-how of an agent on how to achieve these intentions. An agent can have the intention that it answers a query of agent $\mathcal{X}$, that it informs $\mathcal{X}$, or that it queries $\mathcal{X}$ with respect to a literal represented by the constant c. For these intentions we define a *basic know-how base*, as introduced in Section 3.5.2 (Page 3.5.2), as follows:

$$\begin{aligned}
\mathfrak{KH}_b^{asp} = \{ \; & (\mathsf{informed}(\mathcal{X}, \mathsf{c}(\mathsf{L})), (\mathsf{sacted}(\mathsf{informed}, \mathcal{X}, \mathsf{c}(\mathsf{L}))), \emptyset); \qquad (5.6.9)\\
& (\mathsf{informed}(\mathcal{X}, \mathsf{c}(\mathsf{L})), (\mathsf{not\_acted}), \emptyset);\\
\\
& (\mathsf{answered}(\mathcal{X}, \mathsf{c}(\mathsf{L})), (\mathsf{sacted}(\mathsf{answer}, \mathcal{X}, \mathsf{c}(\mathsf{L}))), \emptyset);\\
& (\mathsf{answered}(\mathcal{X}, \mathsf{c}(\mathsf{L})), (\mathsf{sacted}(\mathsf{informed}, \mathcal{X}, \mathsf{c}(\neg\mathsf{L}))), \emptyset);\\
& (\mathsf{answered}(\mathcal{X}, \mathsf{c}(\mathsf{L})), (\mathsf{not\_acted}), \emptyset);\\
\\
& (\mathsf{queried}(\mathcal{X}, \mathsf{c}(\mathsf{L})), \quad (\mathsf{sacted}(\mathsf{query}, \mathcal{X}, \mathsf{c}(\mathsf{L}))), \emptyset);\\
& (\mathsf{queried}(\mathcal{X}, \mathsf{c}(\mathsf{L})), \quad (\mathsf{not\_acted}), \emptyset)\\
\\
& (\mathsf{not\_acted}, \qquad (\mathsf{not\_acted}), \emptyset)\\
\\
& \mid \mathsf{L} \in \mathcal{U}(\mathsf{Lit_{inf}}) \; \}
\end{aligned}$$

In the first line the know-how base expresses that the agent can inform agent $\mathcal{X}$ by performing a corresponding inform speech act. It can do this in all situations, which is expressed by the empty set being the condition of the know-how statement. Alternatively it can execute the empty action.

Note that, through the evaluation of the empty action as an option we implement a method for the reconsideration of intentions [211].

If the agent decides that the empty action is the best option, it can be seen as dropping the corresponding intention. However, in the secrecy setting we consider, "performing" the empty action can have effects as well due to meta-infererences of $\mathcal{A}$. Hence from the secrecy perspective it is adequate to consider it as an option for action. This also makes sense from the perspective of intention reconsideration since the evaluation of the empty action is assumed to consider the general utility of it in comparison to other actions, just as in other approaches to intention reconsideration.

The know-how base gives the agent three alternatives to answer a query for literal L. It can answer with L, it can answer with $\neg L$, or it can perform the empty action. Note that one of the first two options resembles a truthful answer and the other one a lie, and that the last option resembles a refusal to answer. For the realization of the intention to query for a literal, the agent can perform the corresponding *query* speech act, or it can perform the empty action.

The eval function generates and aggregates the two preference relations on actions $\preceq^{s}_{(\mathcal{K},\mathcal{E})}$ and $\preceq^{a}_{\mathcal{K}}$. The $\preceq^{s}_{(\mathcal{K},\mathcal{E})}$ relation is based on a classification of actions as defined in (5.5.10) (Page 140). The classification is generated by a secrecy reasoner (Definition 5.5.11, Page 144) that is implemented by the algorithm presented in Procedure 25 (Page 148). The algorithm relies on the notion of violation sets which we defined in Definition 5.5.4, Page 135. Here, by slight abuse of notation, we define these for the ASP instance as follows. Let $\mathcal{S}$ be a set of active secrets, $V_{\mathcal{D},\mathcal{A}}$ an agent-view and a an action, then

$$\mathsf{vioAfter}(\mathcal{S}, V_{\mathcal{D},\mathcal{A}}, a) = \{\mathsf{vio}(\mathcal{S}, V_{\mathcal{D},\mathcal{A}} *^{\mathsf{asp}} t^{\mathsf{asp}}(\mathcal{D}, \mathcal{A}, a))\} \tag{5.6.10}$$

with

$$\mathsf{vio}(\mathcal{S}, V_W) = \{(\phi, \mathsf{Bel}, \mathcal{A}) \in \mathcal{S} \mid \phi \in \mathsf{Bel}(V_W)\}.$$

The difference to the previous formulation is that we consider agent-views here that are single world-views and not sets of world-views as considered before. Further, we instantiated the belief change operator for the agent-view with the ASP belief change operator for actions that we use for our ASP BDI instance.

The other preference relation, $\preceq^{a}_{\mathcal{K}}$, is based on the informativity of the actions as defined in (5.5.5) (Page 132). In our causes agents to be as informative as possible while protecting their secrets. Further, it is an example of another criterion for the evaluation of actions in our compound preference relation on actions. For the aggregation we use the lexicographic aggregation operator (defined in (5.5.2), Page 131) with the order giving priority to secrecy over informativity, i. e.,

$$f^{\mathsf{lex}}_{agg}((\preceq^{s}_{(\mathcal{K},\mathcal{E})}, \preceq^{a}_{\mathcal{K}})).$$

Hence, we know that $\preceq_{(\mathcal{K},\mathcal{E})}$ satisfies *Safe Preference* and that among the action with the same degree of violation preference is given to more informative actions.

In the following section we model meta-inferences on the basis of the representation of meta-information we developed in Section 5.6.1.

5.6.4 *Modeling meta-inferences*

We already introduced the two levels of information we model in Section 5.6.1. The meta-information level allows the agent to keep a record of the exchanged speech acts. This allows the agent to reason about the communication and to draw meta-inferences from it. On the information level the beliefs of the agent are expressed by means of literals. On the meta-information level these literals are expressed as constants that are the arguments of literals that express the meta-information. We further define rules that connect these two levels by inferring beliefs of the agent from meta-information. This way we model such patterns of meta-level reasoning for communicating agents.

Meta-inferences allow an agent to determine conspicuous behavior of another agent. We define ASP rules in the following that are used to define programs that represent a specific pattern of meta-inference and can be modularly added to a view on an attacker $\mathcal{A}$. These rules are denoted by r_i^m with the superscript m denoting a meta-inference rule and the index i being a natural number.

The most common form of conspicuous behavior are refusals or deflections.

Example 5.6.8. If *Emma* does not answer the query of *Beatriz* if she intends to attend the *strike committee meeting* and starts talking about the weather *Beatriz* will notice that *Emma* does not want to answer the question. From this *Beatriz* might draw the meta-inference that *Emma* does intend to attend the *strike committee meeting*. ♢

We formulate a rule with a head literal representing that an agent $\mathcal{D}$ did not reply to a requesting speech act sent to it by $\mathcal{A}$. For this purpose we have to define which behavior of $\mathcal{D}$ is considered as an answer to a given request. We do this by introducing the literal $\mathsf{related}(\cdot,\cdot)$ that represents that two literals are related. Here, we use the $\mathsf{related}$ predicate to determine if the literal of an answer is related to the literal that was asked for. In our scenario valid answers to a query are to answer that the queried literal is true or false, expressed by sending an answer with the queried literal or with the negation of it. Hence we consider two literals L and L' related if and only if they represent values for the same atom, i. e., if and only if $L = \neg L'$ or $L = L'$.

Example 5.6.9. Answering that the weather is fine is not related to the query if *Emma* intends to attend the *strike committee meeting*. ♢

We assume that any *inform* speech act containing a literal that is related to the literal of the preceding request is seen as an answer to that request. We model that an agent determines that the answer to a query has been refused if it has not received an answer related to the query within a defined response time $t_{\mathsf{resp}} \in \mathbb{N}$. In the following we consider a response time of $t_{\mathsf{resp}} = 2$. That is, if $\mathcal{A}$ sent a query at time t_1 and at time $t_1 + t_{\mathsf{resp}}$ it has not received an answer from $\mathcal{D}$, then $\mathcal{A}$ assumes that the request is not going to be answered. This is modeled in the following rule schema in which we assume V, V' to be variables representing constants that represent literals on the information level, and that T_1 and T_2 are variables representing natural numbers that represent time. Remember that variable names start with an upper-case letter and constants with a lower-case letter.

$$\begin{aligned} r_1^m : \mathsf{refused}(\mathcal{D}, V) \leftarrow\ & \mathsf{sa}(\mathsf{query}, \mathcal{A}, \mathcal{D}, V, T_1), \\ & \mathsf{not}\ \mathsf{sa}(\mathsf{answer}, \mathcal{D}, \mathcal{A}, V', T_2), \\ & \mathsf{at}(T_2), \mathsf{related}(V, V'), T_2 \geqslant T_1 + t_{\mathsf{resp}}. \end{aligned}$$

Having determined that a request has been refused by $\mathcal{D}$, agent $\mathcal{A}$ can base inferences on this. We call an agent $\mathcal{A}$ a *secret aware attacker* if it knows which are the potentially secret formulae of $\mathcal{D}$. That it, $\mathcal{A}$ knows the formulae but not if $\mathcal{D}$ beliefs them or not, i. e., it does not know which secrets are active. We represent the potentially secret formulae by meta-level literals $\mathsf{has\_secret}(\mathcal{D}, \mathsf{c}(L))$, cf. Table 5.6.2. We define a program for a given set of secrets containing the corresponding potentially secret formulae as facts. This program is formalized as

$$P_{\mathcal{S}}(\mathcal{S}(\mathcal{K}_{\mathcal{D}})) = \{\mathsf{has\_secret}(\mathcal{D}\ ,\ \mathsf{c}(L)). \mid (L, \mathsf{Bel}, \mathcal{A}) \in \mathcal{S}(\mathcal{K}_{\mathcal{D}})\}.$$

We consider a *secret aware attacker* $\mathcal{A}$ that believes that $\mathcal{D}$ has a secret with the potentially secret formula L. $\mathcal{A}$ does not know if L holds, and recognizes that $\mathcal{D}$ refuses a request with respect to L. On the basis of this behavior $\mathcal{A}$ infers that L holds. We formalize this meta-inference with the help of the following rule:

$$r_2^m : \mathsf{holds}(V) \leftarrow \mathsf{has\_secret}(\mathcal{D}, V), \mathsf{refused}(\mathcal{D}, V).$$

We call an attacker that infers that a potentially secret formula holds if a query for it has been refused *refusal sensitive*.

Example 5.6.10. If *Beatriz* realizes that *Emma* does not want to answer the question if she intends to attend the *strike committee meeting*, then she believes that *Emma* does intend to attend the *strike committee meeting*. ♢

Hence, for agents with attackers that are secret aware and refusal sensitive there are cases in which not answering violates secrecy. This means in particular that the empty action is not always secrecy preserving.

We have described the knowledge representation on two levels, the information and the meta-information level, for which we introduced the two alphabets At_{inf} and At_{meta}. In the following we describe how we connect both levels to allow for meta-inferences, i. e., to allow inferences on the meta-information level to lead to implications on the information level. On the meta-information level a literal L is represented by means of a constant symbol $c(L)$. That this literal holds is represented by the meta-information level literal $\mathsf{holds}(c(L))$. By means of Rule r_3^m the literal $\mathsf{holds}(c(L))$ is inferred. This is an inference on the meta-information level, but we want the attacking agent to infer information on the information-level from its meta-information. We create a connection from the meta-information level to the information level by defining corresponding rules that form the program P_{link}. This program also formalizes the *related* concept we consider here by defining two literal constants to be related if and only if they are complements of each other or equal. The program is formalized for a given set of information level atoms At_{inf} as follows:

$$
\begin{aligned}
P_{\mathsf{link}}(At_{\mathsf{inf}}) = \{\ & A && \leftarrow \mathsf{holds}(c(A)).; \\
& \neg A && \leftarrow \mathsf{holds}(c(\neg A)).; \\
& \mathsf{related}(c(A), c(\neg A)). && \qquad\qquad \mid A \in At_{\mathsf{inf}}\} \\
\\
\cup\ \{ & \mathsf{related}(V', V) && \leftarrow \mathsf{related}(V, V').; \\
& \mathsf{related}(V, V) && \leftarrow \mathsf{related}(V, V').; \\
& \mathsf{at}(T) && \leftarrow \mathsf{time}(T), \mathsf{not}\ \mathsf{time}(T'), T < T'.\}
\end{aligned}
$$

The program consists of the union of two sets, the first set is generated for all given atoms, the second consists of rule schemata, i. e., rules with variables. With the first two rules of the program directly connects the representation that an information level literal holds on the meta-information level to the fact that the literal holds on the information level. The third rule adds the meta-information level literal $\mathsf{related}(c(A), c(\neg A))$ for all atoms of the information level. The first two rule schemata represent the symmetric and reflexive closure of the related concept. The last rule schema determines the current time as the maximal time point.

We define programs based on the meta level rules and programs defined above. We then use these defined programs to define properties of attacker models.

Definition 5.6.11. Given an ASP agent $\mathcal{D}$ with initial epistemic state $\mathcal{K}_{\mathcal{D}}^0$ and a given set of atoms At. We first define the following programs:

– $P_{\mathcal{S}\text{-aware}}^{\mathsf{meta}} = P_{\mathsf{link}}(At) \cup P_{\mathcal{S}}(\mathcal{S}(\mathcal{K}_{\mathcal{D}}^0))$

– $P^{\mathsf{meta}}_{\mathsf{R\text{-}sensitive}} = \{r^m_1, r^m_2\}$.

An attacker modeling $V_{\mathcal{A}}(\mathcal{K}^0_{\mathcal{D}})$ is

– *secret aware* if $P^{\mathsf{meta}}_{\mathcal{S}\text{-}\mathsf{aware}} \subseteq V_{\mathcal{A}}(\mathcal{K}^0_{\mathcal{D}})$,

– *refusal sensitive* if $P^{\mathsf{meta}}_{\mathcal{S}\text{-}\mathsf{aware}} \cup P^{\mathsf{meta}}_{\mathsf{R\text{-}sensitive}} \subseteq V_{\mathcal{A}}(\mathcal{K}^0_{\mathcal{D}})$.

In the following we consider an attacking agent that is *refusal sensitive* and *secret aware* and attacker models of $\mathcal{D}$ that reflect this. Then $\mathcal{D}$ has to reply to queries for sensitive formulae from $\mathcal{A}$ in order to be secrecy preserving; otherwise $\mathcal{A}$ infers the secret formula by means of the meta-reasoning introduced above.

We showed how we can model meta-inference by means of ASP. In the following we focus on the meta-inference on the basis of sensitive queries. In particular we make the following assumptions:

1. the empty action is always safe, except the defender is expected to answer a query with respect to some sensitive formula.

2. In the latter case lying by answering the negation of the secret formula is safe.

These assumptions guarantee that there is always a safe action, such that the setting we consider is *plain*.

In the next section we consider the generation and selection of desires and options of our agents with respect to secrecy and for our considered setting.

5.6.5 *Deliberation and secrecy preservation*

We have shown in Proposition 5.5.2 (Page 133) that an agent has to consider a subset of all possible actions $\mathsf{Act}'_{(\mathcal{K},\mathcal{E})} \subseteq \mathsf{Act}$ as its options that contains a secrecy preserving action in order to be secrecy preserving. This was formalized by the following condition of Proposition 5.5.2:

2. if there is a sound action $a \in \mathsf{Act}$,
   then there is a sound action $a' \in \mathsf{Act}'_{(\mathcal{K},\mathcal{E})}$.

From the assumptions 1. and 2. above follows that the agents we consider here satisfy this condition if

a) they always consider the empty action as an option,

b) in case they have received a query with respect to a sensitive formula they consider to answer this query for all possible evaluations of the formula.

We show next how this can be achieved.

In general, a BDI agent performs actions that contribute towards the satisfaction of its goals. Hence, to respond to a query with an appropriate answer the agent has to have the goal to answer the query and then it has to decide to satisfy the goal by means of one of the appropriate answers. This means that secrecy has to be considered not only in the option generation but also in the desire generation.

For our basic BDI$^+$ agents, defined in Section 3.5.4 (Page 56), this means that the agent has to generate the desire to answer the query by means of its motivation component. Moreover, this desire has to be selected as the agent's current intention such that the options for its realization are considered. This is achieved by guaranteeing that the desire is the one with the highest motivational value. We guarantee this by requiring that the defending agent's motivations contain the following belief-desire coupling

$$(\mathsf{answered}(\mathcal{A}, V), \mathsf{open\_sens\_query}(\mathcal{A}, V), 1),$$

and by requiring that no other belief-desire coupling with a motivational value of 1 exists. The belief desire coupling contains the condition

$$\mathsf{open\_sens\_query}(\mathcal{A}, V).$$

We define meta-inference rules from the meta-information about the received queries in the world-view of $\mathcal{D}$. It is inferred if $\mathcal{D}$ has received a query with respect to a sensitive formula, and has not yet answered it. This is realized by assuming that the world-view of $\mathcal{D}$ contains the program $P_{\mathcal{S}}(\mathcal{S}(\mathcal{K}))$ and the rules:

$$\begin{aligned} r_3^m : \mathsf{open\_sens\_query}(\mathcal{A}, V) \leftarrow\ & \mathsf{open\_query}(\mathcal{A}, V), \\ & \mathsf{has\_secret}(\mathcal{D}, V'), \\ & \mathsf{related}(V, V'). \end{aligned}$$

$$\begin{aligned} r_{3'}^m : \mathsf{open\_query}(\mathcal{A}, V) \qquad \leftarrow\ & \mathsf{sa}(\mathsf{query}, \mathcal{A}, \mathcal{D}, V, T_1), \\ & \mathsf{not\ sa}(\mathsf{answer}, \mathcal{D}, \mathcal{A}, V', T_2), \\ & \mathsf{related}(V, V'), \mathsf{time}(T_2), T_1 \leqslant T_2. \end{aligned}$$

The body of rule $r_{3'}^m$ is satisfied if $\mathcal{D}$ has received a query from $\mathcal{A}$, and has not sent an answer with a literal that is related to the queried literal to $\mathcal{A}$. Then $\mathcal{D}$ has an open query with respect to $\mathcal{A}$ and the queried literal. The body of rule r_3^m is satisfied if $\mathcal{D}$ has on open query with respect to $\mathcal{A}$ and a literal that is related to one of its potentially secret formulae.

We assume that the motives of $\mathcal{D}$ contain the following motives to answer queries containing the before mentioned motive to answer sensitive queries, and a motive to answer queries in general:

$$\mathfrak{M}_{\mathsf{answer}} =_{def} \{(\mathsf{answered}(\mathcal{X}, V), \{\mathsf{open\_sens\_query}(\mathcal{X}, V)\}, 1), \quad (\mathsf{answered}(\mathcal{X}, V), \{\mathsf{open\_query}(\mathcal{X}, V)\}, 0.6)\} \tag{5.6.11}$$

That is, we assume that $\mathfrak{M}_{\mathsf{answer}} \subseteq \mathfrak{M}_b$.

By means of $\mathfrak{M}_{\mathsf{answer}}$ in combination with the rules r_3^m and $r_{3'}^m$ and the assumption that no other belief-desire coupling with a motivational value of 1 exists we made sure that, in case of an open sensitive query the agent has the intention to answer this query. Then, the know-how base $\mathfrak{KH}_b^{\mathsf{asp}}$ provides three possible actions for the satisfaction of the intention, to answer with the literal, the negation of the literal, or to perform the empty action. Agent $\mathcal{D}$ then evaluates these three options with respect to secrecy and selects one of the least secrecy violating ones. If a secrecy preserving existing option exists the agent will perform a secrecy preserving one. We demanded in the Assumptions 1. and 2. that there always exists a secrecy preserving action such that the *secrecy ASP BDI*$^+$ *agents* preserve secrecy, in our considered setting.

In the next section we illustrate our complete *secrecy ASP BDI*$^+$ *model* by means of our *strike committee meeting* example.

5.6.6 *Complete modeling of the* scm *example*

Example 5.6.12. We completely model the *strike committee meeting* example by use of the *secrecy ASP BDI*$^+$ *model* we developed. Hereby we build on the partial model given in Example 5.6.2. First we define the initial epistemic states of the two agents $\mathcal{E}$ for *Emma* and $\mathcal{B}$ for *Beatriz*, these are

$$\mathcal{K}_{\mathcal{E}}^0 = \langle \{V_{\mathcal{E},W}^0, V_{\mathcal{E},\mathcal{B}}^0, \mathcal{S}_{\mathcal{E}}^0\}, \mathfrak{D}_{\mathcal{E}}^0, \mathfrak{I}_{\mathcal{E}}^0, \mathfrak{M}_{\mathcal{E}}, \mathfrak{KH}_{\mathcal{E}} \rangle$$

and

$$\mathcal{K}_{\mathcal{B}}^0 = \langle \{V_{\mathcal{B},W}^0, V_{\mathcal{B},\mathcal{E}}^0, \mathcal{S}_{\mathcal{B}}^0\}, \mathfrak{D}_{\mathcal{B}}^0, \mathfrak{I}_{\mathcal{B}}^0, \mathfrak{M}_{\mathcal{B}}, \mathfrak{KH}_{\mathcal{B}} \rangle.$$

We assume that the view of $\mathcal{E}$ on the world-view of $\mathcal{B}$ is accurate and complete, such that it is the same as the world-view of $\mathcal{B}$, i.e., $V_{\mathcal{E},\mathcal{B}}^0 = V_{\mathcal{B},W}^0$. In Example 5.6.2 (Page 153) we defined the program $P_{\mathcal{B}}^{bg}$ and stated that $P_{\mathcal{B}}^{bg} \subseteq V_{\mathcal{E},\mathcal{B}}^0$. The complete view on the world-view of $\mathcal{B}$ comprises the program $P_{\mathcal{B}}^{bg}$ and the meta-information level rules defined above. The resulting program is therefore:

$V_{\mathcal{B},W}^0 = V_{\mathcal{E},\mathcal{B}}^0 = P_{\mathcal{B}}^{bg} \cup P_{\mathsf{R\text{-}sensitive}}^{\mathsf{meta}} \cup P_{\mathcal{S}\text{-}\mathsf{aware}}^{\mathsf{meta}} \cup \{r_3^m, r_{3'}^m\} \cup P_{\mathsf{link}}(At) = \{$

$$\begin{array}{ll}
r_1 : \mathsf{attend\_work} & \leftarrow \mathsf{not}\ \neg\mathsf{attend\_work}. \\
r_2 : \neg\mathsf{attend\_work} & \leftarrow \mathsf{day\_off}. \\
r_3 : \mathsf{day\_off} & \leftarrow \neg\mathsf{attend\_scm}. \\
r_4 : \mathsf{day\_off} & \leftarrow \mathsf{attend\_scm}. \\
r_5 : \mathsf{blacklist} & \leftarrow \mathsf{attend\_scm}.
\end{array}$$

$$
\begin{array}{lll}
r_6 : \mathsf{blacklist} & \leftarrow & \mathsf{not\ day\_off}, \neg\mathsf{attend\_work}. \\
r_s^m : \mathsf{has\_secret}(\mathcal{E},\ \mathsf{attend\_scm}). & & \\
r_1^m : \mathsf{refused}(\mathcal{E}, V) & \leftarrow & \mathsf{sa}(\mathsf{query}, \mathcal{B}, \mathcal{E}, V, T_1), \\
 & & \mathsf{not\ sa}(\mathsf{answer}, \mathcal{E}, \mathcal{B}, V', T_2), \\
 & & \mathsf{at}(T_2), \mathsf{related}(V, V'), \\
 & & T_2 \geqslant T_1 + t_{\mathsf{resp}}. \\
r_2^m : \mathsf{holds}(V) & \leftarrow & \mathsf{has\_secret}(\mathcal{E}, V), \mathsf{refused}(\mathcal{E}, V). \\
r_3^m : \mathsf{open\_sens\_query}(\mathcal{E}, V) & \leftarrow & \mathsf{open\_query}(\mathcal{E}, V), \\
 & & \mathsf{has\_secret}(\mathcal{B}, V'), \\
 & & \mathsf{related}(V, V'). \\
r_{3'}^m : \mathsf{open\_query}(\mathcal{E}, V) & \leftarrow & \mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathcal{B}, V, T_1), \\
 & & \mathsf{not\ sa}(\mathsf{answer}, \mathcal{B}, \mathcal{E}, V', T_2), \\
 & & \mathsf{related}(V, V'), \mathsf{time}(T_2), T_1 \leqslant T_2.\}
\end{array}
$$

$\cup\ P_{\mathsf{link}}(At)$

To keep the presentation of the programs comprehensible we do not list the rules of $P_{\mathsf{link}}(At)$. Also for the answer sets we leave out the literals stemming from $P_{\mathsf{link}}(At)$, unless they are important for the illustration of some aspect. We indicate the omission by "$\ldots$". The set of answer sets of $V^0_{\mathcal{E},\mathcal{B}}$ is given as

$$
\begin{aligned}
AS(V^0_{\mathcal{B},W}) = AS(V^0_{\mathcal{E},\mathcal{B}}) = \{\{&\mathsf{attend\_work}, \\
&\mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}), \ldots\}\}.
\end{aligned}
$$

We now turn to the representation of the world-view of $\mathcal{E}$. In Example 5.6.2 (Page 153) we defined the program $P^{bg}_{\mathcal{E}}$ and stated that $P^{bg}_{\mathcal{E}} \subseteq V^0_{\mathcal{E},W}$. The complete view on the world-view of $\mathcal{E}$ comprises the program $P^{bg}_{\mathcal{E}}$ and the meta-information level rules defined above:

$$
V^0_{\mathcal{E},W} = P^{bg}_{\mathcal{E}} \cup P^{\mathsf{meta}}_{\mathsf{R\text{-}sensitive}} \cup P^{\mathsf{meta}}_{\mathcal{S}\text{-}\mathsf{aware}} \cup \{r_4^m, r_{4'}^m\} \cup\ P_{\mathsf{link}}(At) = \{
$$

$$
\begin{array}{lll}
r_1 : \mathsf{attend\_work} & \leftarrow & \mathsf{not}\ \neg\mathsf{attend\_work}. \\
r_2 : \neg\mathsf{attend\_work} & \leftarrow & \mathsf{day\_off}. \\
r_3 : \mathsf{attend\_scm}. & & \\
r_s^m : \mathsf{has\_secret}(\mathcal{E},\ \mathsf{attend\_scm}). & & \\
r_1^m : \mathsf{refused}(\mathcal{B}, V) & \leftarrow & \mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathcal{B}, V, T_1), \\
 & & \mathsf{not\ sa}(\mathsf{answer}, \mathcal{B}, \mathcal{E}, V', T_2), \\
 & & \mathsf{at}(T_2), \mathsf{related}(V, V'), \\
 & & T_2 \geqslant T_1 + t_{\mathsf{resp}}. \\
r_2^m : \mathsf{holds}(V) & \leftarrow & \mathsf{has\_secret}(\mathcal{B}, V), \mathsf{refused}(\mathcal{B}, V). \\
r_3^m : \mathsf{open\_sens\_query}(\mathcal{B}, V) & \leftarrow & \mathsf{open\_query}(\mathcal{B}, V), \\
 & & \mathsf{has\_secret}(\mathcal{E}, V'), \\
 & & \mathsf{related}(V, V'). \\
r_{3'}^m : \mathsf{open\_query}(\mathcal{B}, V) & \leftarrow & \mathsf{sa}(\mathsf{query}, \mathcal{B}, \mathcal{E}, V, T_1), \\
 & & \mathsf{not\ sa}(answer, \mathcal{E}, \mathcal{B}, V', T_2), \\
 & & \mathsf{related}(V, V'), \mathsf{at}(T_2), T_1 \leqslant T_2.\}
\end{array}
$$

$\cup\ P_{\mathsf{link}}(At)$

Now we define the view of $\mathcal{B}$ on the world-view of $\mathcal{E}$. We assume that $\mathcal{B}$ knows the world-view of $\mathcal{E}$ apart from the fact that $\mathcal{E}$ intends to attend the *strike committee meeting*:

$$V^0_{\mathcal{B},\mathcal{E}} = V^0_{\mathcal{E},W} \setminus \{\mathsf{attend\_scm.}\}.$$

The set of answer sets of the initial view of $\mathcal{E}$ on the world is

$$\begin{aligned} AS(V^0_{\mathcal{E},W}) = \{\{&\mathsf{attend\_scm},\\ &\mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}),\\ &\mathsf{attend\_work}, \dots\}\} \end{aligned}$$

and the set of answer sets of the initial view of $\mathcal{E}$ on the world-view of $\mathcal{B}$ is

$$\begin{aligned} AS(V^0_{\mathcal{B},\mathcal{E}}) = \{\{\ &\mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}),\\ &\mathsf{attend\_work}, \dots\}\} \end{aligned}$$

Agent $\mathcal{E}$ has the initial set of secrets

$$\mathcal{S}_{\mathcal{E}} = \{(\mathsf{attend\_scm}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}, \mathcal{B})\},$$

as in Example 5.6.2. $\mathcal{B}$ does not have any secrets, such that

$$\mathcal{S}_{\mathcal{B}} = \emptyset.$$

The static know-how base of both agents is given by the ASP know-how base in Equation (5.6.9) (Page 163), i. e.,

$$\mathfrak{KH}_{\mathcal{E}} = \mathfrak{KH}_{\mathcal{B}} = \mathfrak{KH}^{\mathsf{asp}}_{\mathfrak{b}}.$$

The set of motives of $\mathcal{E}$ is given by the motives to answer queries $\mathfrak{M}_{\mathsf{answer}}$ as given in (5.6.11), $\mathcal{E}$'s motive to ask for the day of the *strike committee meeting* off, if it has to work that day, and the motive to not act with the minimal motivation value.

$$\begin{aligned} \mathfrak{M}_{\mathcal{E}} =_{def} \ &\mathfrak{M}_{\mathsf{answer}} \cup \\ &\{(\mathsf{queried}(\mathcal{B}, \mathsf{c}(\mathsf{day\_off})), \{\mathsf{scm}, \mathsf{attend\_work}\}, 0.8),\\ &(\mathsf{not\_acted}, \emptyset, 0)\}\\ \\ = \ &\{(\mathsf{answered}(\mathcal{X}, V), \{\mathsf{open\_sens\_query}(\mathcal{X}, V)\}, 1),\\ &(\mathsf{answered}(\mathcal{X}, V), \{\mathsf{open\_query}(\mathcal{X}, V)\}, 0.6),\\ &(\mathsf{queried}(\mathcal{B}, \mathsf{c}(\mathsf{day\_off})), \{\mathsf{scm}, \mathsf{attend\_work}\}, 0.8),\\ &(\mathsf{not\_acted}, \emptyset, 0)\} \end{aligned}$$

The last motive lets an agent perform the empty action if there is no other motive with a satisfied condition and higher motivation value.

The set of motives of $\mathcal{B}$ is given by the motives to answer queries $\mathfrak{M}_{\mathsf{answer}}$, the motive to ask an employee if she attends the *strike committee meeting*, if the employee asks for that day off, and the motive to not act:

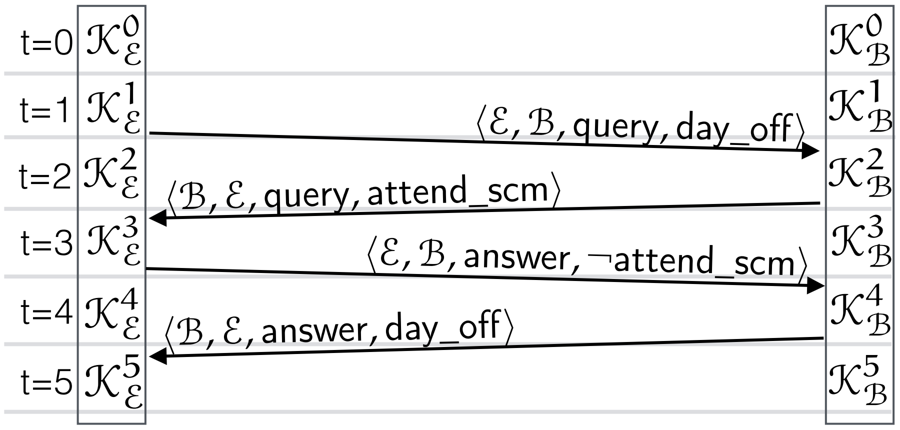

t=0
t=1
t=2
t=3
t=4
t=5
$\mathcal{K}_\mathcal{E}^0$ $\mathcal{K}_\mathcal{E}^1$ $\mathcal{K}_\mathcal{E}^2$ $\mathcal{K}_\mathcal{E}^3$ $\mathcal{K}_\mathcal{E}^4$ $\mathcal{K}_\mathcal{E}^5$
$\mathcal{K}_\mathcal{B}^0$ $\mathcal{K}_\mathcal{B}^1$ $\mathcal{K}_\mathcal{B}^2$ $\mathcal{K}_\mathcal{B}^3$ $\mathcal{K}_\mathcal{B}^4$ $\mathcal{K}_\mathcal{B}^5$
$\langle \mathcal{E}, \mathcal{B}, \text{query}, \text{day\_off} \rangle$
$\langle \mathcal{B}, \mathcal{E}, \text{query}, \text{attend\_scm} \rangle$
$\langle \mathcal{E}, \mathcal{B}, \text{answer}, \neg\text{attend\_scm} \rangle$
$\langle \mathcal{B}, \mathcal{E}, \text{answer}, \text{day\_off} \rangle$

Figure 5.6.2: Timeline for the scenario in Example 3.3.2.

$$\begin{aligned}
\mathfrak{M}_\mathcal{B} =_{def} & \ \mathfrak{M}_{\text{answer}} \cup \\
& \{(\text{queried}(\mathcal{E}, c(\text{attend\_scm})), \{\text{open\_query}(\mathcal{E}, c(\text{day\_off})\}, 0.8), \\
& (\text{not\_acted}, \emptyset, 0)\} \\
= & \ \{(\text{answered}(\mathcal{X}, V), \{\text{open\_sens\_query}(\mathcal{X}, V)\}, 1), \\
& (\text{answered}(\mathcal{X}, V), \{\text{open\_query}(\mathcal{X}, V)\}, 0.6), \\
& (\text{queried}(\mathcal{E}, c(\text{attend\_scm})), \{\text{open\_query}(\mathcal{E}, c(\text{day\_off})\}, 0.8), \\
& (\text{not\_acted}, \emptyset, 0)\}
\end{aligned}$$

The initial sets of desires of $\mathcal{E}$ and $\mathcal{B}$ are initially empty, i. e.,

$$\mathfrak{D}_\mathcal{E}^0 = \mathfrak{D}_\mathcal{B}^0 = \emptyset.$$

Both agents do not have any intentions initially, such that

$$\mathfrak{I}_\mathcal{E}^0 = \mathfrak{I}_\mathcal{B}^0 = \emptyset.$$

The belief operators of the agents are

$$\text{Bel}_\mathcal{B} = \text{Bel}_\mathcal{E} = \text{Bel}_{\text{skep}}^{\text{asp}}.$$

We completely defined both initial epistemic states. They are both safe: for $\mathcal{K}_\mathcal{E}^0$ it holds that $\text{attend\_scm} \notin \text{Bel}_{\text{skep}}^{\text{asp}}(V_{\mathcal{E},\mathcal{B}}^0)$, and $\mathcal{K}_\mathcal{B}^0$ is safe since $\mathcal{S}(\mathcal{K}_\mathcal{B}^0) = \emptyset$.

In the following we show that and how this two agent system with the just defined initial agent states leads to the interaction we described in Example 3.3.2 (Page 43), which is illustrated again in Figure 5.6.2.

Before we do this we make a few notes on the way in which we present the run of the agent system. Since at each point in time only one of the agents is executed and since $\mathcal{E}$ is executed first it is $\mathcal{K}_\mathcal{E}^t = \mathcal{K}_\mathcal{E}^{(t+1)}$, if t is odd; and $\mathcal{K}_\mathcal{B}^t = \mathcal{K}_\mathcal{B}^{(t+1)}$, if t is even. For the same reason speech acts sent by one agent a time t are received by the other agent at time $t+1$.

We structure the presentation of the interaction of the two agents by the time, i. e., $t = 1, t = 2, \ldots$, in each time step one agent cycle is performed. The agent cycle is illustrated in Figure 5.6.1 (Page 151). Each agent cycle consists of the execution of the operators $\circ$, act and $\circ^{a}$, in this order. We use paragraphs for each operator.

We use the super-script t_p to denote that the epistemic state, or components of it after the change by the incoming percept and before the change by the agent's action. That is, for an epistemic state $\mathcal{K}^t$ we denote $\mathcal{K}^t \circ p$ as $\mathcal{K}^{(t+1)p}$ and $\mathcal{K}^{(t+1)p} \circ^a \mathsf{act}(\mathcal{K}^{(t+1)p})$ as $\mathcal{K}^{t+1}$, and the components accordingly. The secrets, intentions and desires are not indexed this way since they are not changed by $\circ^a$. The motives and the know-how base do not change. The description of the execution of the $\circ$ operator is again structured into its sub-operations $\circ_{\mathfrak{B}}$, $\circ_{\mathfrak{D}}$ and $\circ_{\mathfrak{I}}$, as also illustrated in Figure 5.6.1 (Page 151).

We start with the description of the agent cycle of $\mathcal{E}$ in time step 1. We present the steps of the first agent cycle in detail and focus on the relevant steps in the presentation following agent cycles, to keep the presentation of the example clear.

$\underline{t = 1}$:
In its first agent cycle agent $\mathcal{E}$ does not perceive anything, represented by the empty percept p_ϵ .

$\underline{\mathcal{K}^0_{\mathcal{E}} \circ p_\epsilon}$:
As defined in Section 3.5.3 (Page 54) and Section 5.6.1 the change operator $\circ$ is realized by three sub-operators, such that $\circ =_{def} \circ_{\mathfrak{B}} \cdot \circ_{\mathfrak{D}} \cdot \circ_{\mathfrak{I}}$:

$\underline{\circ_{\mathfrak{B}}}$: The belief change operator first changes the world-view, then the agent-view and then the secrets, as defined in Equation (5.6.6) (Page 155). The change operators for the views are realized by two sub-operators, the interpretation operator for ASP t^{asp} that interprets a speech act and outputs an answer set program (Definition 5.6.5, Page 158), and an *ASP multiple base revision operator* $*^{\mathsf{asp}}$ that revises the view by the program output of the interpretation operator (Definition 4.3.17, Page 93). The secrets are changed by the secrets change operator $*_{\mathcal{S}}$ that is defined in Equation 5.3.9 (Page 118).

The interpretation function for the world-view of $\mathcal{B}$ and for the view of $\mathcal{B}$ on $\mathcal{E}$ leads to the following results:

$$t^{\mathsf{asp}}(\mathcal{E}, \mathcal{E}, p_\epsilon) = \{\mathsf{time(1).}\}$$

and

$$t^{\mathsf{asp}}(\mathcal{E}, \mathcal{B}, p_\epsilon) = \{\mathsf{time(0).}\}.$$

The revision operation for the world-view is thus

$$V^0_{\mathcal{E},W} *^{\mathsf{asp}} \{\mathsf{time(1).}\}$$

The $*^{\mathsf{asp}}$ operator satisfies *Inclusion* and *Vacuity*, which we defined in Section 4.3.1, Page 86 as:

Inclusion: $P * Q \subseteq P \cup Q$

Vacuity: If $P \cup Q$ is consistent, then $P \cup Q \subseteq P * Q$

From the satisfaction of these postulates and since $V^{0}_{\mathcal{E},W} + \{\mathsf{time(1).}\} = V^{0}_{\mathcal{E},W} \cup \{\mathsf{time(1).}\}$ is consistent it follows that

$$V^{1_p}_{\mathcal{E},W} = V^{0}_{\mathcal{E},W} *^{\mathsf{asp}} \{\mathsf{time(1).}\} = V^{0}_{\mathcal{E},W} \cup \{\mathsf{time(1).}\}.$$

The same holds for $\mathcal{E}$'s view on the world-view of $\mathcal{B}$, such that

$$V^{1_p}_{\mathcal{E},\mathcal{B}} = V^{0}_{\mathcal{E},\mathcal{B}} *^{\mathsf{asp}} \{\mathsf{time(0).}\} = V^{0}_{\mathcal{E},\mathcal{B}} \cup \{\mathsf{time(0).}\}.$$

As noted before, we determine the sets of answer sets of the epistemic components since they form the basis for the computation of the belief sets for both belief operators ($\mathsf{Bel_{skep}}$ and $\mathsf{Bel_{cred}}$, defined in ASP Instance 6, Page 66) we consider here. If there is just one answer set, then the belief sets given by both operators are equal to this answer set, i. e., if $AS(P) = \{I\}$, then

$$\mathsf{Bel_{skep}}(P) = \cap AS(P) = \mathsf{Bel_{cred}}(P) = \cup AS(P) = I.$$

For the clear presentation of the evolution of the answer sets of the different views we order the answer sets by first listing the literals that are facts in the program, then the literals that are inferred. We order both parts according to the order in which they appeared in the answer set of the respective view. Further, we print the literals that have already been part of the previous version of the answer set of the considered view in gray, and those that were part of the previous version, but are in the current version crossed out.

The sets of answer sets are:

$$\begin{aligned} AS(V^{1_p}_{\mathcal{E},W}) = \{\{ & \mathsf{attend\_scm}, \\ & \mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}), \\ & \mathsf{time(1)}, \\ & \\ & \mathsf{attend\_work}, \\ & \mathsf{at(1)}, \ldots\}\} \end{aligned}$$

and

$$\begin{aligned} AS(V^{1_p}_{\mathcal{E},\mathcal{B}}) = \{\{ & \mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}), \\ & \mathsf{time(0)}, \\ & \\ & \mathsf{attend\_work}, \\ & \mathsf{at(0)}, \ldots\}\} \end{aligned}$$

If there is just one answer set, then the belief set is equal to the answer set for both belief operators such that the secret of $\mathcal{E}$ is not violated, since

$$\mathsf{attend\_scm} \notin \mathsf{Bel}_{\mathsf{skep}}(AS(V^{1_p}_{\mathcal{E},\mathcal{B}})).$$

No secret is violated such that, since $*_{\mathcal{S}}$ satisfies the *Min-Secrecy-Weakening*$_\circ$ property, (5.3.7) Page 115, the set of secrets of $\mathcal{E}$ is not changed by the $*_{\mathcal{S}}$ operator (as defined in (5.3.9), Page 118), i. e.:

$$\mathcal{S}^1 = *_{\mathcal{S}}(\mathcal{S}^0, V^{1_p}_{\mathcal{E},W}, V^{1_p}_{\mathcal{E},\mathcal{B}}) = \mathcal{S}^0$$

$\underline{\circ_{\mathfrak{D}}}$: On the basis of the changed belief component, the desire operator generates the current desires of the agent. The secrecy desire operator is defined as

$$\circ_{\mathfrak{D},s}(\mathfrak{M}_b, (V_{\mathcal{D},W}, V_{\mathcal{D},\mathcal{A}}, \mathcal{S})) =_{def}$$

$$\{(D, \mu) \mid (D, \Phi, \mu) \in \mathfrak{M}_b, \Phi \subseteq \mathsf{Bel}_{\mathcal{D}}(V_{\mathcal{D},W})\}$$

in Equation (5.6.7) (Page 161).

The conditions of all belief-desire couplings in $\mathfrak{M}_{\mathcal{E}}$ are checked, with

$$\begin{aligned}\mathfrak{M}_{\mathcal{E}} =_{def} \{&(\mathsf{answered}(\mathcal{X}, V), \{\mathsf{open\_sens\_query}(\mathcal{X}, V)\}, 1),\\ &(\mathsf{answered}(\mathcal{X}, V), \{\mathsf{open\_query}(\mathcal{X}, V)\}, 0.6),\\ &(\mathsf{queried}(\mathcal{B}, \mathsf{c}(\mathsf{day\_off})), \{\mathsf{scm}, \mathsf{attend\_work}\}, 0.8),\\ &(\mathsf{not\_acted}, \emptyset, 0)\}.\end{aligned}$$

Only the conditions of the third and fourth belief-desire coupling are satisfied, since $\{\mathsf{scm}, \mathsf{attend\_work}\} \subseteq AS(V^{1_p}_{\mathcal{E},W})$. Hence, the resulting current set of desires is

$$\mathfrak{D}^1_{\mathcal{E}} = \{(\mathsf{queried}(\mathcal{B}, \mathsf{c}(\mathsf{day\_off}), 0.8), (\mathsf{not\_acted}, 0)\}.$$

$\underline{\circ_{\mathfrak{I}}}$: The intention operator is defined in Procedure 3.5.1 (Page 60), the opt_b operator is adapted to the secrecy setting in Section 5.6.8. The intention operator first sets a maximally motivated desire as the agent's current intention. Since there is only one desire this one is selected, it is the unique maximally motivated desire. Then, the options for the satisfaction of this intention are generated by the know-how base $\mathfrak{KH}^{\mathsf{asp}}_b$ (see (5.6.9), Page 163) of the agent. The options are the subgoals of all know-how statements whose conditions are satisfied, these are

$$\mathsf{sacted}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off})) \text{ and } \mathsf{not\_acted}.$$

These options are evaluated by the eval function which determines the preference relation $\preceq^s_{(\mathcal{K},\xi)}$ with respect to secrecy and the preference relation with respect to informativity $\preceq^a_{\mathcal{K}}$ and then aggregates these.

For the construction of the $\preceq^s_{(\mathcal{K},\xi)}$ relation for each option a the hypothetically evolved world-view $V^{3p}_{\mathcal{E},\mathcal{B}} *^{asp} t^{asp}(\mathcal{D}, \mathcal{A}, a)$ is constructed and the violation of secrets with respect to it considered, as defined in Equation (5.6.10) (Page 164). The currently active secrets are given as $\mathcal{S}_{\mathsf{active}}(V^{1p}_{\mathcal{E},W}, \mathcal{S}^1) = \mathcal{S}^1$ since $\mathsf{attend\_scm} \in \mathsf{Bel}_{\mathcal{E}}(V_{\mathcal{E},W})$. In both hypothetically evolved world-views no active secret is violated:

$$\begin{aligned}&\mathsf{vioAfter}(\mathcal{S}_{\mathsf{active}}(V^{1p}_{\mathcal{E},W}, \mathcal{S}^1), V^{1p}_{\mathcal{E},\mathcal{B}},\\ &\qquad\qquad \alpha(\mathsf{sacted}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off}))))\\ &= \mathsf{vioAfter}(\mathcal{S}_{\mathsf{active}}(V^{1p}_{\mathcal{E},W}, \mathcal{S}^1), V^{1p}_{\mathcal{E},\mathcal{B}}, \alpha(\mathsf{not\_acted}))\\ &= \{\emptyset\}\end{aligned}$$

Both options are secrecy preserving and classified with rank 0. Hence they are equally preferred with respect to the $\preceq^s_{(\mathcal{K},\xi)}$ relation, i. e.,

$$\alpha(\mathsf{sacted}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off}))) =^s \alpha(\mathsf{not\_acted}).$$

By the $\preceq^a_{\mathcal{K}}$ relation, as defined in (5.5.5) (Page 132), the first one is preferred to the second one, i. e.

$$\alpha(\mathsf{not\_acted}) \preceq^a_{\mathcal{K}} \alpha(\mathsf{sacted}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off})))$$

For the aggregated preference relation, by the lexicographic aggregation operator,

$$\preceq_{(\mathcal{K},\xi)} =_{def} f^{\mathsf{lex}}_{agg}(\preceq^s_{(\mathcal{K},\xi)}, \preceq^a_{\mathcal{K}})$$

(cf. (5.5.2), Page 131) we thus get that

$$\alpha(\mathsf{not\_acted}) \preceq_{(\mathcal{K},\xi)} \alpha(\mathsf{sacted}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off})))$$

and consequentlyis

$$\mathcal{I}^1_{\mathcal{E}} = \{\sigma(\max_{\preceq_{(\mathcal{K},\xi)}}(\mathsf{options}))\} = \{\mathsf{sacted}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off}))\}$$

act: As defined in Equation (3.5.8) (Page 59) the act function is given as:

$$\mathsf{act}(\mathcal{K}) =_{def} \alpha(I).$$

The speech act that directly satisfies the only atomic intention of the agent is executed, this is:

$$\alpha(\mathsf{sacted}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off}))) = \langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off}\rangle$$

$\mathcal{K}^{1p}_{\mathcal{E}} \circ^a \langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off}\rangle$:
The $\circ^a$ operator only changes the beliefs of an ASP *secrecy* BDI$^+$ agent, as defined in Equation (3.5.9) (Page 60). That is, $\circ^a = \circ^a_{\mathfrak{B}}$. By this operation $\mathcal{E}$ changes its views to reflect the changes implied by the action it just executed.

$\underline{\circ^{a}_{\mathfrak{B}}}$: The change operator for actions is defined in Equation 5.6.6 as

$$\begin{aligned}(V_{\mathcal{D},\mathcal{W}}, V_{\mathcal{D},\mathcal{A}}, \mathcal{S}) \circ^{a,asp}_{\mathfrak{B}} a = \langle & V_{\mathcal{D},\mathcal{W}} *^{asp} t^{asp}(\mathcal{D},\mathcal{D},a), \\ & V_{\mathcal{D},\mathcal{A}} *^{asp} t^{asp}(\mathcal{D},\mathcal{A},a), \\ & \mathcal{S}\rangle.\end{aligned}$$

The results of the interpretation function applied to the given speech act for the world-view of $\mathcal{E}$ and the interpretation function for the view of $\mathcal{E}$ on the world-view of $\mathcal{B}$ are

$$\begin{aligned}&t^{asp}(\mathcal{E},\mathcal{E},\langle\mathcal{E},\mathcal{B},\mathsf{query},\mathsf{day\_off}\rangle) \\ &= \{\mathsf{sa}(\mathsf{query},\mathcal{E},\mathsf{c}(\mathsf{day\_off}),1).,\mathsf{time}(1).\},\end{aligned}$$

and

$$\begin{aligned}&t^{asp}(\mathcal{E},\mathcal{B},\langle\mathcal{E},\mathcal{B},\mathsf{query},\mathsf{day\_off}\rangle) \\ &= \{\mathsf{sa}(\mathsf{query},\mathcal{E},\mathsf{c}(\mathsf{day\_off}),2).,\mathsf{time}(2).\}.\end{aligned}$$

Both programs are consistent with the world-view of $\mathcal{E}$ such that

$$\begin{aligned}&V^{1p}_{\mathcal{E},\mathcal{W}} *^{asp} \{\mathsf{sa}(\mathsf{query},\mathcal{E},\mathsf{c}(\mathsf{day\_off}),1).,\mathsf{time}(1).\} \\ &= V^{1p}_{\mathcal{E},\mathcal{W}} \cup \{\mathsf{sa}(\mathsf{query},\mathcal{E},\mathsf{c}(\mathsf{day\_off}),1).,\mathsf{time}(1).\} \\ &= V^{1}_{\mathcal{E},\mathcal{W}}.\end{aligned}$$

and

$$\begin{aligned}&V^{1p}_{\mathcal{E},\mathcal{B}} *^{asp} \{\mathsf{sa}(\mathsf{query},\mathcal{E},\mathsf{c}(\mathsf{day\_off}),2).,\mathsf{time}(2).\} \\ &= V^{1p}_{\mathcal{E},\mathcal{B}} \cup \{\mathsf{sa}(\mathsf{query},\mathcal{E},\mathsf{c}(\mathsf{day\_off}),2).,\mathsf{time}(2).\} \\ &= V^{1}_{\mathcal{E},\mathcal{B}}.\end{aligned}$$

The sets of answer sets are:

$$\begin{aligned}AS(V^{1}_{\mathcal{E},\mathcal{W}}) = \{\{&\mathsf{attend\_scm}, \\ &\mathsf{has\_secret}(\mathcal{E},\mathsf{attend\_scm}), \\ &\mathsf{time}(1), \\ &\mathsf{sa}(\mathsf{query},\mathcal{E},\mathsf{c}(\mathsf{day\_off}),1), \\ \\ &\mathsf{attend\_work}, \\ &\mathsf{at}(1),\ldots\}\}\end{aligned}$$

and

$$\begin{aligned}AS(V^{1}_{\mathcal{E},\mathcal{B}}) = \{\{&\mathsf{has\_secret}(\mathcal{E},\mathsf{attend\_scm}), \\ &\mathsf{time}(0), \\ &\mathsf{sa}(\mathsf{query},\mathcal{E},\mathsf{c}(\mathsf{day\_off}),2), \\ &\mathsf{time}(2), \\ \\ &\mathsf{attend\_work}, \\ &\mathsf{open\_query}(\mathcal{E},\mathsf{c}(\mathsf{day\_off})), \\ &\mathsf{at}(2),\ldots\}\}.\end{aligned}$$

From the agent-view of $\mathcal{E}$ on $\mathcal{B}$ it thus follows that $\mathcal{B}$ has an open query with respect to $\mathcal{E}$ and the literal day_off.

It holds that the secret of $\mathcal{E}$ is not violated, since

$$\mathsf{attend\_scm} \notin \mathsf{Bel}_{\mathsf{skep}}(AS(V^1_{\mathcal{E},\mathcal{B}})).$$

In the following we describe only the relevant steps in the presentation of the agent cycles. In particular we leave out the detailed presentation of the belief change operation. The result of the interpretation operator results directly from its definition. The revision operation is reduced to the union operation since the interpretation result is always consistent with the respective view in this example. We do list the sets of answer sets, which give a concise presentation of the changes.

$\underline{t = 2}$:

$\underline{\mathcal{K}^0_{\mathcal{B}} \circ \langle \mathcal{E}, \mathcal{B}, \mathsf{query}, \mathsf{day\_off} \rangle}$:

$\underline{\circ_{\mathfrak{B}}}$: Agent $\mathcal{B}$ receives the query for the day off from $\mathcal{E}$ and changes its beliefs component accordingly. The resulting sets of answer sets are:

$$\begin{aligned} AS(V^{2p}_{\mathcal{B},W}) = \{\{ & \mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}),\\ & \mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{day\_off}), 2),\\ \\ & \mathsf{attend\_work},\\ & \mathsf{open\_query}(\mathcal{E}, \mathsf{c}(\mathsf{day\_off})),\\ & \mathsf{at}(2), \ldots\}\} \end{aligned}$$

and

$$\begin{aligned} AS(V^{2p}_{\mathcal{B},\mathcal{E}}) = \{\{ & \mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}),\\ & \mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{day\_off}), 1),\\ \\ & \mathsf{attend\_work},\\ & \mathsf{at}(1), \ldots\}\} \end{aligned}$$

The belief set of the world-view of agent $\mathcal{B}$ now contains the literal $\mathsf{open\_query}(\mathcal{E}, \mathsf{c}(\mathsf{day\_off}))$ that denotes that $\mathcal{B}$ has not answered the query for a day off from $\mathcal{E}$. Since $\mathcal{B}$ has no secrets, no secret is violated.

$\underline{\circ_{\mathfrak{D}}}$: The conditions of all belief-desire couplings in $\mathfrak{M}_{\mathcal{B}}$ are checked, with

$$\begin{aligned} \mathfrak{M}_{\mathcal{B}} = \{ & (\mathsf{answered}(\mathcal{X}, V), \{\mathsf{open\_sens\_query}(\mathcal{X}, V)\}, 1),\\ & (\mathsf{answered}(\mathcal{X}, V), \{\mathsf{open\_query}(\mathcal{X}, V)\}, 0.6),\\ & (\mathsf{queried}(\mathcal{E}, \mathsf{c}(\mathsf{attend\_scm})),\\ & \qquad \{\mathsf{open\_query}(\mathcal{E}, \mathsf{c}(\mathsf{day\_off})\}, 0.8)\\ & (\mathsf{not\_acted}, \emptyset, 0)\}. \end{aligned}$$

The conditions of the second, third and fourth belief-desire coupling are satisfied in this case, such that the desire operator generates the following set of desires

$$\begin{aligned}\mathfrak{D}^2_{\mathcal{B}} = \{&(\mathsf{queried}(\mathcal{E}, \mathsf{c}(\mathsf{attend\_scm})), 0.8),\\ &(\mathsf{answered}(\mathcal{E}, \mathsf{c}(\mathsf{day\_off})), 0.6),\\ &(\mathsf{not\_acted}, 0)\}.\end{aligned}$$

The intention $\mathsf{queried}(\mathcal{E}, \mathsf{c}(\mathsf{attend\_scm}))$ results from the maximally motivated desire. The options for its realization, as given by $\mathfrak{KH}^{\mathsf{asp}}_{\mathsf{b}}$ in Equation (5.6.9) (Page 163), are

$$\mathsf{sacted}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{attend\_scm})) \text{ and } \mathsf{not\_acted}.$$

Both options do not lead to the violation of any secrets, but, since it is more informative, the former is preferred over the latter by the $\preceq^{a}_{\mathcal{K}}$ relation (Equation (5.5.5), Page 132) such that

$$\mathfrak{I}^2_{\mathcal{B}} = \{\mathsf{sacted}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{attend\_scm}))\}.$$

act: The executed action is

$$\alpha(\mathsf{sacted}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{attend\_scm}))) = \langle \mathcal{B}, \mathcal{E}, \mathsf{query}, \mathsf{attend\_scm}\rangle.$$

$\mathcal{K}^{2p}_{\mathcal{B}} \circ^{a} \langle \mathcal{B}, \mathcal{E}, \mathsf{query}, \mathsf{attend\_scm}\rangle$:

$\circ^{a}_{\mathfrak{B}}$: $\mathcal{B}$ changes its views to reflect the changes implied by the action it just executed. The sets of answer sets of the changed world and agent-view are:

$$\begin{aligned}AS(V^2_{\mathcal{B},W}) = \{\{ &\mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}),\\ &\mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{day\_off}), 2),\\ &\mathsf{sa}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{attend\_scm}), 2),\\ \\ &\mathsf{attend\_work},\\ &\mathsf{open\_query}(\mathcal{E}, \mathsf{c}(\mathsf{day\_off})),\\ &\mathsf{at}(2), \ldots\}\}\end{aligned}$$

and

$$\begin{aligned}AS(V^2_{\mathcal{B},\mathcal{E}}) = \{\{ &\mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}),\\ &\mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{day\_off}), 1),\\ &\mathsf{sa}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{attend\_scm}), 3),\\ \\ &\mathsf{attend\_work},\\ &\mathsf{open\_query}(\mathcal{B}, \mathsf{c}(\mathsf{attend\_scm})),\\ &\mathsf{open\_sens\_query}(\mathcal{B}, \mathsf{c}(\mathsf{attend\_scm})),\\ &\mathsf{at}(3), \ldots\}\}\end{aligned}$$

The belief set of the view on $\mathcal{E}$ contains the new literals expressing that $\mathcal{E}$ has not answered the sensitive query if it intends to attend the *strike committee meeting* from $\mathcal{B}$. The epistemic state is safe since there is no secret.

$\underline{t = 3}$:

$\underline{\mathcal{K}^1_{\mathcal{E}} \circ \langle \mathcal{B}, \mathcal{E}, \mathsf{query}, \mathsf{attend\_scm} \rangle}$:

$\underline{\circ_{\mathfrak{B}}}$: The sets of answer sets are:

$$\begin{aligned} AS(V^{3_p}_{\mathcal{E},W}) = \{\{ & \mathsf{attend\_scm}, \\ & \mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}), \\ & \mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{day\_off}), 1), \\ & \mathsf{sa}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{attend\_scm}), 3), \\ & \\ & \mathsf{attend\_work}, \\ & \mathsf{open\_query}(\mathcal{B}, \mathsf{c}(\mathsf{attend\_scm})), \\ & \mathsf{open\_sens\_query}(\mathcal{B}, \mathsf{c}(\mathsf{attend\_scm})), \\ & \mathsf{at}(3), \ldots\}\} \end{aligned}$$

and

$$\begin{aligned} AS(V^{3_p}_{\mathcal{E},\mathcal{B}}) = \{\{ & \mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}), \\ & \mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{day\_off}), 2), \\ & \mathsf{sa}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{attend\_scm}), 2), \\ & \\ & \mathsf{attend\_work}, \\ & \mathsf{open\_query}(\mathcal{E}, \mathsf{c}(\mathsf{day\_off})), \\ & \mathsf{at}(2), \ldots\}\} \end{aligned}$$

The world-view of $\mathcal{E}$ contains the a new literal expressing that $\mathcal{E}$ has not answered the sensitive query if it intends to attend the *strike committee meeting* from $\mathcal{B}$.

The secret of $\mathcal{E}$ is active but not violated since:

$$\mathsf{attend\_scm} \notin \mathsf{Bel}_{\mathsf{skep}}(AS(V^{3_p}_{\mathcal{E},\mathcal{B}})).$$

It follows that the set of secrets of $\mathcal{E}$ is not changed by the $*_{\mathcal{S}}$ operator (as defined in (5.3.9), Page 118), i. e.:

$$\mathcal{S}^3_{\mathcal{E}} = *_{\mathcal{S}}(\mathcal{S}^1_{\mathcal{E}}, V^{3_p}_{\mathcal{E},W}, V^{3_p}_{\mathcal{E},\mathcal{B}}) = \mathcal{S}^1_{\mathcal{E}}$$

$\underline{\circ_{\mathfrak{D}}}$: The conditions of the belief-desire-couplings

$$(\mathsf{answered}(\mathcal{B}, \mathsf{attend\_scm}), \mathsf{open\_sens\_query}(\mathcal{B}, \mathsf{attend\_scm}), 1)$$

and

$$(\mathsf{not\_acted}, \emptyset, 0)$$

are satisfied such that

$$\mathfrak{D}^3_{\mathcal{E}} = \{(\mathsf{answered}(\mathcal{B}, \mathsf{attend\_scm}), 1), (\mathsf{not\_acted}, 0)\}$$

$\underline{o_3}$: Agent $\mathcal{E}$ has the intention to answer the sensitive query. The intention $\text{answered}(\mathcal{B}, \text{attend\_scm})$ can be satisfied by satisfying the atomic intentions

$\text{sacted}(\text{answer}, \mathcal{B}, c(\text{attend\_scm}))$, or
$\text{sacted}(\text{answer}, \mathcal{B}, c(\neg\text{attend\_scm}))$, or
not_acted.

All these options are evaluated with respect to secrecy. The currently active secrets are given as $\mathcal{S}_{\text{active}}(V^{3_p}_{\mathcal{E},W}, \mathcal{S}^3) = \mathcal{S}^3$ since $\text{attend\_scm} \in \text{Bel}_{\mathcal{E}}(V_{\mathcal{E},W})$.

For each option a the hypothetically evolved world-view $V^{3_p}_{\mathcal{E},\mathcal{B}} *^{asp} t^{asp}(\mathcal{D}, \mathcal{A}, a)$ is constructed and the violation of secrets with respect to it considered, as defined in (5.6.10) (Page 164). We consider these for the three options considered here in the following:

1.) $\alpha(\text{sacted}(\text{answer}, \mathcal{B}, c(\text{attend\_scm})))$:
The first option leads to the addition of the fact attend_scm. to $V^{3_p}_{\mathcal{E},\mathcal{B}}$ such that

$$\text{attend\_scm} \in \cap AS(V^{3_p}_{\mathcal{E},\mathcal{B}} *^{asp} t^{asp}(\mathcal{D}, \mathcal{A}, a)).$$

The secret is thus violated.

2.) $\alpha(\text{sacted}(\text{answer}, \mathcal{B}, c(\neg\text{attend\_scm})))$:
The second option leads to the addition of the fact ¬attend_scm. to $V^{3_p}_{\mathcal{E},\mathcal{B}}$ such that

$$\neg\text{attend\_scm} \in \cap AS(V^{3_p}_{\mathcal{E},\mathcal{B}} *^{asp} t^{asp}(\mathcal{D}, \mathcal{A}, a)).$$

The secret is not violated in this case.

3.) $\alpha(\text{not\_acted})$:
The third option leads to

$$t^{asp}(\mathcal{E}, \mathcal{B}, \epsilon) = \{\text{time}(4).\}.$$

Since $\mathcal{B}$ is *refusal sensitive*, Definition 5.6.11 (Page 167) secrecy is violated by this option. In particular, from the following subset of $V^3_{\mathcal{E},\mathcal{B}}$ follows attend_scm:

$$\begin{array}{ll}
\{\text{has\_secret}(\mathcal{E},\ c(\text{attend\_scm})). & \\
\ \text{time}(4). & \\
\ \text{at}(4) & \leftarrow \text{time}(4), \text{not time}(T'), 4 < T'. \\
\ \text{sa}(\text{query},\mathcal{B},\mathcal{E},c(\text{attend\_scm}),2). & \\
\ \text{refused}(\mathcal{E}, c(\text{attend\_scm})) & \leftarrow \text{sa}(\text{query}, \mathcal{B}, \mathcal{E}, c(\text{attend\_scm}), 3), \\
 & \quad \text{not sa}(\text{answer},\mathcal{E},\mathcal{B},V',4), \\
 & \quad \text{at}(4), \text{related}(c(\text{attend\_scm}), V'), \\
 & \quad 4 \geqslant 2 + 2.
\end{array}$$

$$\begin{array}{lll} \mathsf{holds(c(attend\_scm))} & \leftarrow & \mathsf{has\_secret}(\mathcal{E}, \mathsf{c(attend\_scm)}), \\ & & \mathsf{refused}(\mathcal{E},\mathsf{c(attend\_scm)}). \\ \mathsf{attend\_scm} & \leftarrow & \mathsf{holds(c(attend\_scm))}.\} \subset V^3_{\mathcal{E},\mathcal{B}} \end{array}$$

Agent $\mathcal{E}$ has not answered the query such that it holds

$$\mathsf{not\ sa}(\mathsf{answer}, \mathcal{E}, \mathcal{B}, V', 4),$$

with $V' \in \{\mathsf{c(attend\_scm)}, \mathsf{c}(\neg\mathsf{attend\_scm})\}$. It follows that $\mathcal{E}$ has refused to answer the sensitive query for attend_scm, represented by the meta-information literal $\mathsf{refused}(\mathcal{E}, \mathsf{c(attend\_scm)})$. It further follows that the meta-information literal holds(c(attend_scm)) holds and is used by the linking rule to infer that attend_scm holds. Therefore it holds that

$$\mathsf{attend\_scm} \in \cap \mathcal{AS}(V^{3p}_{\mathcal{E},\mathcal{B}} *^{\mathsf{asp}} t^{\mathsf{asp}}(\mathcal{D}, \mathcal{A}, a)).$$

such that the secret is violated. That is, the third option would violate secrecy by the meta-inference that results from not answering the query of $\mathcal{B}$ for a sensitive formula of $\mathcal{E}$.

The violation sets for the considered options are:

$$\begin{aligned} & \mathsf{vioAfter}(\mathcal{S}^3, V^{3p}_{\mathcal{E},\mathcal{B}}, \alpha(\mathsf{sacted}(\mathsf{answer}, \mathcal{B}, \mathsf{c(attend\_scm)}))) \\ = {} & \mathsf{vioAfter}(\mathcal{S}^3, V^3_{\mathcal{E},\mathcal{B}}, \alpha(\mathsf{not\_acted})) \\ = {} & \{(\mathsf{attend\_scm}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}, \mathcal{B})\} \end{aligned}$$

and

$$\begin{aligned} & \mathsf{vioAfter}(\mathcal{S}^3, V^{3p}_{\mathcal{E},\mathcal{B}}, \alpha(\mathsf{sacted}(\mathsf{answer}, \mathcal{B}, \mathsf{c}(\neg\mathsf{attend\_scm})))) \\ = {} & \{\emptyset\} \end{aligned}$$

Hence, we get the following preferences with respect to secrecy:

$$\begin{array}{ll} & \alpha(\mathsf{sacted}(\mathsf{answer}, \mathcal{B}, \mathsf{c(attend\_scm)})) \\ =^s_{\mathcal{K},\mathcal{E}} & \alpha(\mathsf{not\_acted}) \\ \preceq^s_{(\mathcal{K},\mathcal{E})} & \alpha(\mathsf{sacted}(\mathsf{answer}, \mathcal{B}, \mathsf{c}(\neg\mathsf{attend\_scm}))) \end{array}$$

There is exactly one maximal element according to the $\preceq^s_{(\mathcal{K},\mathcal{E})}$ relation. Since the lexicographic aggregation operator with highest priority given to the $\preceq^s_{(\mathcal{K},\mathcal{E})}$ relation is used, the maximal element of the $\preceq_{(\mathcal{K},\mathcal{E})}$ relation is the same.

Hence the new set of intentions of $\mathcal{E}$ is

$$\begin{aligned} \mathfrak{I}^3_{\mathcal{E}} = {} & \{\sigma(\max_{\preceq_{(\mathcal{K},\mathcal{E})}}(\mathsf{options}))\} = \\ & \{\mathsf{sacted}(\mathsf{answer}, \mathcal{B}, \mathsf{c}(\neg\mathsf{attend\_scm}))\}. \end{aligned}$$

<u>act</u>: The action that $\mathcal{E}$ executes is

$$\begin{aligned} & \alpha(\mathsf{sacted}(\mathsf{answer}, \mathcal{B}, \mathsf{c}(\neg\mathsf{attend\_scm}))) = \\ & \quad \langle \mathcal{E}, \mathcal{B}, \mathsf{answer}, \neg\mathsf{attend\_scm} \rangle. \end{aligned}$$

<u>$\mathcal{K}^{3p}_{\mathcal{E}} \circ^a \langle \mathcal{E}, \mathcal{B}, \mathsf{answer}, \neg\mathsf{attend\_scm} \rangle$:</u>

$\underline{\circ^{a}_{\mathfrak{B}}:}$ $\mathcal{E}$ changes its views to reflect the changes implied by the action it just executed. The resulting sets of answer sets are:

$AS(V^{3}_{\mathcal{E},W}) = \{\{$attend_scm,
has_secret($\mathcal{E}$, attend_scm),
sa(query, $\mathcal{E}$, c(day_off), 1),
sa(query, $\mathcal{B}$, c(attend_scm), 3),
sa(answer, $\mathcal{E}$, c($\neg$attend_scm), 3),

attend_work,
~~open_query($\mathcal{B}$, c(attend_scm)),~~
~~open_sens_query($\mathcal{B}$, c(attend_scm)),~~
at(3), ...$\}\}$

and

$AS(V^{3}_{\mathcal{E},\mathcal{B}}) = \{\{$ has_secret($\mathcal{E}$, attend_scm),
sa(query, $\mathcal{E}$, c(day_off), 2),
sa(query, $\mathcal{B}$, c(attend_scm), 2),
sa(answer, $\mathcal{E}$, c($\neg$attend_scm), 4),
$\neg$attend_scm,

~~attend_work,~~
open_query($\mathcal{E}$, c(day_off)),
day_off,
$\neg$attend_work,
at(4), ...$\}\}$

The shown changes in the world-view of $\mathcal{E}$ reflect that after responding with $\neg$attend_scm to $\mathcal{B}$, agent $\mathcal{E}$ has answered the sensitive query. The fact $\neg$attend_scm. is added to its view on agent $\mathcal{B}$, such that day_off follows. From day_off again $\neg$attend_work follows. The resulting epistemic state is safe, since the only secret is not violated.

$\underline{t = 4}$:

$\underline{\mathcal{K}^{2}_{\mathcal{B}} \circ \langle \mathcal{E}, \mathcal{B}, \mathsf{answer}, \neg\mathsf{attend\_scm} \rangle}$:

$\underline{\circ_{\mathfrak{B}}:}$ Agent $\mathcal{B}$ revises its beliefs on reception of the percept. The resulting sets of answer sets are:

$AS(V^{4_p}_{\mathcal{B},W}) = \{\{$ has_secret($\mathcal{E}$, attend_scm),
sa(query, $\mathcal{E}$, c(day_off), 2),
sa(query, $\mathcal{B}$, c(attend_scm), 2),
sa(answer, $\mathcal{E}$, c($\neg$attend_scm), 4),
$\neg$attend_scm,

~~attend_work,~~

$\quad\quad\quad\quad\quad\quad\quad\quad$ open_query($\mathcal{E}$, c(day_off)),
day_off,
$\neg$attend_work,
at(4), ...}}

and

$AS(V^{4p}_{\mathcal{B},\mathcal{E}}) = \{\{$ has_secret($\mathcal{E}$, attend_scm),
sa(query, $\mathcal{E}$, c(day_off), 1),
sa(query, $\mathcal{B}$, c(attend_scm), 3),
sa(answer, $\mathcal{E}$, c($\neg$attend_scm), 3),
$\neg$attend_scm,

attend_work,
at(3), ...}}

Agent $\mathcal{B}$ receives the answer to its query, which adds $\neg$attend_scm. to its world-view. From $\neg$attend_scm follows day_off, and from day_off follows $\neg$attend_work.

$\underline{\circ_{\mathfrak{D}}}$:
The current belief set of $\mathcal{B}$'s world-view contains

open_query($\mathcal{E}$, c(day_off)

such that the desire operator results in the set of desires:

$$\mathfrak{D}^4_{\mathcal{B}} = \{(\{\text{answered}(\mathcal{E}, c(\text{day\_off})\}, 0.6), (\text{not\_acted}, 0)\}$$

$\underline{\circ_{\mathfrak{I}}}$:
The options for the satisfaction of the intention resulting from the maximally motivated desire, answered($\mathcal{E}$, c(day_off), are

sacted(answer, $\mathcal{E}$, c(day_off)),
sacted(answer, $\mathcal{E}$, c($\neg$ day_off)), and
not_acted.

All three actions are secrecy preserving, since $\mathcal{B}$ has no secret. The most informative, because it is honest, and therefore maximally preferred action of these according to $\preceq^a_{\mathcal{K}}$ is the first one:

$$\mathfrak{I}^4_{\mathcal{B}} = \{\sigma(\max_{\preceq_{(\mathcal{K},\mathcal{E})}}(\text{options}))\} = \{\text{sacted}(\text{answer}, \mathcal{E}, c(\text{day\_off}))\}$$

$\underline{\text{act}}$: The action that $\mathcal{B}$ executes is

$$\alpha(\text{sacted}(\text{answer}, \mathcal{E}, c(\text{day\_off}))) = \langle \mathcal{B}, \mathcal{E}, \text{answer}, \text{day\_off} \rangle.$$

$\underline{\mathcal{K}^{4p}_{\mathcal{B}} \circ^a \langle \mathcal{B}, \mathcal{E}, \text{answer}, \text{day\_off} \rangle}$:

$\underline{\circ^{a}_{\mathfrak{B}}}$: The resulting sets of answer sets are:

$AS(V^4_{\mathcal{B},W}) = \{\{$ has_secret($\mathcal{E}$, attend_scm),
sa(query, $\mathcal{E}$, c(day_off), 2),
sa(query, $\mathcal{B}$, c(attend_scm), 2),
sa(answer, $\mathcal{E}$, c($\neg$attend_scm), 4),
$\neg$attend_scm,
sa(answer, $\mathcal{B}$, c(day_off), 4),

~~open_query($\mathcal{E}$, c(day_off)),~~
day_off,
$\neg$attend_work,
at(4), $\ldots\}\}$

and

$AS(V^4_{\mathcal{B},\mathcal{E}}) = \{\{$ has_secret($\mathcal{E}$, attend_scm),
sa(query, $\mathcal{E}$, c(day_off), 1),
sa(query, $\mathcal{B}$, c(attend_scm), 3),
sa(answer, $\mathcal{E}$, c($\neg$attend_scm), 3),
$\neg$attend_scm,
sa(answer, $\mathcal{B}$, c(day_off), 5),
day_off,

~~attend_work,~~
$\neg$attend_work,
at(5), $\ldots\}\}$.

The resulting epistemic state is safe since there is no secret.

$\underline{t = 5}$:

$\underline{\mathcal{K}^3_{\mathcal{E}} \circ \langle \mathcal{B}, \mathcal{E}, \text{answer}, \text{day\_off} \rangle}$:

$\underline{\circ_{\mathfrak{B}}}$: The resulting sets of answer sets are:

$AS(V^{5_p}_{\mathcal{E},W}) = \{\{$ attend_scm,
has_secret($\mathcal{E}$, attend_scm),
sa(query, $\mathcal{E}$, c(day_off), 1),
sa(query, $\mathcal{B}$, c(attend_scm), 3),
sa(answer, $\mathcal{E}$, c($\neg$attend_scm), 3),
sa(answer, $\mathcal{B}$, c(day_off), 5),
day_off,

~~attend_work,~~
$\neg$ attend_work,
at(5), $\ldots\}\}$.

and

$AS(V^{5_p}_{\mathcal{E},\mathcal{B}}) = \{\{$ has_secret($\mathcal{E}$, attend_scm),

$\mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{day\_off}), 2),$
$\mathsf{sa}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{attend\_scm}), 2),$
$\mathsf{sa}(\mathsf{answer}, \mathcal{E}, \mathsf{c}(\neg\mathsf{attend\_scm}), 4),$
$\neg\mathsf{attend\_scm},$
$\mathsf{sa}(\mathsf{answer}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off}), 4),$
$\mathsf{day\_off},$

~~$\mathsf{open\_query}(\mathcal{E}, \mathsf{c}(\mathsf{day\_off})),$~~
$\neg\ \mathsf{attend\_work},$
$\mathsf{at}(4), \ldots\}\}$

Agent $\mathcal{E}$ receives the answer informing it that it has the day off. Consequently it follows $\neg\mathsf{attend\_work}$. The resulting epistemic state is safe, since the only secret is not violated.

$\underline{\circ_{\mathfrak{D}}}$: The only belief-desire couplings whose condition is satisfied is the one for the desire not to act, hence

$$\mathfrak{D}^5_{\mathcal{E}} = \{(\mathsf{not\_acted}, 0)\}.$$

$\underline{\circ_{\mathfrak{I}}}$: The maximally motivated desire is not to act and therefore the considered intention. The know-how of the agent defines the same intention to be the only option to realize it:

$$\mathfrak{I}^5_{\mathcal{E}} = \{\mathsf{not\_acted}\}.$$

$\underline{\mathtt{act}}$: The agent's action is therefore

$$\alpha(\mathsf{not\_acted}) = \epsilon.$$

$\underline{\mathcal{K}^{5p}_{\mathcal{E}} \circ^{a} \epsilon}$:

The resulting sets of answer sets are:

$$AS(V^5_{\mathcal{E},W}) = \{\{\ \mathsf{attend\_scm},$$
$\mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}),$
$\mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{day\_off}), 1),$
$\mathsf{sa}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{attend\_scm}), 3),$
$\mathsf{sa}(\mathsf{answer}, \mathcal{E}, \mathsf{c}(\neg\mathsf{attend\_scm}), 3),$
$\mathsf{sa}(\mathsf{answer}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off}), 5),$
$\mathsf{day\_off},$

$\neg\ \mathsf{attend\_work},$
$\mathsf{at}(5), \ldots\}\}.$

and

$$AS(V^5_{\mathcal{E},\mathcal{B}}) = \{\{\ \mathsf{has\_secret}(\mathcal{E}, \mathsf{attend\_scm}),$$
$\mathsf{sa}(\mathsf{query}, \mathcal{E}, \mathsf{c}(\mathsf{day\_off}), 1),$
$\mathsf{sa}(\mathsf{query}, \mathcal{B}, \mathsf{c}(\mathsf{attend\_scm}), 3),$

$\mathsf{sa}(\mathsf{answer}, \mathcal{E}, \mathsf{c}(\neg \mathsf{attend\_scm}), 3),$
$\neg \mathsf{attend\_scm},$
$\mathsf{sa}(\mathsf{answer}, \mathcal{B}, \mathsf{c}(\mathsf{day\_off}), 5),$
$\mathsf{day\_off},$

$\neg\ \mathsf{attend\_work},$
$\mathsf{at}(6), \ldots\}\}$

The resulting epistemic state is safe, the only secret is not violated.

At this point it is clear that in the following both agents continue to exchange empty actions and that their views will not change apart from the representation of time. Hence $\mathcal{E}$ has the day_off and has preserved its secrets. ♢

We have just shown that and how we can use our *secrecy ASP BDI$^+$ agent model* to simulate the motivating example about the *strike committee meeting* that we introduced in the introduction and formalized throughout this thesis. In the next section we discuss the related approaches to the work we presented in this chapter. Then we conclude.

5.7 RELATED WORK

Secrecy (also called *confidentiality* or *privacy* in other works) has been studied intensively, foremost from a theoretical perspective on the specification and verification of secrecy preserving computing systems. In this domain of computer security several definitions of secrecy properties have been proposed, e. g., [176, 177, 120, 221]. Later, general frameworks for the formulation of such properties have been developed, most notably the MAKS framework [174] and the *runs-and-systems* framework [121]. Modal logics have also been used to reason about secrecy, see e. g., [11, 232]. Also the runs-and-systems framework has a modal logical interpretation on the basis of *interpreted systems* [121]. The runs-and-systems framework is the most general framework, and the most appropriate for multi-agent systems. We elaborate on the relation of our framework to the runs-and-systems framework in Section 5.7.1.

The works on the formal specification and verification of secrecy do not provide constructive approaches for the design and the implementation of secrecy preserving systems. Constructive approaches to secrecy preservation in logic based systems have been developed mainly for the problem of *secrecy preserving query answering*, also called *controlled query evaluation*, or in a more general form *controlled interaction execution*. The general question hereby is how an information system or knowledge base can answer queries, being as informative as possible, while preserving secrecy. Hereby, the *inference problem* [93] is considered. In contrast to the problem of *access control* that demands that the direct access to sensitive information has to be prevented, the *inference problem* demands that the collection of sufficient information to infer the sensitive information has to be prevented. In the database setting the *controlled query evaluation (CQE)* framework [213] considers ways to preserve secrecy while answering queries from a user to a database. This framework has been extensively studied and extended by Biskup et al., see e. g., [31, 35, 39, 33, 37, 34, 38, 40], for an overview see [32]. In a more generalized form it is called *controlled interaction execution (CIE)* framework. We elaborate on the relation of the CIE framework to our work in Section 5.7.2. We discuss other works in the field of secrecy preserving query answering that are interesting with respect to our work in Section 5.7.3.

Only few works on the consideration of secrecy in agent theory have been conducted. These are focused on the consideration of the problem of secrecy preserving query answering and secrecy in some other specific problem areas have also been considered for agent systems. Another concept from agent theory that is related to the work we presented here are maintenance goals. We discuss these approaches in Section 5.7.4.

5.7.1 *Runs-and-systems base secrecy*

The most general and accepted framework for definitions of secrecy in multiagent systems is the one of Halpern and O'Neill [121]. It generalizes several other notions of secrecy and forms a basis for discussions of general issues of secrecy. In [36] Biskup and Tadros generalized the notions of Halpern and O'Neil by their policy-based notion of secrecy. In this section we discuss the runs-and-systems framework and notions of secrecy based on it in relation to our work.

In [121] Halpern and O'Neill base their work on secrecy on the runs-and-systems framework [89]. A *system* $\mathcal{R}$ is a set of runs r. A run is a sequence of global states. Global states of a system are identified using the two dimensions of runs r and time-points m such that a point is given by (r, m) and the corresponding global state is denoted by $r(m)$. A global state consists of all local states of all agents $\mathsf{Ag} = \{\mathcal{X}_1, \dots, \mathcal{X}_n\}$, such that

$$r(m) = (s_{\mathcal{X}_1}, s_{\mathcal{X}_2}, \dots, s_{\mathcal{X}_n}).$$

The local state of agent $\mathcal{X}$ in a global state $r(m)$ is given by $r_{\mathcal{X}}(m)$. The set of all points of a system $\mathcal{R}$ is denoted by $\mathcal{PT}(\mathcal{R})$. The view of agent $\mathcal{X} \in \mathsf{Ag}$ at point (r, m) on the system $\mathcal{R}$, called *$\mathcal{X}$-information set*, is defined as the set of points that it considers possible in (r, m), i. e., all points that are consistent with the local state of the agent:

$$\mathcal{K}_{\mathcal{X}}(r, m) = \{(r', m') \in \mathcal{PT}(\mathcal{R}) \mid r'_{\mathcal{X}}(m') = r_{\mathcal{X}}(m)\}.$$

In [121] several qualitative notions of secrecy are presented which all base on the principle that an agent[2] $\mathcal{D}$ maintains secrecy with respect to an agent $\mathcal{A}$ if and only if for all possible points $(r, m) \in \mathcal{PT}(\mathcal{R})$ agent $\mathcal{A}$ cannot rule out secrecy relevant information $I_{\mathcal{D}}$ of $\mathcal{D}$. The idea is that $I_{\mathcal{D}}$ describes a (secrecy-)relevant property of agent $\mathcal{D}$ that needs to be protected. The strongest notion of secrecy is the one of total secrecy in which $I_{\mathcal{D}}$ is the set of all possible local states of $\mathcal{D}$.

Definition 5.7.1 (Total Secrecy [121]). Agent $\mathcal{D}$ maintains *total secrecy* with respect to $\mathcal{A}$ in system $\mathcal{R}$ if, for all points (r, m) and $(r', m') \in \mathcal{PT}(\mathcal{R})$ it holds that $\mathcal{K}_{\mathcal{A}}(r, m) \cap \mathcal{K}_{\mathcal{D}}(r', m') \neq \emptyset$.

This notion of secrecy inhibits any information flow and, as stated in [121], "for almost any imaginable system, it is, in fact, too strong to be useful."

Example 5.7.2. In our example, total secrecy would imply that *Beatriz* is not allowed to believe that *Emma* knows her own name since all information is declared secrecy relevant. ◇

Several weaker notions of secrecy are presented in [121] that restrict the amount of relevant information $I_{\mathcal{D}}$ of agent $\mathcal{D}$. In *total f-secrecy* relevant information is defined by relevant values of $\mathcal{D}$ such that not the entire local state is relevant. In *C-secrecy* an $\mathcal{A}$ -allowability function explicitly defines a set of points $\mathcal{A}$ is allowed to rule out in each point. The most general definition of secrecy is *policy-based secrecy* presented in [36]. In policy-based secrecy the relevant information of agent $\mathcal{D}$ is defined by a set $\mathcal{I}_{\mathcal{D}}$ of sets of $\mathcal{D}$ -information sets $I_{\mathcal{D}}$, i. e., $\mathcal{I}_{\mathcal{D}} = \{I^1_{\mathcal{D}}, \ldots, I^l_{\mathcal{D}}\}$. The construction via sets of information sets allows for a more expressive formulation of relevant information. It can be expressed that out of a set $I_{\mathcal{D}}$ some local states can be excluded but not all, which allows to formulate disjunctive of information. That is, every set of $\mathcal{D}$ -information sets characterizes some relevant property and the set of all relevant properties of agent $\mathcal{D}$. This is formalized by a $\mathcal{D}$ -possibility policy.

Definition 5.7.3 ($\mathcal{D}$ -Possibility Policy [36]). A *$\mathcal{D}$ -possibility policy* policy is a function:

$$\mathcal{PT}(\mathcal{R}) \to \mathcal{P}(\mathcal{P}(\mathcal{P}(\mathcal{PT}(\mathcal{R}))))$$

such that $\mathrm{policy}(r, m) := \{\mathcal{I}^1_{\mathcal{D}}, \mathcal{I}^2_{\mathcal{D}}, \ldots\}$ contains sets $\mathcal{I}_{\mathcal{D}}$ of $\mathcal{D}$ -information sets.

Policy-based secrecy is then defined as follows.

2 In [121] the attacking agent $\mathcal{A}$ is denoted by i and the defending $\mathcal{D}$ by j.

Definition 5.7.4 (Policy-based Secrecy [36]). If policy is a $\mathcal{D}$ -possibility policy, agent $\mathcal{D}$ maintains policy-based secrecy with respect to agent $\mathcal{A}$ in $\mathcal{R}$ if, for all points $(r, m) \in \mathcal{PT}(\mathcal{R})$ and for all $\mathcal{I}^k_{\mathcal{D}} \in \mathrm{policy}(r, m)$:

$$\mathcal{K}_{\mathcal{A}}(r, m) \cap \bigcup_{I_{\mathcal{D}} \in \mathcal{I}^k_{\mathcal{D}}} I_{\mathcal{D}} \neq \emptyset$$

That is, no property of $\mathcal{D}$ characterized by an $\mathcal{I}^k_{\mathcal{D}}$ should be ruled out by another agent. Note that each $I_{\mathcal{D}}$ and each $\mathcal{I}^k_{\mathcal{D}}$ is dependent on the considered point (r, m).

C-secrecy and total f-secrecy have been shown to be special cases of policy-based secrecy in [36]. It has been shown in [121] that separability [178] and generalized non-interference [176] are stricter than C-secrecy and total f-secrecy, which are special cases of policy-based secrecy. In [36] the close relation of policy-based secrecy to opaqueness properties of function views as defined in [134] is shown. Hence, the most expressive definition of runs-and-systems based secrecy is the one of policy-based secrecy. In [155] we showed that and how we can express policy-based secrecy in an epistemic agent-based system.

There are several fundamental differences between the concept of runs-and-systems based secrecy and ours. Most strikingly the runs-and-systems model models the entire agent systems from a global perspective and for all possible global states and evolutions thereof. Our notion of agent-based secrecy is focused on the local state of an agent. Further, in runs-and-systems based notions of secrecy the defending agent $\mathcal{D}$ needs to have complete information about the attacking agent $\mathcal{A}$ in order to satisfy runs-and-systems based notions of secrecy. This is unrealistic in any practical multiagent system. Moreover, in practical multiagent systems it seems natural that several agents have secrets and should be able to preserve secrecy. In particular $\mathcal{A}$ should be designed symmetrically and should be able to have secrets with respect to $\mathcal{D}$. Moreover, both should be able to preserve secrecy at the same time. This is impossible if both have a complete view on the other. In our notion of agent-based secrecy the defending agent can have an uncertain and incomplete view on the information available to each attacking agent and the preservation of secrecy is dependent on the view $\mathcal{D}$ has on the attacking agents. This corresponds to a realistic setting of an autonomous agent in a multi-agent system.

Secrecy in the runs-and-systems model is preserved if an attacking agent can not exclude all states of a defined set. This resembles a skeptical inference behavior of the attacking agent. Our notion of agent-based secrecy is explicitly based on families of belief operators that resemble different inference behaviors. Each secret can be protected against a different belief operator. Moreover we consider the weakening of secrets and violations to different degrees on the basis of the order among different belief operators.

Another difference between the notion of runs-and-systems based secrecy and our notion of agent based secrecy is the definition of secret information. The information to be kept secret in runs-and-systems based secrecy is implicit. The relevant information is specified and does not represent secrets but information not to be ruled out, that is, information agent $\mathcal{A}$ has to consider possible. In comparison to the notion of secrets in our framework this means that all information is to be kept secret in which some relevant information is ruled out. Formally, this means that if $\mathcal{I} = \{I_1, \dots I_l\} \subseteq \mathcal{P}(\mathcal{PT}(\mathcal{R}))$ characterizes relevant information, the set of secret information is given by

$$\Gamma(\mathcal{I}) = \{I \mid \exists I' \in \mathcal{I}, I = \mathcal{PT}(\mathcal{R}) \setminus I'\}.$$

This set of secrets in form of information-sets forms the basis for the secrets of agent $\mathcal{D}$. The formulation of secrets in policy-based secrecy specifies the information an attacking agent shall believe. Hence if we consider our running example we have.

Example 5.7.5. *Beatriz* should not be able to rule out that *Emma* does not attend the *strike committee meeting*. Hence $\neg\mathsf{attend\_scm}$ is security relevant information. ◇

5.7.2 *Controlled interaction execution*

In the framework for *controlled interaction execution* (CIE) [32] a logic-based client-server system with a confidentiality policy is considered. The policy contains *potential secrets* that are formulae that should not be inferred by an attacker if they are satisfied in the defender's knowledge base. Clients pose queries, and update-requests, to the server, which reacts to these. A censor component inspects the potential answer of the information system to a query, given the history of queries and answers, and the client's assumed background knowledge. In case of secrecy violation by the original answer of the information system the censor modifies the answer. In CIE considered options for modifications are lying and refusing. An answer is a lie if it is an answer value for the query that is inconsistent with the actual database instance. In the case of a refusal, a refusal notification is sent as a dedicated answer value, e. g., refuse. Various instances of this setting have been studied that differ in several parameters. One is the type of information system that is considered, for instance propositional and relational, complete and incomplete databases are considered. Others parameters are the considered interaction language (e. g., open/closed queries, update requests), the policy language (e. g., propositional or modal logic) and the types of censor (lying, refusal, combined).

In the CIE setting several assumptions are made. One assumption is an open design. That is, the attacker (client) is assumed to know the potential secrets of the defender, i. e., the potentially secret formulae but not whether they hold in the knowledge base of the defender.

Further, the attacker is assumed to know the particular algorithm (including any distortion strategy) of the defender (censor / information system). The attacker is assumed to be a skeptical reasoner with unlimited computational power. These assumptions allow the attacker to apply meta-inferences on the basis of the defender's reactions. The effectiveness of the devised censor functions depends crucially on these assumptions.

In our approach we do not make any of these assumptions in general. We explicitly model the information available to the attacker and we explicitly model its reasoning capabilities. This way we capture a more realistic and accurate attacker model. The resulting approach is less restrictive and more flexible.

The considerations with respect to meta-inference in the CIE setting are interesting in our setting as well. In the following we summarize the main results for the fundamental strategies of a censor in the CIE framework [32] and then relate them to our approach.

UNIFORM LYING The *uniform lying* censor returns a different answer value then the actual answer value of the database if returning the actual value would violate secrecy. For the *uniform lying* censor it has been shown that, in order to avoid hopeless situations, the protection of the secret formulae is not sufficient to guarantee preservation of secrecy; but protecting the disjunction of the secrets as well guarantees preservation of secrecy.

UNIFORM REFUSING The *uniform refusing* censor returns a notification that the last request will not be answered if returning the correct answer would violate secrecy. For the *uniform refusing* censor it has been shown that it allows meta-inferences if it only refuses to answer if the truthful answer would violate secrecy. This is due to the assumption of an open design by which the attacker knows the censoring algorithm of the defender and its potential secrets. That is, the attacker knows that if the defender refuses to answer, then answering truthfully would violate a secret. From this it can possibly infer that a potential secret is actually true. It has been shown that this is not possible if the censor refuses to answer a query whenever answering with an answer-value could violate one of its potential secrets, independently of whether they actually hold or not.

LYING+REFUSING A *combined lying and refusal censor* is also devised. It combines the advantage of the uniform lying censor that not all alternative answer values and their effect on secrecy have to be considered (as has to be for the uniform refusing censor) with the advantage of the uniform refusing censor that the disjunction of potential secrets does not have to be protected (as has to be for the uniform lying censor). It does not protect the disjunction of secret formulae and lies if answering the truthful evaluation would violate secrecy and the lie would not. If the truthful evaluation and all possible lied answer values would violate secrecy the censor refuses to answer. Since the refusal is independent of the actual answer value, it does not allow the meta-inference described in the context of the uniform refusing censor.

In our approach the agents can perform arbitrary sequences of speech acts, of which we considered the types

inform, query, and answer.

Further, they can "not act" by performing the empty action. The agent always generates and considers a set of alternative options for action, evaluates them and performs a maximally preferred one according to its evaluation. It evaluates all options with respect to secrecy. Hereby it is not restricted to answer with an evaluation of a query to a query but might consider to send some other answer or inform action, or respond with a query. It evaluates all options with respect to secrecy, not only by considering an action to be secrecy violating or not but by classifying the action with respect to secrecy. This evaluation gives the means to minimize the violation of secrecy if it is unavoidable, or necessary to accomplish an important goal.

The scenario of query answering and the strict preservation of secrecy is hence a special case of our scenario. If we model the query answering scenario the defending agent receives a query, generates the goal to answer this query, and then considers its possible actions to realize this goal. We consider an answer to a query as refused if the empty action is performed after a query has been received. We assume that an agent always considers not to act (resembled by the empty action) as an option. It evaluates if the empty action is preferable over the other actions that are options. Our agents prefer to execute actions that violate secrecy as little as possible, and among these it prefers actions that are as informative as possible. When considering to answer a query for the evaluation of some formula ϕ the agent generates at least the following options to answer:

a) to answer truthfully with the evaluation of ϕ,

b) to answer with any alternative evaluation of ϕ,

c) to not answer by executing the empty speech act.

The first option should, in a cooperative setting, be the preferred one by the agent, if it is not violating secrecy. The second option is a lie and the third is seen as a refusal to answer. Hence, our agents always consider at least all of the options that are considered in the CIE setting. In our approach the CIE strategies are addressed as follows:

UNIFORM LYING We showed in Proposition 5.4.15 (Page 126) that if we restrict our scenario and agent to a propositional query-answer scenario we can prove that an agent in this scenario is secrecy preserving if and only if it protects the disjunction of secrets.

UNIFORM REFUSING We do not generally assume that the attacking agent has information about the potentially secret formulae of the defending agent and its answering behavior. In our approach we protect secrecy on the basis of the attacker model of the defending agent. We developed attacker models on the basis of answerset programming in Section 5.6. We showed in Section 5.6.4 that if we model an attacker that has this information we can model meta-inferences from a refusal with respect to questions for sensitive formulae. If we model an attacking agent that knows that the defending agent refuses to answer any queries with respect to sensitive formulae in any case, then it cannot perform meta-inferences from the refusal, as for the propositional case.

LYING+REFUSING We defined a preference relation on options in Section 5.5 and in particular a preference order $\preceq^{a}_{\mathcal{K}}$ based on the informativity of the options in (5.5.5) (Page 132) that prefers the empty action, i. e., refusing, over lying. If we change this relation to a relation that prefers lying over refusing we would get an agent that shows the same behavior as the Lying+Refusing censor.

We have illustrated how our approach of agent based secrecy relates to important aspects of the CIE approach, which is the most sophisticated approach in the field of *secrecy preserving query answering*. In the following section we consider other works on secrecy preserving query answering and interesting aspects of these.

5.7.3 *Secrecy preserving query answering*

In [224] the general problem of secrecy preserving query answering with multiple querying agents that pose queries to a single knowledge base is considered. The interaction of the querying agents is modeled by a communication graph. It is assumed that an agent shares all answers it received from the knowledge base with its direct successors in the graph, but it does not share answers received from other agents with another agent. The secrets are different for different agents. No background knowledge or reasoning capabilities of the agents are modeled. Secrecy is considered under the open world assumption and preserved by responding with *unknown* whenever responding with *yes* or *no* would violate secrecy. This is realized by constructing and maintaining a *secrecy envelope*. It contains for each agent a subset of the knowledge base such that if it is removed from the knowledge base the secret can not be inferred from the resulting knowledge base. Queries are answered according to the knowledge base from which the secrecy envelope with respect to the considered agent is subtracted. The setup used in this work is simpler than the one considered in our approach or in the CIE setting. The interesting aspect of this work, that is not present in other works, is the modeling of the interaction of several attacking agents. The used model on the basis of communications graphs is quite simple, but would make a good first step towards the extension of our agents to consider the interaction of several attacking agents.

In [79] multi-context systems are used to model a knowledge base and a set of users with background knowledge and secrets. The secrets of the users define the information to be kept secret from other agents. The authors show how a multi-context system can be constructed to compute privacy preserving answers to queries from users to the knowledge base. An instance based on default logic is exemplarily shown. Only a one-step process of computing answers for a given query is considered. The computed answers are guaranteed to be privacy preserving but are not necessarily maximal. This work considers a very simplified secrecy setting, but is interesting since it considers a general framework for the use of non-monotonic logics in the context of secrecy.

In [135] the authors consider the problem of privacy preserving data publishing for answer set programming. They show how an abduction framework for extended disjunctive logic programs can be used to compute a privacy preserving version of a given program with respect to a secrecy policy. This work considers an approach to secrecy that is based on modifying a knowledge base such that the resulting knowledge base can be directly used to answer any queries. That is, it considers a preprocessing approach to secrecy, in contrast to the dynamic approach that is considered in our approach and others. In a preprocessing approach potentially less information is available as in a dynamic approach for a specific query sequence. Further, it is not applicable for complex interacting agents in a dynamic environment. The interesting aspect of this work with respect to our work is its use of an answer set programming based approach to compute the secrecy preserving version of a logic program. The abductive approach determines the possible options to modify a logic program such that no secret is entailed, on the basis on a set of abducibles. In our approach the know-how of the agent represents the options for these modifications. As part of future work it would be interesting to see how the abductive approach could be used to determine options for action in the light of look ahead planning of an agent. That is, if the agent considers sequences of actions it could perform to achieve some goal it could determine options for these actions by means of abductive reasoning. It would be interesting to investigate how these approaches can be combined and to make use of the abductive approach in a dynamic setting.

5.7.4 *Secrecy and Agent models*

The design and implementation of secrecy aware agents has not been considered in the general form we present here. Different forms of secrecy have been considered for specialized application domains only. In the following we discuss the most relevant of these works.

Logic programming has also been considered in combination with agents and secrecy in [171]. There, secrecy is considered in a distributed abductive logic programming system. It models agents that have to collaboratively compute consistent answers to a query and protect their private information at the same time. Secrecy is expressed by means of the specification of *private* and *askable* literals which are used in a distributed abduction algorithm. That is, the considered problem is an access control problem rather than an inference problem. There is no attacker modeling, no considered inferences or meta-inferences. Also, no general agent model and general actions are considered.

In [238] and [102] secrecy is considered in distributed constraint satisfaction problems with application to distributed meeting scheduling. No explicit secrets are considered, but a quantitative assessment of privacy loss on the basis of number of possible states is defined which can be used to define secrecy vs. efficiency tradeoffs. No agent model is considered, but the assessment of privacy loss is somewhat related to our concept of the classification of actions with respect to secrecy.

The closest to the general consideration of secrecy in agent models are the works on the integration of the CIE approach, which we discussed above, into agent models. In [39] it is proposed to use a *censor* component, which is external to the agent cycle, to control the agent's actions prior to execution and modifying them if necessary. While this might be adequate for censoring the answers of a database, it is not adequate in agent theory since it contradicts the autonomy of an agent in its decision making. Moreover, it is very restricted in its possibilities since the censor is not involved in the deliberation or planning of the agent. This approach is discussed for a scenario of a negotiating agents. Moreover, the actual realization of this approach is left open in [39].

In her dissertation, [222], Cornelia Tadros considers a defending agent that reacts to iterated query, revision and update requests from an attacking agent. For the definition of secrecy similar assumptions as in the CIE setting are made, i. e., complete information and unlimited computational power of the attacker. In this context she considers non-monotonic reasoning formalisms and belief change operators from belief revision theory of the agents are considered. However, also there, the CIE approach is used as an external control instance of the agent that modifies the agent's actions. No complete agent model or agent-based notion of secrecy, as we consider in this work, is considered.

In contrast to these works that use a censor component to control the agent's behavior, we consider the design of secrecy aware agents. That is, we integrate secrecy into the agents' reasoning, deliberation and means-end reasoning processes to achieve general autonomous secrecy preserving agents apt to perform well in a dynamic setting.

Our integration of secrecy into the agent's deliberation and means-ends reasoning is comparable to the consideration of secrecy as a kind of *maintenance goal* of the agent. A maintenance goal is represented by a maintenance-condition, which the agent aims to keep satisfied. In most of works in agent theory and practical systems that consider maintenance goals only *reactive maintenance goals* [82] are considered. If the maintenance-condition of a reactive maintenance goal is violated the agent starts acting towards its recovery. For instance, if the battery level of a robot drops below a set threshold it moves to the charging station. This approach is obviously not appropriate for the preservation of secrecy. *Proactive maintenance goals* on the other hand are taken into consideration by the agent's reasoning and make anticipation of the actions of the agent necessary to prevent the violation of the maintenance-condition [81, 129]. This formulation seems to be more adequate to capture the preservation of secrecy. However, only little work on proactive maintenance goals exists. Most of these focus on the definition and formalization of different types of maintenance goals [17, 81, 129, 67] that can be elegantly formulated by means of modal logics, e. g., *linear temporal logic* formulae [67].

Few more practical approaches to proactive maintenance goals exist. In [129] proactive maintenance goals are considered for agents based on the agent programming language GOAL [130]. The authors specify the semantics of a maintenance goal on the basis of an *n-step look-ahead operator* which is assumed to be capable of determining the possible next n steps of the run of the agent. The actions of the agent are then constrained to those that maintain the condition that the maintenance-condition is not violated in any of the possible next n steps. Other works also rely on such a look-ahead operator, e. g., the *future* operator in [82]. However, no constructive approach for such an operator is described, it is assumed to be given.

Secrecy has not been considered as a maintenance goal and the existing approaches to maintenance goals only specify the semantics of maintenance goals abstractly and do not address the issue of the design of agents with proactive maintenance goals. The examples used in the literature consist of simple physical agents. As we have shown, agent based secrecy calls for an appropriate epistemic state of the agent that includes the specification of secrets, views of the information available to other agents. Belief change operators, different belief operators and an involved evaluation of options with respect to secrecy are needed. Moreover, in contrast to agents that perform ontic actions considered in the literature on maintenance goals, with respect to secrecy there do not generally exist actions that can be used to "undo" other actions. For example, in the literature on maintenance goals an *unload* action is considered that recovers from a state in which the maximum load of a container has been exceeded, or a *charge_battery* action that recovers from a state in which the battery level is too low.

In our approach of agent based secrecy we defined *secrecy aware* agents and *secrecy preserving* agents. The former are required to "do their best" to preserve, and recover, secrecy and the latter are demanded to never violate a secret by their actions. Our definition of a safe epistemic state can be seen as our maintenance-condition and our semantics with respect to a strictly secrecy preserving agent is that this condition is *always* satisfied. While this, agent based, definition is very weak from the point of view of the literature on secrecy it has been deemed too strong in the literature on maintenance goals for agents. The proposed weaker conditions, such as *always eventually* or *always after* k *steps*, in the latter field have mostly also been dismissed as being to strong in practice. In [82] it is argued that an agent should have proactive and reactive methods for maintenance goals available since it is highly unlikely that the proactive approach is capable of avoiding that the maintenance-condition is violated in practice. Our approach of *secrecy aware agents* captures both methods. The proactive part is captured since the agent simulates the effects of its actions before it performs them. This way the action selection of the agent is constrained. The reactive part is captured by the classification of action with respect to secrecy. If secrecy is violated the agent chooses the actions that recover secrecy as good as possible. Secrecy in our approach is a dynamic maintenance goal due to the change operator for secrets. In other works the dynamics of proactive maintenance goals are not addressed.

5.8 conclusion

In this chapter we presented an approach to secrecy from the perspective of an epistemic agent. In this perspective, the information of an agent is uncertain and incomplete, and the agent acts in a dynamic environment with other agents. The agent is interested in achieving its goals, and has to take care to violate its secrets as little as possible. We call such agents *secrecy aware agents*. These agents are *secrecy preserving*, if it is possible in their setting. This is mostly not the case in uncertain dynamic environments. In case this is not possible, the agent does its best to violates secrecy as little as possible, and it tries to restore secrecy if that is feasible. We presented a thorough theoretical, conceptional and practical account of secrecy from this perspective.

We used our general epistemic agent model, which we presented in Chapter 3, to define an epistemic agent model for secrecy. It makes use of non-monotonic reasoning and well founded belief change operators. We defined secrets that specify the reasoning behavior against which a formula shall be protected, based on the families of (non-monotonic) belief operators we introduced in Chapter 3. We used the credulity order on belief operators to compare the strength of secrets, and to weaken them. We defined a notion of secrecy preserving agents that are required to never perform a secrecy violating action. Then we elaborated on aspects of the change operators, which we formalized in Chapter 4, with respect to secrecy, and defined a set of postulates for these. We further developed a change operator for secrets that minimally weakens secrets.

We formulated the notion of a setting and a game theoretic formulation of winning strategies with respect to secrecy. We defined *sound states* as states in which the attacking agent does not have a winning strategy. Based on this definition we showed that the defending agent has to maintain a sound state in order to be secrecy preserving. We defined a setting for a typical query-answer scenario used in works on secrecy in databases. For this setting we showed that, as was shown in the literature on secrecy in databases, an agent is secrecy preserving if and only if it protects the disjunction of its potentially secret formulae. In our formulation of secrecy this means that, for this scenario, determining if the attacker has a winning strategy is realized by checking if the disjunction of secrets is preserved. Further, we formalized *plain settings* in which an agent does not have to consider future actions, and considered these in the following.

We defined a preference relation on the options for action of an agent that forms the basis for the agent's deliberation. The relation is constructed by an operator that aggregates several preference relations with respect to different aspects. In particular, we defined a lexicographic aggregation operator and two preference relations, one with respect to secrecy and one with respect to informativity. We based the preference relation with respect to secrecy on a numerical classification of actions. For secrecy aware agents that cannot preserve secrecy, the fine grained classification of possible actions with respect to the violation of secrecy is essential to not violate secrecy more than necessary. We formalized a set of principles for such a classification with respect to two dimensions of uncertainty of the defenders view on the attacker. One is the uncertainty of the world-view of the attacker, and the other is the uncertainty of the defender's view on the world-view of the attacker. These principles consider the amount of potentially simultaneously violated secrets as well as their degree of violation on the basis of the credulity order on belief operators. Further, we formalized principles for the intuitions that a defending agent should be the more cautious the more uncertain it is, and the more cautious the more credulous the reasoning behavior is against which it protects its secret formulae. We showed that these last two principles are implied by the first three principles. Moreover, we developed an algorithm and proved that it satisfies all principles. We also proved that the algorithm terminates and that it produces a complete classification.

We considered a complete agent model for secrecy aware agents on the basis of the *basic BDI$^+$ agent model* we presented in Chapter 3. We instantiated this agent model by use of ASP and the ASP change operator that we presented in Chapter 4. Based on ASP we elaborated a representation of the information available to communicating agents on two levels, the information level and the meta-information level. Further we developed patterns of meta-inferences based on the meta-information of the agents that lead to inferences on the information level. In particular, we formalized the meta-inference of an attacking agent if the defending agent does not answer a query with respect to a sensitive literal. This meta-inference makes it necessary that the defending agent considers to answer a query with respect to sensitive information. To realize such a behavior the agent has to consider secrecy already on the level of its motives. For this, we introduced motives to answer such queries in combination with meta-information level rules that determine if the agent has been queried for a sensitive literal. Further, we showed that and how we can model our *strike committee meeting* example that we introduced in the introduction and that we used for illustration throughout the thesis.

We discussed in detail the relation of our approach to related work. In particular, we discussed the relation to two important frameworks, the abstract framework for secrecy in multiagent systems by Halpern and O'Neill [121], and the work on controlled interaction execution by Biskup et al. [32]. Moreover, we discussed the relation to other interesting works on secrecy preserving query answering and pointed out interesting aspects for further work. We also discussed other approaches to secrecy for practical agents, and work on maintenance goals in multiagent systems.

To sum up, we developed a novel notion and framework of secrecy in multiagent systems that is agent-based and apt for the setting of agents with uncertain information in a dynamic environment. It formulates the notion of secrecy aware agents as the adequate and necessary general notion for this setting, in which agents often cannot be guaranteed to preserve secrecy. We established a sound theoretical basis for our framework and developed constructive approaches for the realization of secrecy aware, and secrecy preserving, agents. As a complete concrete instance we developed secrecy aware agents on the basis of an extended BDI model and a sophisticated ASP knowledge representation. Our work forms a substantial basis for further theoretical investigation as well as the implementation of secrecy aware agents.

6 THE ANGERONA FRAMEWORK

In this chapter we present the Angerona multiagent programming framework for the implementation of knowledge-based agents with a strong focus on flexibility, extensibility, and compatibility with diverse knowledge representation formalisms. It supports the development of epistemic agents, that is, agents based on logical formalisms for knowledge representation and reasoning. For these it supports the use of belief change operators based on belief change theory. Moreover, it facilitates the development of diverse agents with respect to their architecture and knowledge representation. It allows the formation of multiagent systems comprising heterogeneous agents which interact in a common simulated environment.

A variety of logical formalisms with different expressivity and computational properties have been developed for knowledge representation with the agent paradigm in mind, for an overview see, e. g., [55, 234]. Especially non-monotonic formalisms are designed to deal with incomplete information and to enable an agent to act in uncertain environments. Moreover, the field of research on belief change has been working on solutions on how to change an agent's beliefs in the light of new information for over 25 years already, for an overview see, e. g., [94]. Yet, very little of the approaches developed in these two fields of research are available in practical multiagent frameworks. The Angerona framework narrows the gap between theoretical considerations of knowledge representation, reasoning and belief change on the one side, and practical multiagent frameworks on the other side. Existing agent programming frameworks are also largely fixed to a certain agent model and knowledge representation formalism. The Angerona framework allows to easily implement many different agent models, it provides tools to implement these from scratch, it provides different variants and extensions of BDI agent models and allows to easily modify, mix and extend these.

The Angerona framework is based on the conceptual work on the epistemic agent model we elaborated in Chapter 3. In this model, an agent comprises an epistemic state and a functional component that can both be composed. This model is realized on the basis of a plug-in architecture. The epistemic state and the functional component are based on knowledge representation plug-ins and on operator plug-ins, and tied together by an *XML* based script language and configuration files. The plug-in for the functional component is based on the general Angerona operator interface and the corresponding operators provided by the angerona framework. For the knowledge representation plug-ins we build on the Tweety *library* for knowledge representation [227]. The Tweety library contains interfaces and implementations for diverse knowledge representation formalisms with inference and change operators for them. Tweety is under active development, in which we participated. It currently contains implementations for: first-order logic [55], ordinal conditional functions [190, 143], relational conditional logic [146], probabilistic conditional logic [205], relational probabilistic conditional logic [145], Markov logic [204], epistemic logic [89], description logic [12], deductive argumentation [26], structured argumentation frameworks [228], defeasible logic programming [105], probabilistic argumentation [225], answer set programming [112, 154], and more. For a comprehensive overview of the Tweety library see [227]. The Tweety library is also open source and available at *sourceforge*[1]. Angerona is open source and available at *github*[2].

The Angerona agent architecture can be freely defined by specifying the types of operators to be used and their order of execution. This way Angerona allows to easily design different types of agents. Not only the used language for knowledge representation can differ, but also to which amount an agent's functionality is logic based. It is, for instance, easily possible to realize the agent's deliberation and means-ends reasoning by Java operators and simple data components, or by simple Java operators which make use of logical formalisms, e. g., answer set programming (ASP) [112], ordinal conditional functions (OCF) [220], argumentation formalisms [24], or propositional logic or horn logic, or any other formalism from the Tweety library.

While the general Angerona framework allows for a high degree of flexibility it also allows to define partially instantiated plug-ins and default agent configurations, which represent sub-frameworks with more predefined structure and functionality. The latter might fix the general agent cycle by specifying the types of operators to be used and provide different implementations for these. Hence, the sub-frameworks provide more support for easy and rapid development of agents. We distinguish three different roles of users in the Angerona framework:

– the *core developer* that uses Angerona as a toolkit to define its own agent types;

– the *plug-in developer* that uses provided agent types, instantiates them with given plug-ins, and potentially implements its own plug-ins or modifies existing ones;

– and the *knowledge engineer* that defines the background and initial knowledge, and all other initial instances of the components of the agents.

1 http://sourceforge.net/projects/tweety/
2 https://github.com/Angerona

ANGERONA provides default implementations for BDI style agents and diverse extensions that can be modularly used to build agents. These realize the BDI^+ agent type presented in Section 3.5. This defines a sub-framework for the development of knowledge based BDI^+ agents in general and for those presented in this thesis in particular. We implemented almost all approaches presented in this thesis, and several others, in ANGERONA and ran simulations with them. These implementations include in particular an implementation of the BDI^+ agent model presented in Chapter 3, extensions of it towards the full know-how approach [229, 157] and the full motivation approach [159], and of the complete secrecy agent model, and the *ASP* instantiation of it we presented in Chapter 5. Also the change operators presented in Chapter 4 are implemented, among several others. Alternative instances that use propositional logic or ordinal conditional functions for knowledge representation, reasoning and change have been implemented in ANGERONA as well. Several, mostly secrecy, scenarios and simulations are available. ANGERONA also features a plug-in interface for different environments, with a communication environment for agents as defined in Section 3.1 implemented. A graphical user interface (GUI) allows the selection, execution, observation, and inspection of multi-agent simulations. The GUI can be extended by plug-ins to feature displays of specific knowledge representation formalisms, for instance dependency graphs [4]. The angerona framework has been used, and continually is used, and extended in several final thesis, e. g., in [3, 137, 19, 52, 131, 244], and in an internship financed by the German Academic Exchange Service (DAAD), see [78] for a report of the internship.

In the next section we introduce the main theoretical concepts of the ANGERONA framework. Following on this we describe how these are realized in the agent framework of ANGERONA. Then we describe the multiagent framework of ANGERONA in the following section. Afterwards, we describe how the secrecy aware BDI and ASP BDI agents we developed in Section 5.6 are implemented in the ANGERONA framework. Finally, we discuss our framework and its relation to other frameworks, and conclude.

6.1 AGENT FRAMEWORK

The ANGERONA agent framework is designed to implement the general agent model as introduced in Section 3.3. In this model an agent has an epistemic state $\mathcal{K}$ and a functional component ξ . The epistemic state can be a composition of different general epistemic components, and other components. Epistemic components are based on a knowledge representation formalism. This formalism also specifies the family of belief operators that can be used to infer the belief set for the given epistemic component, and the change operator for epis-

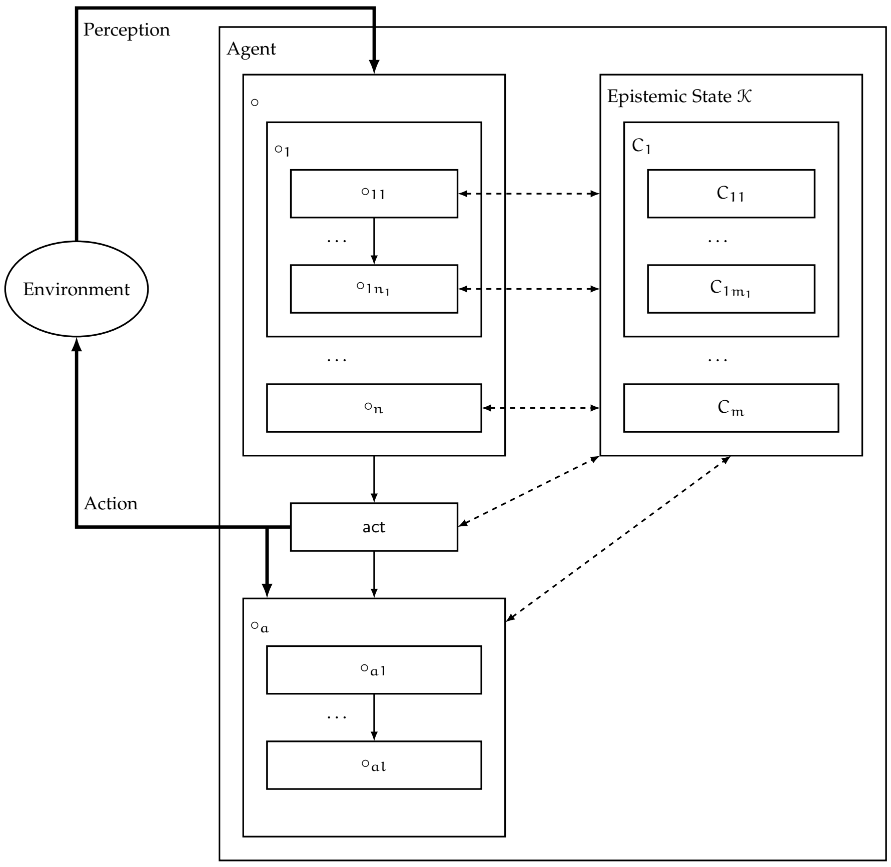


Figure 6.1.1: Compound agent model

temic components. The functional component comprises an ordered composition of operators that operate on and change the epistemic state of the agent. The resulting general perspective on an agent is illustrated in Figure 6.1.1, which is the same as Figure 3.4.2 (Page 49).

Angerona agents consist of *agent components* which can be *epistemic components*, i. e., epistemic components with associated operators, and other data components, or *functional components*, i. e., operators used in the agent cycle. Epistemic components are based on the *epistemic plug-in*. Operators for the agent cycle are based on the *operator plug-in*. For the realization of plug-ins in Angerona we use the *Java Simple Plugin Framework (JSPF)* [193].

Example 6.1.1. We model for instance the *BDI$^+$ agent instance* from Section 3.5.3. A BDI$^+$ agent state is a tuple $(\mathcal{K}_{\mathsf{BDI}^+}, \mathcal{E}_{\mathsf{BDI}^+})$ with the epistemic state being of the form

$$\mathcal{K}_{\mathsf{BDI}^+} = \langle \mathfrak{B}, \mathfrak{D}, \mathfrak{I}, \mathfrak{M}, \mathfrak{K}\mathfrak{H} \rangle.$$

This epistemic state is directly realized by means of corresponding epistemic components from epistemic plug-ins, for instance the ASP or the OCF plug-in. The functional component comprises the opera-

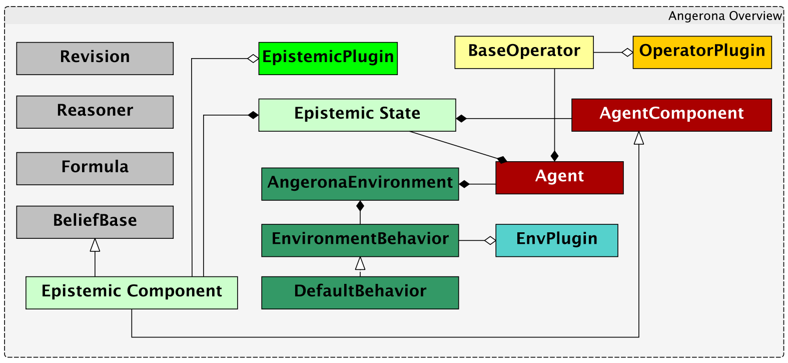


Figure 6.1.2: Simplified *UML* [207] class diagram of an Angerona agent

tors $\circ_{\mathfrak{B}}, \circ_{\mathfrak{D}}, \circ_{\mathfrak{I}}, \circ^{\mathfrak{a}}$, act. These are realized by corresponding operators from the operator plug-in. ◊

The class diagram in Figure 6.1.2 illustrates the realization of the conceptual model in the Angerona framework. An Angerona agent contains an epistemic state and a list of operators. An epistemic state consists of agent components. One type of agent components are epistemic components, which are defined via an epistemic plug-in. The epistemic plug-in implements the interfaces of the Tweety library, in particular those for an epistemic component, a belief base, a formula, a change operator and a belief operator. Different belief operators might be available for the same formalism. Different agents might use the same knowledge representation formalism but different belief operators.

Angerona implements the concept of epistemic components, the reasoning and change operators presented in this thesis. The belief operator families with ordering we introduced in Section 3.6 are implemented in Angerona by a belief operator family class. Several instances are also implemented, including the *ASP* instance. Each belief operator corresponds to the implementation of a *reasoner* of the Tweety library.

We implemented a full fledged epistemic plug-in for *ASP*. It is capable of using several *ASP* solver such as *clasp* [108], *DLV* [77, 85, 165], *DLV-Complex* [62], and *smodels* [184]. It provides parsers for the different language versions of ASP by means of *JAVACC* [147]. Different belief operators and belief change operators are implemented and can be used, in particular the *ASP* change operator presented in Section 4.3 is implemented and an *ASP* and argumentation based selection function on the basis of the approach presented in Section 4.2. Also the approaches presented in [75, 76, 152] are implemented.

Moreover, *ASP*-based version of know-how for hierarchical planning and reasoning about know-how as presented in [157] is implemented. For the visualization of ASPs as extended dependency graphs and explanations graphs, based on [4], are implemented as a GUI plug-in.

The agent cycle of an agent is realized by operators provided by operator plug-ins. Operators in Angerona exist on two fundamental levels of abstraction. *Operator types* represent types of operators, e. g., a change operator. By means of operator types we can define the agent cycle for an agent type without instantiating the concrete operators to be used. An operator is a class that implements a particular *operator type*, e. g., an ASP change operator. There might be several operators with the same operator type implemented, e. g., ASP change operators for the approaches [75, 76, 152, 154]. The *knowledge engineer* can select which operator shall be used for which of its agents in the respective agent configuration files.

The agent cycle of an Angerona agent is specified by means of the Angerona *Script Markup Language (*ASML*)*. *ASML* is an *XML* format, which features operator invocation and basic control structures to design sequences of these. It also supports access to variables in a given context, which is normally provided by the agent. For the operator invocation in *ASML* only the operator type is specified in the *ASML* file, the concrete operator instance is specified in the agent configuration file. Listing 1 shows a simple *ASML* example script for a BDI-style agent. The *ASML* language also features basic control structures such as assertions, conditions and loops. For a full reference of *ASML* see [162].

Listing 1: Simple BDI Agent Cycle in *ASML*

```xml
<asml-script name="BDICycle">
    <operator type="ChangeBeliefs">
        <param name="percept" value="$percept" />
        <param name="beliefs" value="$beliefs" />
        <output>beliefs</output>
    </operation>

    <operator type="ChangeDesires">
        <param name="beliefs" value="$beliefs" />
        <param name="desires" value="$desires" />
        <output>desires</output>
    </operation>

    <operator type="ChangeIntentions">
        <param name="desires" value="$desires" />
        <param name="intentions" value="$intentions" />
        <output>intentions</output>
    </operation>

    <execute action="$action" />
</asml-script>
```

By use of these the BDI cycle could be modified by replacing the simple invocation of the *Intention Change* operator in Listing 1 by a loop that iteratively refines the current plan until the next action is determined.

Listing 2: Loops and Conditions in *ASML*

```
<assign name="running" value="TRUE" />
<while condition="$running==TRUE">

        <operator type="Intention Change">
            <param name="plan" value="$plan" />
        <output>action</output>
        </operation>

        <conditional>
                <if condition="$action==null">
                        <operator type="SubgoalGeneration">
                            <param name="plan" value="$plan" />
                            <output>running</output>
                        </operation>
                </if>
                <else>
                    <assign name="running" value="FALSE" />
                </else>
        </conditional>
</while>
```

The Angerona framework contains the implementation of the BDI$^+$ agents we developed in Section 3.5, default operators for these, the base versions a presented in 3.5.4, and sophisticated operators. These operators include the full know-how approach we introduced in Section 3.5.2 and worked out by means of answer set programming in [157], the full motivation approach we introduced in Section 3.5.1 and worked out in [159].

The *secrecy BDI$^+$ agents* and the *secrecy ASP BDI$^+$ agents* that we introduced in Chapter 5 are also implemented in the Angerona framework. The secrecy BDI$^+$ agent model is illustrated in Figure 6.1.3. It refines the belief component of the BDI$^+$ agents that now consists of the agent's view on the world, views on the world-views of other agents, and a representation of its secrets. The change operator then changes all components of the beliefs. This agent model can then be instantiated by use of different knowledge representation formalisms. This far, we used propositional logic, ordinal conditional functions, and answer set programming to instantiate it and run simulations.

In this scenario we make use of several of Angerona's particular features. We build on the BDI sub-framework provided by Angerona and refine the composition of the epistemic state. The views on other agents are epistemic components such that for each agent a different knowledge representation formalism and different belief operators can be used. Changes to the views on other agents are performed by operators from the epistemic plug-in used for the respective view. We also implemented operators that change the secrets of an agent

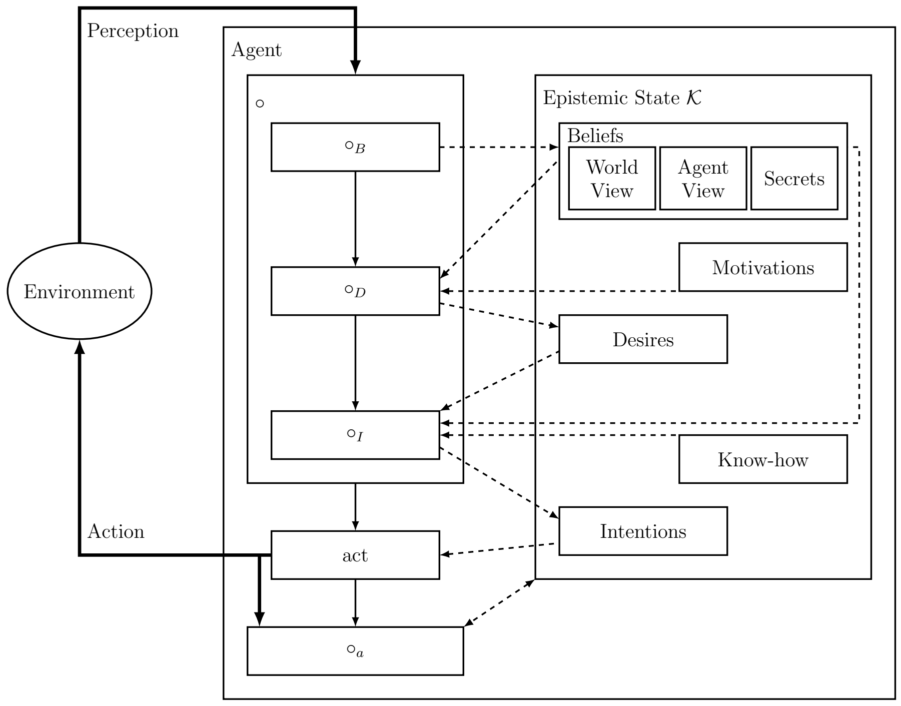
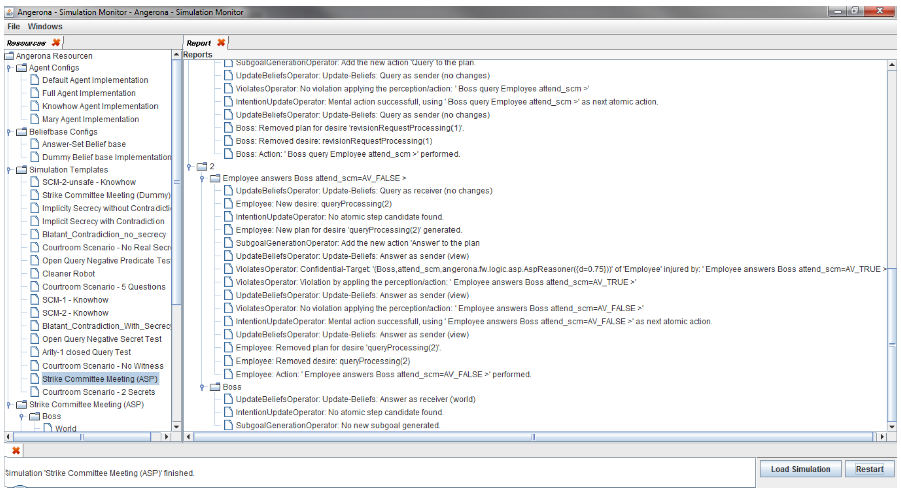


Figure 6.1.3: BDI$^+$ secrecy agent model and agent cycle

in the light of new information. We used different knowledge representation plug-ins for the agent instantiations. For the ASP instance we build on the ASP know-how [157] implementation of the intention-update operator in Angerona and extended it to take the secrets into consideration. The report system and the GUI serve well to inspect the evaluation process of actions with respect to secrecy since the internal, change operations can be observed. This is, for instance, very useful to trace the temporarily considered changes to views on other agents in the process of the evaluation of actions and their effect on secrecy. Figure 6.2.2 shows the resource view and an example ASP epistemic component view from the *strike committee meeting* secrecy scenario.

6.2 MULTIAGENT FRAMEWORK

The multiagent framework of Angerona is what is commonly referred to as the *middleware* of agent programming frameworks. It organizes and starts the execution of the individual agents and the environment and implements the interaction between these. The execution order of the agents, the *multiagent cycle*, is flexible. The default is the sequential execution of agents such that each agent gets the percepts from the previous multiagent cycle, and not those created by the execution of agents in the same cycle. This way the order of the execution of agents does not matter.


Angerona - Simulation Monitor - Angerona - Simulation Monitor
File
Windows
Resources
Report
Load Simulation
Restart

```
Angerona Resourcen
  Agent Configs
    Default Agent Implementation
    Full Agent Implementation
    Knowhow Agent Implementation
    Mary Agent Implementation
  Beliefbase Configs
    Answer-Set Belief base
    Dummy Belief base Implementation
  Simulation Templates
    SCM-2-unsafe - Knowhow
    Strike Committee Meeting (Dummy)
    Implicity Secrecy without Contradicti
    Implicit Secrecy with Contradiction
    Blatant_Contradiction_no_secrecy
    Courtroom Scenario - No Real Secr
    Open Query Negative Predicate Tes
    Cleaner Robot
    Courtroom Scenario - 5 Questions
    SCM-1 - Knowhow
    SCM-2 - Knowhow
    Blatant_Contradiction_With_Secrec
    Open Query Negative Secret Test
    Arity-1 closed Query Test
    Courtroom Scenario - No Witness
    Strike Committee Meeting (ASP)
    Courtroom Scenario - 2 Secrets
  Strike Committee Meeting (ASP)
    Boss
      World
```

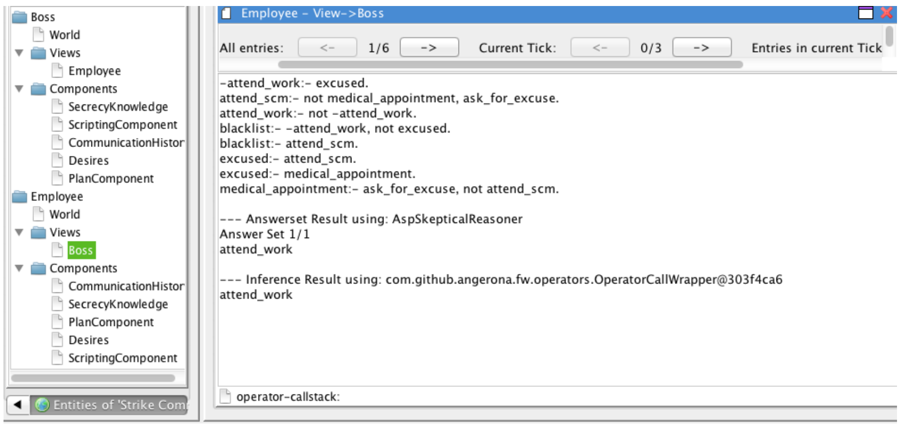
```
Reports
    SubgoalGenerationOperator: Add the new action 'Query' to the plan.
    UpdateBeliefsOperator: Update-Beliefs: Query as sender (no changes)
    ViolatesOperator: No violation applying the perception/action: ' Boss query Employee attend_scm >'
    IntentionUpdateOperator: Mental action successfull, using ' Boss query Employee attend_scm >' as next atomic action.
    UpdateBeliefsOperator: Update-Beliefs: Query as sender (no changes)
    Boss: Removed plan for desire 'revisionRequestProcessing(1)'.
    Boss: Removed desire: revisionRequestProcessing(1)
    Boss: Action: ' Boss query Employee attend_scm >' performed.
  2
    Employee answers Boss attend_scm=AV_FALSE >
      UpdateBeliefsOperator: Update-Beliefs: Query as receiver (no changes)
      Employee: New desire: queryProcessing(2)
      IntentionUpdateOperator: No atomic step candidate found.
      Employee: New plan for desire 'queryProcessing(2)' generated.
      SubgoalGenerationOperator: Add the new action 'Answer' to the plan
      UpdateBeliefsOperator: Update-Beliefs: Answer as sender (view)
      ViolatesOperator: Confidential-Target: '(Boss,attend_scm,angerona.fw.logic.asp.AspReasoner({d=0.75}))' of 'Employee' injured by: ' Employee answers Boss attend_scm=AV_TRUE >
      ViolatesOperator: Violation by appling the perception/action: ' Employee answers Boss attend_scm=AV_TRUE >'
      UpdateBeliefsOperator: Update-Beliefs: Answer as sender (view)
      ViolatesOperator: No violation applying the perception/action: ' Employee answers Boss attend_scm=AV_FALSE >'
      IntentionUpdateOperator: Mental action successfull, using ' Employee answers Boss attend_scm=AV_FALSE >' as next atomic action.
      UpdateBeliefsOperator: Update-Beliefs: Answer as sender (view)
      Employee: Removed plan for desire 'queryProcessing(2)'.
      Employee: Removed desire: queryProcessing(2)
      Employee: Action: ' Employee answers Boss attend_scm=AV_FALSE >' performed.
    Boss
      UpdateBeliefsOperator: Update-Beliefs: Answer as receiver (world)
      IntentionUpdateOperator: No atomic step candidate found.
      SubgoalGenerationOperator: No new subgoal generated.
```

```
Simulation 'Strike Committee Meeting (ASP)' finished.
```

Figure 6.2.1: Angerona GUI - Overview

The environment in Angerona is formed by the set of agents in the system and the environment behavior. The environment behavior might range from a communication infrastructure that delegates the speech acts between agents, similar to, e. g., [23], to a simulator for physical environments [242, 243]. It is implemented in form of an environment plug-in which allows to use external environment simulators, or to develop new ones. The interrelation of the environment classes is shown in Figure 6.1.2. The environment behavior for communicating agents, as introduced in Section 3.1, is implemented as the default behavior in Angerona. The actions of agents are speech acts and are transmitted to the receiver agents as their percept. Since different agents might use different knowledge representation formalisms in Angerona a common logical language has to be determined for which each agent has an appropriate interpretation function. As a language that is appropriate for agents that use such different formalisms as *ASP* and OCF we chose *nested logic programs* [169] as common language for the agents. It supports both, propositional logic and its connectives as well as conditional or rule like connectives, and default negation. However, this is only the default implementation, any other language might be used as common communication language.

Angerona also features a versatile graphical user interface (GUI). It is based on a docking panes approach which allows to display various aspects in different panes, which tile the window. The tiling can be changed individually. Panes can be grouped by means of tabs or be detached from the main window to form new windows that might be moved to a secondary screen. The plug-in architecture of Angerona allows for UI plug-ins, which allow the development of plug-ins for the specific visualization of components stemming from plug-ins such as the representation of epistemic components specific



```
-attend_work:- excused.
attend_scm:- not medical_appointment, ask_for_excuse.
attend_work:- not -attend_work.
blacklist:- -attend_work, not excused.
blacklist:- attend_scm.
excused:- attend_scm.
excused:- medical_appointment.
medical_appointment:- ask_for_excuse, not attend_scm.

--- Answerset Result using: AspSkepticalReasoner
Answer Set 1/1
attend_work

--- Inference Result using: com.github.angerona.fw.operators.OperatorCallWrapper@303f4ca6
attend_work
```

Figure 6.2.2: Secrecy Scenario Ressources (left), *ASP* Belief Base UI Component (right)

to the used formalism, potentially with alternative views such as text-based and graph-based perspectives. For example for ASP we implemented a representation based on explanation and extended dependency graphs, as presented in [4].

Another important feature of the user interface is the *report system* used in Angerona. The *report* defines an interface to post new *report entries* and to query the existing ones. A report entry consists of the identifier of its poster, the tick (number of multiagent cycle) and the realtime (system time) in which it was posted, a call-stack and an optional attachment in form of an epistemic component. Poster of report entries can be the agent, one of its operators or one of its epistemic components. The queries to the report then allow to construct a timeline of posts with filters based on the poster, the type of attachment and the call-stack; for instance to inspect the changes of an agent's beliefs during runtime. The report system is extensively used by the GUI to allow for the inspection on the level of an agent cycle and of a multiagent cycle. Every pane that displays the content of an epistemic component uses the report system to provide a timeline for its displayed component.

Figure 6.2.1 shows the GUI after a variant of the *strike committee meeting* scenario simulation has been selected and run. The window is tiled by three docking panes, the resource pane to the left, the workspace pane to the right and the status pane at the bottom. The resources are displayed in a tree-view and are given by agent configuration files, epistemic component configuration files, simulations templates and resources of a loaded simulation. The resources of a simulation are typically given by its agents and their components. The workspace pane has its default tab, which views the report for the current simulation. Resources of the resource pane are opened and displayed as an additional tab of the workspace pane by double-clicking on them. The status pane displays the current status of ANGERONA and holds buttons to load and run a simulation.

Figure 6.2.2 shows how an agent component can be selected and inspected in the *strike committee meeting* scenario by example of the view of the employee (*Emma*) on the world view of the boss (*Beatriz*), which is an extended logic program. The logic program of the epistemic component is shown as well as its answer sets and the corresponding belief set that is produced by the selected belief operator. The controls at the top allow for the navigation through the timeline of the epistemic component given by the current report. It is shown how many entries for the epistemic component exist in the entire report, how many ticks the report covers, and how many entries for the epistemic component exist in the currently selected tick. The controls allow the navigation on the basis of these three parameters. The changes to the epistemic component with respect to the previous report entry for the epistemic component are shown by highlighting new parts in green and missing parts in red. These controls and the form of representation allows to not only inspect the epistemic component but also to track its evolution throughout the simulation.

6.3 RELATED WORK

A multitude of multiagent programming frameworks has been proposed. Most of these are rather theoretical studies and relatively few, but still a lot, have been actually implemented and are available. A good overview of the most prominent available frameworks is given in [51]. An even more extensive list of such frameworks can be found in [1]. These frameworks haven been build with very different goals in mind and by use of very different means. In the following we survey those coming closest to the ANGERONA frameworks main features. These are:

1. to provide a means to build agents capable of using (different) non-monotonic knowledge representation and reasoning techniques,
2. to allow for flexible agent cycles,
3. to allow multiple levels of use and customization of the framework,
4. to feature the development of secrecy aware and secrecy preserving agents.

We have shown in this thesis that and how we realized these goals. In the following we discuss the existing frameworks being closest to satisfying some of these goals.

With respect to non-monotonic knowledge representation, to our knowledge, the only formalism that has been used in practical multiagent systems is ASP. But most implemented works on ASP agents treat only the planning problem independently of the rest of the agent, e. g., in the APLAgent Manager [16] or the DLV^K system [86]. In the literature several proposals for the design of an agent entirely based on ASP have been made, e. g., [164, 182]. However, for these no implemented systems or documentation on how to implement such a system are available. To the best of our knowledge, there are only two complete and available multiagent programming frameworks that facilitate the use of ASP for knowledge representation, namely *Jazzyk* [186] and *GOAL* [130, 148]. Both of them also feature a modular approach with respect to knowledge representation.

A *Jazzyk* agent consists of several knowledge bases, which are realized by knowledge representation modules and an agent program. The agent program consists of a set of rules of the form "when *Query*, then *Update*". The knowledge representation modules implement the *Query* and *Update* methods. The semantics of the agent programs is based on the *Behavioural State Machines* approach developed for *Jazzyk* on the basis of abstract state machines [53]. The knowledge representation modules allow to use different knowledge representation formalisms, and an implemented ASP module is available. With respect to belief operators it implements credulous and skeptical ASP querying. But with respect to belief change it only supports a pure addition of new formulae to the knowledge base and no actual belief change. The only other existing available KR module is based on the *Ruby* programming language which cannot be considered as a logic based knowledge representation formalism.

The *GOAL* Framework [130, 148] also features the specification of modules for knowledge representation and allows, in principle, for the use of different knowledge representation formalisms. There are general interfaces for knowledge representation, but we could not find implementations or examples for any formalism other than *Prolog*. The agent programs used in *GOAL* feature a clear syntax and semantics, but are rather inflexible with respect to the use of different agent cycles and architectures. The structure is fixed, goals are defined explicitly and are blindly committed to.

There are no other multiagent frameworks that consider the development of secrecy aware and secrecy preserving agents with explicitly represented secrets and views on other agents, as considered in ANGERONA. The closest implemented frameworks on the consideration of privacy in multiagent systems consider rather specific problems in distributed problem solving. The *DAREC*^{2} system [171] considers the problem of a group of agents that have to collaboratively compute consistent answers to a query and protect their private information at the same time. Confidentiality is expressed by means of the specification of *private* and *askable* literals which are used in a distributed abduction algorithm based on state passing. In [238] quantitative privacy loss is considered in distributed constraint satisfaction problems.

6.4 CONCLUSION

In this chapter we presented the ANGERONA framework for the implementation of knowledge-based agents. It bases on the epistemic agent concept we introduced in Chapter 3. It realizes the general concept by means of a flexible plug-in architecture and an *XML* based scripting language called ANGERONA *Script Markup Language* (*ASML*). Agents comprise an epistemic state that can be composed of epistemic components and a functional component. The epistemic components are realized by the epistemic plug-in and the operators of the functional component by the operator plug-in. The agent cycle in ANGERONA is specified by means of the *ASML* script in combination with the operator interface.

The distinction between operator types and their implementation in combination with predefined agent cycles, e. g., the BDI cycle, allows for multiple levels of use and customization. By means of the operator type concept the BDI^+ agent model we introduced in Chapter 3 is realized as a sub-framework of ANGERONA. Furthermore the secrecy BDI^+ model we introduced in Chapter 5 is realized this way.

The epistemic plug-ins use the TWEETY *library* for knowledge representation [227]. This way all knowledge representation formalisms of TWEETY can be used for ANGERONA agents. We extended the TWEETY library by our own plug-ins and in particular the ASP plug-in. The ANGERONA framework ASP Plug-in supports the use of various ASP solvers and different extensions thereof, such as *DLV-complex*. On the basis of the latter a planning component on the basis of know-how [157, 229] is implemented. ANGERONA implements the change operator concept we developed in Chapter 4. Further the selective revision operator by deductive argumentation and the belief base change operator for ASP we presented in the same chapter are implemented. By use of the ASP Plug-in and the ASP belief base change operator, the secrecy ASP BDI^+ instantiation we developed in Chapter 5 is implemented and the simulation of the *strike committee meeting* scenario realized.

Moreover, many other approaches are implemented in ANGERONA. For the ANGERONA framework, plug-ins for ASP, OCF and propositional logic can be used in several available complete simulations. ANGERONA is open source and available at *github*[3].

3 `https://github.com/Angerona`

7 CONCLUSION

In the following we first summarize the content of this thesis, then recapitulate our main contributions, and finally we discuss some further and future work.

7.1 SUMMARY

In this thesis we contributed to and combined the fields of research in *multiagent systems*, *non-monotonic reasoning*, *belief revision* and *secrecy*. In a nutshell, we defined a general model of epistemic agents in Chapter 3. Then we elaborated on the integration of non-monotonic reasoning, belief revision and secrecy on the basis of the general epistemic agent model in Chapters 3, 4, and 5. And then, in Chapter 6 we presented a multiagent framework that implements the general concept of epistemic agents, and that implements all concepts presented in this thesis. In the following we summarize these chapters in more detail.

In Chapter 3 we developed a general model of epistemic agents. This model unifies diverse agent architectures and allows for the specification and realization of agent models. We focused on communicating agents and agents that make use of logical knowledge representation formalisms. For these we defined a set of speech acts by means of which our agents interact. Then, we introduced epistemic components and belief operators for the epistemic state of the agent. This includes in particular the formalization of belief operators for (non-monotonic) reasoning, ordered families of these and properties for those. Further, we defined concrete instances on the basis of answer set programming and propositional logic. We formalized the decomposition of the agent's epistemic state and functional component of an agent into epistemic components and sub-functions for the realization of compound agent models. We developed the BDI^+ agent model as an instance of the general compound model. This agent model is based on the well known BDI agent model and extends it by a *motivation* and a *know-how* component that we developed. We further concretized this agent model by defined a basic instance of the BDI^+ model.

In Chapter 4 we first developed the general structure of belief change operations for epistemic agents. In this approach, the new information is first interpreted by an *interpretation function* that returns a set of sentences. This set is then evaluated in the light of the information represented in the considered epistemic component by means of a *selection function* that decides which part of the input shall be accepted. This resulting set is then used by a prioritized revision operator to change the epistemic component accordingly.

The operation that consists of the last two steps, i. e., selection and revision, is called a *selective revision operator* in the literature. We built on the selective revision framework and extended it to multiple revision operations that take sets of sentences as input. Further, we presented the first construction of non-trivial selection functions, and that by use of non-monotonic deductive argumentation. We formally proved that our selection functions satisfy a set of desirable properties and lead to a proper selective revision operator.

With respect to the prioritized revision operator, we studied the applicability of the base revision approach to non-monotonic belief bases in general and ASP in particular. We developed a set of desirable postulates for a *multiple ASP base revision operator* on the basis of the postulates for propositional base revision and ASP specific postulates. Then we developed a constructive approach by defining a new *screened consolidation operator*, which we also characterized by a set of postulates. For this operator we defined a construction for it on the basis of *screened remainder sets* and a *global selection function*. The latter leads to a general operator that can be iterated. We proved a representation theorem for the resulting consolidation operator. Moreover, we showed that the ASP specific postulates of *Weak Disjointness* and *Weak Parallelism* are implied by the property of *Topic Independence* from the belief base revision literature. We also defined the property of *monotony* for selection functions on remainder sets and showed that monotone selection functions lead to consolidation operators that satisfy *Topic Independence*.

In Chapter 5 we developed a novel, subjective, *agent-based notion of secrecy*. For this we used and build on our epistemic agent framework. We developed a notion of secrets and of secrecy preservation of an agent. And we introduced a secrecy agent model and elaborated various aspects of agent models with respect to secrecy. Hereby we developed *secrecy aware* agents that are capable to take secrecy into consideration and that are aware of the effect of their actions with respect to secrecy. Secrecy aware agents are *secrecy preserving* whenever it is feasible. If it is possible form them, in their settings these can be shown to be *secrecy preserving*.

We defined properties of change operators with respect to secrecy for the change of the world-view of an agent, the agent-view of an agent, and the change of the secrets of an agent. Further we defined a secrets change operator that satisfies the minimality of change property we demand for it.

We defined properties of settings, as well as necessary and sufficient properties of agents to guarantee the preservation of secrecy. As part of this, we developed a game-theoretic formulation of the consideration of future actions of the defending and the attacking agent. In this formulation the defending agent has to ensure that the attacking agent does not have a winning strategy to get to know one of its secrets. We showed that this requirement can be ensured without the consideration of future actions in some settings. For a query-answer setting we showed that the attacking agent has a winning strategy if and only if it knows that the disjunction of the potentially secret formulas of the defender holds.

Moreover, we considered the evaluation of options for action with respect to secrecy in detail. We defined a set of principles for the classification of actions with respect to secrecy that use the available information as well as possible to determine a fine grained and justified classification of the actions. This classification is needed for agents that do not only distinguish if secrecy is violated or not, but can distinguish between degrees of violation, such as secrecy aware agents that might have to violate secrecy. In addition to the principle based characterization we devised an algorithm for the classification of a set of actions that satisfies all of our declared principles.

Further, we built on our BDI^+ agent model to construct complete secrecy aware agents. We showed that and how *motives* and *know-how* have to be considered with respect to secrecy. We elaborated the complete instantiation of the basic BDI^+ model by use of ASP. For this, we developed ASP representations for meta-information and meta-inferences. We showed that and how the *strike committee meeting* example from the introduction can realized by use of the *secrecy ASP* BDI^+ *agents*. Finally, we discussed the relation of our notion of secrecy to other important works on secrecy in multiagent systems and in information systems detail.

In Chapter 6 we presented the multiagent programming framework ANGERONA. We described its general architecture and how it realizes the epistemic agent model presented in Chapter 3 and the instances based on the results of Chapters 4 and 5.

7.2 MAIN CONTRIBUTIONS

In the introduction we motivated and described several challenges and the contributions that we make with respect to these. Picking up the motivation and the structure of the introduction we recapitulate main contributions of this thesis in the following.

The work in this thesis is motivated by the *strike committee meeting* example and the main research question:

> *How can secrecy be formalized and realized for practical agents with incomplete information in a dynamic environment?*

There is a lot of thorough work on the topic of secrecy for multiagent systems. The definitions of secrecy are very strict and are based on strong assumptions. Further these works only consider multiagent systems abstractly. There is also a lot of thorough work on the topic of secrecy in information systems. It considers mostly query answering in client-server architectures. There is, however, no work on a general notion of secrecy for practical, knowledge-based agents.

General and flexible concept for developing epistemic agents

Hence we started from scratch to approach the topic of secrecy from the perspective of a bounded rational agent with incomplete information. We argued, that we need a general model of agents that is strongly based on logical knowledge representation and reasoning, and that can be instantiated and used to develop practical agents. The agent needs to handle incomplete information, which calls for non-monotonic reasoning formalisms, and it needs to change its information adequately, which calls for belief change operators. We called the agents satisfying these properties epistemic agents and we developed a general epistemic agent model. In this model we focused on communicating agents and stressed that these have to have the ability to reflect the anticipated changes of its actions in its epistemic state to perform well. To this end, we introduced of a change operator for actions $\circ^{a}$ as a part of the agent cycle.

Non-monotonic reasoning formalisms for epistemic agents

We elaborated on (non-monotonic) belief operators and defined a general notion of belief operator families whose operators can be ordered based on how credulous the inference behavior is. Further, we defined desirable properties for belief operators and ordered families of these. The assumptions we made are general enough to capture a big variety of knowledge representation formalisms. In particular we included formalisms for which effective solvers exist, such as answer set programming.

Belief change operators and non-monotonic reasoning

We contributed to the field of the combination of belief revision and non-monotonic reasoning by developing change operators that use non-monotonic reasoning in the belief revision process, and by developing belief change operations for non-monotonic formalisms. We defined the first complex instantiation of selection functions in the selective revision framework. To this end we used a deductive argumentation framework. For the selection functions and for the resulting selective revision operators we formally proved the satisfaction of a set of desirable properties

We developed a belief base approach for answer set programming. For this we considered and adapted postulates from propositional base revision and specific postulates for change operations of answer set programs. Moreover, we defined a new construction based on a screened consolidation operator and showed a representation theorem for it. We could also show that an independence property for consolidation operators implies, for our construction, the satisfaction of ASP specific independence properties.

Agent-based secrecy and secrecy aware agents

On the basis of the epistemic agent framework we developed a theoretical definition of *secrecy preserving agents* as a special case of more practical *secrecy aware agents*. We also consider the violation of secrecy by an agent and develop a model that is capable to handle such violations. In particular we developed principles and an algorithm to classify actions with respect to secrecy. We defined a game theoretic formulation for our notion of secrecy and showed for a query-answer setting the attacking agent has a winning strategy if and only if it knows that the disjunction of the potentially secret formulas of the defender holds. Hereby we showed that this important result from the work on secrecy for databases [213] is a special case of our formulation of winning strategies.

We elaborated on belief change, the consideration of future actions and action selection with respect to secrecy independently of a concrete agent model. Additionally we built secrecy aware ASP BDI^+ agents that model patterns of meta-inferences of an $\mathcal{A}$. We related our work in particular to the formulations of secrecy in the runs-and-systems framework of Halpern and O'Neil and to the works of Biskup et al. on controlled interaction execution. And we discussed further related work. By our work, we showed that the consideration of secrecy in practical agents is complex and demands for a thorough consideration in the agent's functioning on all levels.

Epistemic agent programming framework ANGERONA

We developed the ANGERONA[1] multiagent programming framework on the basis of our epistemic agent model. It is general framework for the implementation of autonomous agents with different agent architectures and different knowledge representation formalisms. It contains implementations of virtually all approaches presented in this thesis, and several others, e. g., [75, 76, 152, 131, 244, 4, 229, 157, 159]. Further it integrates the TWEETY[2] library for knowledge representation which contains implementations of many knowledge representation formalisms.

7.3 FURTHER AND FUTURE WORK

In the presentation of our work in this thesis we, naturally, did not consider all aspects in full detail, but focused on the aspects that are

1 `https://github.com/Angerona`
2 `http://sourceforge.net/projects/tweety/`

most relevant for the topic of this thesis. For instance, we defined a comprehensive, but basic instance of the BDI$^+$ model. The next step lies in consideration of the BDI$^+$ model instantiated with the full approaches of know-how and motivation, which we published in [157] and [159] respectively. In particular, with respect to secrecy this allows for the modeling and handling of more complex scenarios. The know-how component realizes a hierarchical planning component. The thorough consideration of secrecy and planning is an interesting topic for future work in itself. We already laid the foundations for this by the integration of the know-how approach in our BDI$^+$ agent model. Further considerations in this direction can be found in our publication [162] and the diploma thesis [137].

Another interesting aspect for future work is the consideration of the communication of more than two agents and the modeling of information flow and the implications with respect to secrecy. We already noted in Section 5.7.3 (Page 195) that a first step would be to integrate a communication graph of all agents as the one used in [224].

The change operator that we introduce for our agents enables the agents to decide if and to which extent they want to accept information from other agents. This can be augmented by models of the credibility of the information, as we considered in [223]. Moreover the communication models can be combined with the credibility of information, which we considered in [158]. Future work lies in the inclusion of these approaches in the consideration of secrecy aware agents.

With respect to our work on belief change operators future work lies in the extension of the results to other formalisms, for instance the formulation of argumentation based selection functions for answer set programming. First results on this can be found in the master's thesis [131]. It would also be interesting future work to extend the base revision approach to answer set programming we presented to non-prioritized change operations and merge operators.

The ANGERONA framework can be used to simulate more complex scenarios and can be extended by the implementation of new approaches. Moreover, different approaches, agent models and knowledge representation formalisms can be directly compared in one multiagent system. First steps in this direction can be found in our report [78]. ANGERONA agents that combine planning by means of our know-how approach [157] with secrecy have been developed in the diploma thesis [137]. In the bachelor's thesis [52] ANGERONA has been used to model secrecy in an e-market scenario as a possible application area of our secrecy approach. The implementation of agents in a physical environment in ANGERONA has been started in the bachelor's thesis [19]. The agents in that thesis are based on the BDI$^+$ model with the full motivation component as defined in our work in [159].

A APPENDIX

A.1 PROOFS

A.1.1 *Proofs for Chapter 3*

Proposition 3.6.4. (On Page 64) The pair $(\Xi^{\mathsf{prop}}, \preceq^{\mathsf{prop}}_{\mathsf{bel}})$ is a belief operator family that satisfies *Consistency*, *Suprality*, *Right-Weakening* and *Credulity*$_{\preceq_{\mathsf{bel}}}$.

Proof.

Consistency:
Let the epistemic component $E \subseteq \mathcal{L}^{\mathsf{prop}}_{At}$ be consistent, i. e., $\mathsf{Mod}(()E) \neq \emptyset$ and $\mathsf{Bel}_p \in \Xi^{\mathsf{prop}}$. For any $\phi \in \mathsf{Bel}_p(E)$ it holds by definition of Bel_p that

$$\frac{|\mathsf{Mod}(\phi) \cap \mathsf{Mod}(E)|}{|\mathsf{Mod}(E)|} > 0.5.$$

For the models of the negated formula $\neg\phi$ it holds that $\mathsf{Mod}(\neg\phi) \cap \mathsf{Mod}(E) = \mathsf{Mod}(E) \setminus \mathsf{Mod}(\phi)$. Consequently it holds that

$$\frac{|\mathsf{Mod}(\neg\phi) \cap \mathsf{Mod}(E)|}{|\mathsf{Mod}(E)|} \leqslant 0.5$$

and therefore by definition of Bel_p it holds that $\neg\phi \notin \mathsf{Bel}_p(E)$. Hence we showed that all $\mathsf{Bel}_p \in \Xi^{\mathsf{prop}}$ satisfy *Consistency*.

Credulity:
Let $E \subseteq \mathcal{L}^{\mathsf{prop}}_{At}$ be an epistemic component that is consistent, i. e., $\mathsf{Mod}(E) \neq \emptyset$, and $\mathsf{Bel}_p \in \Xi^{\mathsf{prop}}$ and $\mathsf{Bel}_{p'} \in \Xi^{\mathsf{prop}}$ such that $\mathsf{Bel}_p \preceq^{\mathsf{prop}}_{\mathsf{bel}} \mathsf{Bel}_{p'}$. By definition of $\preceq^{\mathsf{prop}}_{\mathsf{bel}}$ it holds that $p \leqslant p'$. By definition of Bel_p for each $\phi \in \mathsf{Bel}_p(E)$ it holds that

$$\frac{|\mathsf{Mod}(\phi) \cap \mathsf{Mod}(E)|}{|\mathsf{Mod}(E)|} \leqslant p \leqslant p'$$

such that $\phi \in \mathsf{Bel}_{p'}(E)$ and consequently $\mathsf{Bel}_p(E) \subseteq \mathsf{Bel}_{p'}(E)$, as was to be shown.

Suprality:
Let the epistemic component $E \subseteq \mathcal{L}^{\mathsf{prop}}_{At}$ be consistent, i. e., $\mathsf{Mod}(E) \neq \emptyset$. By definition of propositional consequence it holds that

$$Cn^{\mathsf{prop}}(E) = \{\phi \in \mathcal{L}^{\mathsf{prop}}_{At} \mid \mathsf{Mod}(E) \subseteq \mathsf{Mod}(\phi)\}.$$

This implies that for all $\phi \in Cn^{\mathsf{prop}}(E)$ it holds that

$$\frac{|\mathsf{Mod}(\phi) \cap \mathsf{Mod}(E)|}{|\mathsf{Mod}(E)|} = 1$$

such that $Cn^{\mathsf{prop}}(E) = \mathsf{Bel}_p(E)$. It follows from the definition of $\preceq^{\mathsf{prop}}_{\mathsf{bel}}$ and its satisfaction of *credulity* that for all $\mathsf{Bel}_p \in \Xi^{\mathsf{prop}}$ it holds that $\mathsf{Bel}_p(E) \subseteq Cn^{\mathsf{prop}}(E)$, which was to be shown.

Right-Weakening:
Let the epistemic component $E \subseteq \mathcal{L}^{\mathsf{prop}}_{At}$ be consistent, i. e., $\mathsf{Mod}(E) \neq \emptyset$, and $\phi \in \mathsf{Bel}_p(E)$. It holds by definition of Cn^{prop} that $Cn^{\mathsf{prop}}(E) = \{\phi \in \mathcal{L}^{\mathsf{prop}}_{At} \mid \mathsf{Mod}(E) \subseteq \mathsf{Mod}(\phi)\}$. Hence, for all $\psi \in Cn^{\mathsf{prop}}(\phi)$ it holds by transitivity of $\subseteq$ that $\mathsf{Mod}(\phi) \subseteq \mathsf{Mod}(\psi)$ and therefore

$$p \leqslant \frac{|\mathsf{Mod}(\phi) \cap \mathsf{Mod}(E)|}{|\mathsf{Mod}(E)|} \leqslant \frac{|\mathsf{Mod}(\psi) \cap \mathsf{Mod}(E)|}{|\mathsf{Mod}(E)|}.$$

Consequently $\psi \in \mathsf{Bel}_p(E)$, which proves the claim. □

Proposition 3.6.6. (On Page 66) In our ASP instance the extension family function *AS* satisfies the postulates *Suprality*, *Relative-Left-Absorption*, *Closure*$_{Cn^{\mathsf{asp}}}$, *Consistency* and *Separation*.

Proof.

Suprality:
Let P be an extended logic program. We show that for all $P \subseteq \mathcal{L}^{\mathsf{asp}}_{At}$ holds that $Cn^{\mathsf{asp}}(P) \subseteq \mathsf{Bel}(P)$.

For each answer set $e \in AS(P)$ it holds that e is the minimal model of P^e. By definition of the reduct and the definition of P^{strict} it holds that $P^{\mathsf{strict}} \subseteq P^e$. From the monotony of strict programs it follows for the minimal model of P^{strict}, denoted e_s, that $e_s \subseteq e$.

Relative-Left-Absorption:
As already mentioned in Example 3.6.3 for any $BS \subseteq \mathcal{L}^{\mathsf{Form}}_{BS} = \mathsf{Lit}$ it holds that:

$$Cn^{\mathsf{prop}}(BS) \cap \mathcal{L}^{\mathsf{Form}}_{BS} = BS.$$

Since $\mathcal{L}^{\mathsf{asp}}_{BS} = \mathsf{Lit}$ this proves that *Relative-Left-Absorption* is satisfied.

Closure$_{Cn^{\mathsf{asp}}}$*:*
For each answer set $e \in AS(P)$ it holds that e is the minimal model of P^e. This means that for each literal $l \in e$ there is a rule r with $\mathsf{head}(r) = l$ and its body $\mathsf{body}(r)$ is satisfied by e. We consider the program P extended by facts for each literal in the minimal model of the program:

$$P \cup^{\mathsf{asp}} e = P \cup \{l. \mid l \in e\}$$

It holds that all new rules, i.e., $\{l. \mid l \in e\}$, are, trivially, satisfied. For each $l. \in \{l. \mid l \in e\}$ there exists a rule $r \in P$ such that $\mathsf{head}(r) = l$ and $\mathsf{body}(r) \subseteq e$. Hence the minimal model of $P \cup \{l. \mid l \in e\}$ is e such that $Cn^{\mathsf{asp}}(P \cup^{ASP} e) = e$.

Consistency
A set of literals is consistent if it does not contain any complementary literals L and $\neg L$. By definition (see Section 2.4) an answer set is a state. A state is defined as a consistent set of literals. Hence any answer set is consistent.

Separation
Suppose that for a program P there are two answer sets $S, S' \in AS(P)$ such that $S \subset S'$. Then by construction of a reduct it holds that $P^{S'} \subseteq P^{S}$. By definition of answer sets it holds that S is the minimal model of P^{S}, and that S' is the minimal model of $P^{S'}$. Since reducts are strict programs they are monotonic and it follows that $S' \subseteq S$, in contradiction to the assumption. $\square$

Proposition 3.6.7. (On Page 66) For the *ASP belief operator family* the following results hold:

1. $\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}$ satisfies *Suprality*$_{\mathsf{Bel}}$, *Consistency*$_{\mathsf{Bel}}$ and *Relative-Left-Absorption*$_{\mathsf{Bel}}$.
2. $\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}$ satisfies *Suprality*$_{\mathsf{Bel}}$, *Relative-Left-Absorption*$_{\mathsf{Bel}}$.
3. $\preceq_{\mathsf{bel}}$ satisfies *Credulity*$_{\preceq_{\mathsf{bel}}}$.

Proof. Let P be an extended logic program.

1. *Suprality*$_{\mathsf{Bel}}$:
   Since AS satisfies *Suprality*$_{\mathsf{Bel}}$ it holds that $Cn^{\mathsf{asp}}(P) \subseteq S$ for all $S \in AS(P)$. It follows directly that $Cn^{\mathsf{asp}}(P) \subseteq \cap AS(P) = \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}$.

   Consistency$_{\mathsf{Bel}}$:
   We have shown in Proposition 3.6.6 that the extension family function AS satisfies *Consistency*$_{\mathsf{ext}}$. This means that for all consistent programs P, all answer sets $AS(P)$ do not contain any complementary literals L and $\neg L$. It follows directly that $\cap AS(P)$ does not contain any complementary literals L and $\neg L$. Since $\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}} = \cap AS(P)$ *Consistency*$_{\mathsf{Bel}}$ is satisfied by $\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}$.

   Relative-Left-Absorption$_{\mathsf{Bel}}$:
   As already mentioned in Example 3.6.3 for any beliefset $BS \subseteq \mathcal{L}^{\mathsf{Form}}_{BS} = \mathsf{Lit}$ it holds that:

   $$Cn^{\mathsf{prop}}(BS) \cap \mathcal{L}^{\mathsf{Form}}_{BS} = BS.$$

   Since $\mathcal{L}^{\mathsf{asp}}_{BS} = \mathsf{Lit}$ it follows that *Relative-Left-Absorption*$_{\mathsf{Bel}}$ is satisfied.

2. *Suprality*$_{\mathsf{Bel}}$*:*
   Since *AS* satisfies *Suprality*$_{\mathsf{ext}}$ it holds that $Cn^{\mathsf{asp}}(P) \subseteq S$ for all $S \in AS(P)$. It follows directly that $Cn^{\mathsf{asp}}(P) \subseteq \cup AS(P) \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}$.

   Relative-Left-Absorption$_{\mathsf{Bel}}$*:*
   As already mentioned in Example 3.6.3 for any $BS \subseteq \mathcal{L}^{\mathsf{Form}}_{BS} = \mathsf{Lit}$ it holds that:

   $$Cn^{\mathsf{prop}}(BS) \cap \mathcal{L}^{\mathsf{Form}}_{BS} = BS.$$

   Since $\mathcal{L}^{\mathsf{asp}}_{BS} = \mathsf{Lit}$ it follows that *Relative-Left-Absorption*$_{\mathsf{Bel}}$ is satisfied.

3. *Credulity*$_{\mathsf{Bel}}$*:*
   We show for each pair of the relation $\preceq^{asp,b}_{\mathsf{bel}}$ that the necessary subset relation holds. For $(\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}})$ it holds for any program P that

   $$\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}(P) = \cap AS(P) \subseteq \cup AS(P) = \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}(P).$$

   For $(\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}})$ and $(\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}, \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}})$ it clearly holds for all programs P that

   $$\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}(P) = \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}(P) \text{ and } \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}(P) = \mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}(P).$$

□

A.1.2 *Proofs for Chapter 4*

Proposition 4.2.5. (Page 80) Let $*$ be a prioritized multiple base revision operator and let $f_{\mathcal{B}}$ satisfy *Inclusion*$_f$, *Weak extensionality*$_f$, *Consistency preservation*$_f$, and *Weak maximality*$_f$. Then $\star_{f_{\mathcal{B}}}$ defined via (4.2.2) is a non-prioritized multiple base revision operator.

Proof. We have to show that $\star_{f_{\mathcal{B}}}$ satisfies *Inclusion*$_\Phi$, *Consistency*$_\Phi$, *Weak extensionality*$_\Phi$, *Weak success*$_\Phi$, and *Consistent expansion*$_\Phi$:

INCLUSION$_\Phi$ It holds that $f_{\mathcal{B}}(\Phi) \subseteq \Phi$ as $f_{\mathcal{B}}$ satisfies *Inclusion*$_f$. Also, $*$ satisfies *Inclusion*$_\Phi$ and it follows $\mathcal{B} * f_{\mathcal{B}}(\Phi) \subseteq \mathcal{B} \cup f_{\mathcal{B}}(\Phi) \subseteq \mathcal{B} \cup \Phi$.

CONSISTENCY$_\Phi$ If Φ is consistent also $f_{\mathcal{B}}(\Phi)$ is consistent as $f_{\mathcal{B}}$ satisfies *Consistency preservation*$_f$. As $*$ satisfies *Consistency*$_\Phi$ it follows that $\mathcal{B} * f_{\mathcal{B}}(\Phi)$ is consistent.

WEAK EXTENSIONALITY$_\Phi$ If $\Phi \cong^p \Phi'$ then $f_{\mathcal{B}}(\Phi) \cong^p f_{\mathcal{B}}(\Phi')$ as $f_{\mathcal{B}}$ satisfies *Weak extensionality*$_f$. It follows that $\mathcal{B} * f_{\mathcal{B}}(\Phi) \equiv^p \mathcal{B} * f_{\mathcal{B}}(\Phi')$ as $*$ satisfies *Weak extensionality*$_\Phi$.

WEAK SUCCESS$_\Phi$ If $\mathcal{B} \cup \Phi$ is consistent it follows that $f_\mathcal{B}(\Phi) = \Phi$ as $f_\mathcal{B}$ satisfies *Weak maximality*$_f$. As $*$ satisfies *Vacuity*$_\Phi$ it follows $\mathcal{B} + \Phi \subseteq \mathcal{B} * f_\mathcal{B}(\Phi)$. Hence, $\star_{f_\mathcal{B}}$ satisfies vacuity as well. As noted before and as proven in [96], the satisfaction of *Vacuity*$_\Phi$ implies the satisfaction of *Weak success*$_\Phi$.

CONSISTENT EXPANSION$_\Phi$ Suppose $\mathcal{B} \not\subseteq \mathcal{B} * f_\mathcal{B}(\Phi)$. The $*$ operator satisfies *Consistent expansion*$_\Phi$ as $*$ satisfies *Vacuity*$_\Phi$ and *Success*$_\Phi$, cf. [96]. It follows that $\mathcal{B} \cup \{\mathcal{B} * f_\mathcal{B}(\Phi)\}$ is inconsistent.

□

Proposition 4.2.13. (Page 84) Let γ be a well-behaving categorizer and κ be a well-behaving accumulator. Then the selection functions $S^{\gamma,\kappa}_\mathcal{B}$ and $C^{\gamma,\kappa}_\mathcal{B}$ satisfy *Inclusion*$_f$, *Weak inclusion*$_f$, *Weak extensionality*$_f$, *Consistency preservation*$_f$ and *Weak maximality*$_f$.

Proof.

INCLUSION$_f$ *Inclusion*$_f$ is satisfied by definition as for $\alpha \in S^{\gamma,\kappa}_\mathcal{B}(\Phi)$ and each $\alpha \in C^{\gamma,\kappa}_\mathcal{B}(\Phi)$ it follows $\alpha \in \Phi$.

WEAK INCLUSION$_f$ *Weak inclusion*$_f$ follows directly from the satisfaction of *Inclusion*$_f$.

WEAK EXTENSIONALITY$_f$ Let $\Phi \cong^p \Phi'$ and let $\sigma : \Phi \to \Phi'$ be a bijection such that for every $\phi \in \Phi$ it holds that $\phi \equiv^p \sigma(\phi)$. We extend σ to $\mathcal{B}$ via $\sigma(\psi) = \psi$ for every $\psi \in \mathcal{B}$. For $\Psi \subseteq \mathcal{B} \cup \Phi$ we abbreviate

$$\sigma(\Psi) = \bigcup_{\psi \in \Psi} \{\sigma(\psi)\}.$$

Let $\langle \Psi, \phi \rangle$ be an argument for some $\phi \in \Phi$ with respect to $\mathcal{B} \cup \Phi$. Then $\langle \sigma(\Psi), \sigma(\phi) \rangle$ is an argument for $\sigma(\phi)$ in $\mathcal{B} \cup \Phi'$. It follows that if τ is an argument tree for $\langle \Psi, \phi \rangle$ in $\mathcal{B} \cup \Phi$ then τ' is an argument tree for $\langle \sigma(\Psi), \sigma(\phi) \rangle$ in $\mathcal{B} \cup \Phi'$ where τ' is obtained from τ by replacing each sentence ϕ with $\sigma(\phi)$. This generalizes also to argument structures and it follows that

$$\kappa(\gamma(\Gamma_{\mathcal{B}\cup\Phi}(\phi))) = \kappa(\gamma(\Gamma_{\mathcal{B}\cup\Phi'}(\sigma(\phi)))).$$

Hence, $\phi \in S^{\gamma,\kappa}_\mathcal{B}(\Phi)$ if and only if $\sigma(\phi) \in S^{\gamma,\kappa}_\mathcal{B}(\Phi')$ for every $\phi \in \Phi$. It follows that $S^{\gamma,\kappa}_\mathcal{B}(\Phi) \cong^p S^{\gamma,\kappa}_\mathcal{B}(\Phi')$. The same argumentation holds for $C^{\gamma,\kappa}_\mathcal{B}$.

CONSISTENCY PRESERVATION$_f$ Every subset of a consistent set of sentences is consistent. From the satisfaction of *Inclusion*$_f$ follows that $S^{\gamma,\kappa}_\mathcal{B}(\Phi) \subseteq \Phi$ and $C^{\gamma,\kappa}_\mathcal{B}(\Phi) \subseteq \Phi$.

WEAK MAXIMALITY$_f$ If $\mathcal{B} \cup \Phi$ is consistent, then for all arguments for a sentence $\alpha \in \Phi$ there do not exist any undercuts as these would have to entail the negation of some sentence of the argument for α which implies inconsistency of $\mathcal{B} \cup \Phi$. The argument structure $\Gamma_\Phi(\alpha) = (\mathcal{T}^+, \mathcal{T}^-)$ consists of one or more single node trees $\mathcal{T}^+$ and $\mathcal{T}^- = \emptyset$. As both γ and κ are well-behaving it follows that $\kappa(\gamma(\Gamma_\Phi(\alpha))) > 0$ for each $\alpha \in \Phi$ and therefore $S^{\gamma,\kappa}_{\mathcal{B}}(\Phi) = \Phi$ and $C^{\gamma,\kappa}_{\mathcal{B}}(\Phi) = \Phi$.

□

Corollary 4.2.14. (Page 84) Let γ be a well-behaving categorizer and let κ be a well-behaving accumulator. Then both $\circ^{\gamma,\kappa}_S$ and $\circ^{\gamma,\kappa}_C$ are non-prioritized multiple base revision operators.

Proof. This follows directly from Propositions 4.2.5 and 4.2.13. □

Lemma 4.3.2. (On Page 87) If $*$ satisfies *Fullness*$_Q$, then it satisfies *Relevance*$_Q$.

Proof. If $*$ satisfies *Fullness*$_Q$, then if $r \in (P \cup Q) \setminus (P * Q)$, then $P * Q$ is consistent and $(P * Q) \cup \{r\}$ is inconsistent. If we set $H = P * Q$ it satisfies $P * Q \subseteq H \subseteq P \cup Q$ and H is consistent but $H \cup \{r\}$ is inconsistent such that $*$ satisfies *Relevance*$_Q$. □

Proposition 4.3.4. (On Page 88) If a revision operator $*$ satisfies *NM-Consistency*$_Q$, then it satisfies *Consistency*$_Q$.

Proof. Let $*$ be a revision operator that satisfies *NM-Consistency*$_Q$. Consistency$_Q$ is only applicable if Q is consistent. We set $X = Q$, the premise of *NM-Consistency*$_Q$ is satisfied becaus $Q \subseteq X \subseteq P \cup Q$ holds such that $P * Q$ is consistent. Hence *Consistency*$_Q$ is satisfied. □

Proposition 4.3.6. (On Page 88) If a revision operator $*$ satisfies *NM-Fullness*$_Q$, then it satisfies *Fullness*$_Q$ and *Vacuity*$_Q$.

Proof. If $r \in (P \cup Q) \setminus (P * Q)$, then $(P \cup Q) \setminus P * Q \neq \emptyset$. By *NM-Fullness*$_Q$ it holds that $P * Q$ is consistent and since $\{r\} \subseteq (P \cup Q) \setminus P * Q$ it holds that $(P * Q) \cup \{r\}$ is inconsistent. Hence, *Fullness*$_Q$ is satisfied.

If $P + Q$ is consistent, then for any $P * Q$ with $R = (P \cup Q) \setminus P * Q \neq \emptyset$ it holds that $(P * Q) \cup R = P + Q$ is consistent such that $*$ violates *NM-Fullness*$_Q$. Hence *NM-Fullness*$_Q$ can only be satisfied by not satisfying the precondition $(P \cup Q) \setminus P * Q \neq \emptyset$. This is the case if and only if $(P \cup Q) \setminus P * Q = \emptyset$ such that $P + Q = P * Q$. Hence, *Vacuity*$_Q$ is satisfied. □

Proposition 4.3.9. (On Page 90) Let $*$ be a revision operator on logic programs.

1. If $*$ satisfies *Success*$_Q$ and *Inclusion*$_Q$, then it satisfies Initialisation$_P$.
2. If $*$ satisfies *Success*$_Q$ and *Inclusion*$_Q$, then it satisfies Idempotence$_P$.
3. If $*$ satisfies *Success*$_Q$, *NM-Consistency*$_Q$, *Inclusion*$_Q$ and *NM-Fullness*$_Q$, then it satisfies Absorption$_P$.

Proof.

1. From *Success*$_Q$ follows $P \subseteq \emptyset * P$ and from *Inclusion*$_Q$ follows $\emptyset * P \subseteq P$.
2. From *Success*$_Q$ follows $P \subseteq P * P$ and from *Inclusion*$_Q$ follows $P * P \subseteq P$.
3. We first show that $(P * Q) * Q \subseteq P * Q$:
   From *Inclusion*$_Q$ follows that $(P * Q) * Q \subseteq (P * Q) \cup Q$ and from *Success*$_Q$ follows that $(P * Q) \cup Q = P * Q$ such that $(P * Q) * Q \subseteq P * Q$.

   Now it is left to show that $P * Q \subseteq (P * Q) * Q$:
   Assume to the contrary that $P * Q \nsubseteq (P * Q) * Q$. Then $P * Q \setminus (P * Q) * Q \neq \emptyset$. By *Success*$_Q$ it holds that $P * Q = (P * Q) \cup Q$ such that $(P * Q) \cup Q \setminus (P * Q) * Q \neq \emptyset$, which is the premise of *NM-Fullness*$_Q$. By *NM-Fullness*$_Q$ it now follows that $(P * Q) * Q$ is consistent and $(P * Q) \cup Q = P * Q$ is inconsistent. This is a contradiction to the assumption that $*$ satisfies *NM-Consistency*$_Q$ since $Q \subseteq (P * Q) * Q \subseteq P \cup Q$ and $(P * Q) * Q$ is consistent it follows from *NM-Consistency*$_Q$ that $P * Q$ is consistent.

$\square$

Proposition 4.3.11. (On Page 91) Let $*$ be a revision operator on logic programs.

1. If $*$ satisfies *Consistency*$_Q$, then it violates Tautology$_{AS}$.
2. If $*$ satisfies *Success*$_Q$, then it violates Tautology$_P$.
3. If $*$ satisfies *Vacuity*$_Q$, then it violates Tautology$_P$.

Proof.

1. Given a program P with $AS(P) = \emptyset$ and a tautologic program $P_\top$ then consistency demands that $AS(P * P_\top) \neq \emptyset$, in contradiction to Tautology$_{AS}$

2. Given a program P and a tautologic program $P_\top$ such that $P \cap P_\top = \emptyset$. It follows from *Success*$_Q$ that $P * P_\top \neq P$, in contradiction to Tautology$_P$.

3. Given a program P and a tautologic program $P_\top$ such that $P \cup P_\top$ is consistent. It follows from *Vacuity*$_Q$ that $P \cup P_\top \subseteq P * P_\top$, in contradiction to Tautology$_P$.

$\square$

Proposition 4.3.12. (On Page 91) Let $*$ be a revision operator on logic programs. If $*$ satisfies Consistent Irrelevance$_\bullet$ and $\bullet \in \{AS, UE, SE\}$, then $*$ satisfies Consistent Tautology$_\bullet$.

Proof. If $*$ satisfies Consistent Irrelevance$_{SE}$, then for $P_\top$ it directly follows that $P * P_\top \equiv_{SE} P$. $\square$

Proposition 4.3.13. (On Page 92) Let $*$ be a revision operator on logic programs. If $*$ satisfies *Inclusion*$_Q$ and *Vacuity*$_Q$, then it satisfies Consistent Irrelevance$_\bullet$ for $\bullet \in \{AS, UE, SE, P\}$.

Proof. Assume that P is consistent, i.e., $AS(P) \neq \emptyset$, $P \equiv_\bullet P \cup Q$. Further assume that $*$ satisfies *Inclusion*$_Q$ and *Vacuity*$_Q$. Since P is consistent it follows from *Vacuity*$_Q$ that $P + Q \subseteq P * Q$, and together with *Inclusion*$_Q$ follows $P + Q = P * Q$. Hence it holds that $P + Q \equiv_\bullet P * Q$ for $\bullet \in \{AS, UE, SE, P\}$ and from the premise follows that $P * Q \equiv_\bullet P$ such that Irrelevance$_\bullet$ is satisfied. $\square$

Corollary 4.3.14. (On Page 92) If $*$ satisfies *Inclusion*$_Q$ and *Vacuity*$_Q$, then it satisfies Consistent Tautology$_\bullet$ for $\bullet \in \{AS, UE, SE\}$.

Proof. If $*$ satisfies *Inclusion*$_Q$ and *Vacuity*$_Q$ then it satisfies Consistent Irrelevance$_\bullet$ for $\bullet \in \{AS, UE, SE, P\}$ by Proposition 4.3.13. Then it follows from Proposition 4.3.12 that $*$ satisfies Consistent Tautology$_\bullet$ for $\bullet \in \{AS, UE, SE\}$. $\square$

Proposition 4.3.16. (On Page 92) Let $*$ be a revision operator on logic programs.

1. If $*$ satisfies *Success*$_Q$, *Consistency*$_Q$ and *Vacuity*$_Q$, then it violates *Parallelism*$_{AS}$.

2. If $*$ satisfies *Vacuity*$_Q$ and *Consistency*$_Q$, then it violates Disjointness$_{AS}$.

Proof.

1. Given Q_1 and Q_2 with disjoint sets of literals and some program P. Assume that $Q_1 \cup P$ is inconsistent and $Q_2 \cup P$ is consistent and that P, Q_1 and Q_2 are strict programs. From the

satisfaction of *Consistency*$_Q$ follows that $AS(P * (Q_1 \cup Q_2)) \neq \emptyset$. It follows from *Vacuity*$_Q$ that $P \subseteq P * Q_2$ and from *Success*$_Q$ that $Q_1 \subseteq P * Q_1$. For the satisfaction of *Parallelism*$_{AS}$ it has to be the case that $AS(P * (Q_1 \cup Q_2)) = AS((P * Q_1) \cup (P * Q_2))$. It holds that $P \cup Q_1 \subseteq (P * Q_1) \cup (P * Q_2)$. Since $P \cup Q_1$ is inconsistent and P and Q_1 are strict programs it holds that $AS((P * Q_1) \cup (P * Q_2)) = \emptyset$, in contradiction to *Consistency*$_Q$.

2. Given $P = P_1 \cup P_2$ and P_1 and P_2 have disjoint sets of literals and some program Q such that $P_1 \cup Q$ and $P_2 \cup Q$ are consistent and $P \cup Q$ is inconsistent. Assume P and Q are strict programs. It follows from *Consistency*$_Q$ that $AS(P * Q) \neq \emptyset$. From *Vacuity*$_Q$ it follows that $P_1 \cup Q \subseteq P_1 * Q$ and $P_2 \cup Q \subseteq P_2 * Q$. Disjointness$_{AS}$ demands that $AS(P * Q) = AS((P_1 * Q) \cup (P_2 * Q))$. It holds that $P \cup Q \subseteq (P_1 * Q) \cup (P_2 * Q)$. Since $P \cup Q$ is inconsistent and P and Q are strict programs we have that $AS((P_1 * Q) \cup (P_2 * Q)) = \emptyset$ in contradiction to *Consistency*$_Q$.

□

Proposition 4.3.18. (On Page 95) Let $!_R$ be a screened consolidation operator. If $!_R$ satisfies *Fullness*$_!$, then it satisfies *Relevance*$_!$.

Proof. Let $r \in P$ and $r \notin P!_R$. We have $P!_R \subseteq P!_R \subseteq P$ and from *Fullness*$_!$ it follows that $P!_R$ is consistent and $P!_R \cup \{r\}$ is inconsistent such that *Relevance*$_!$ is satisfied. □

Proposition 4.3.19. (On Page 95) Let $!_R$ be a screened consolidation operator. If $!_R$ satisfies *NM-Fullness*$_!$, then it satisfies *Fullness*$_!$.

Proof. If $r \in P \setminus P!_R$, then $P \setminus P!_R \neq \emptyset$. By *NM-Fullness*$_Q$ it holds that $P!_R$ is consistent and, since $\{r\} \subseteq P \setminus P!_R$, that $P!_R \cup \{r\}$ is inconsistent. Hence, *Fullness*$_!$ is satisfied. □

Proposition 4.3.23. (On Page 96) Let $*$ be a multiple base revision operator defined as $P * Q = (P \cup Q)!_Q$. If $!_R$ is a screened consolidation operator that satisfies *Topic-Independence*$_!$, then $*$ satisfies *Success*$_Q$, *Inclusion*$_Q$, *Vacuity*$_Q$, *Consistency*$_Q$, *NM-Consistency*$_Q$, *Relevance*$_Q$, *Fullness*$_Q$, *NM-Fullness*$_Q$, *Uniformity*$_Q$, *Weak-Disjointness*$_P$ and *Weak-Parallelism*$_P$.

Proof. Screen$_!$ implies that $Q \subseteq (P \cup Q)!_Q = P * Q$, i. e., *Success*$_Q$. From the satisfaction of *Inclusion*$_!$ follows that $(P \cup Q)!_Q = P * Q \subseteq P \cup Q$, i. e., *Inclusion*$_Q$. From Screen-Consistency$_!$ follows that if Q is consistent then $(P \cup Q)!_Q = P * Q$ is consistent, i. e., *Consistency*$_Q$. The satisfaction of NM-Fullness$_!$ implies that if $r \in (P \cup Q) \setminus P * Q$ then $P * Q$ is consistent and $(P * Q) \cup \{r\}$ is inconsistent, i. e., *Fullness*$_Q$ is satisfied. Note that *Relevance*$_Q$ follows from *Fullness*$_Q$ and *Vacuity*$_Q$ follows from *Relevance*$_Q$ and *Inclusion*$_Q$.

For the proofs of *Weak Disjointness*$_P$ and Weak Parallelism$_P$ we first introduce some definitions:

Definition A.1.1. Given programs P and R such that $R \subseteq P$. We define a closure operator $Cl_P(R)$ of R with respect to P as the set satisfying:

1. $R \subseteq Cl_P(R)$
2. There is no rule $r \in P \setminus R$ such that $\mathcal{A}(r) \cap \mathcal{A}(R) \neq \emptyset$
3. There is no set $R' \subset R$ satisfying 1. and 2.

Clearly it holds that $\mathcal{A}(Cl_P(R)) \cap P \setminus Cl_P(R) = \emptyset$.

Lemma A.1.2. Given programs R, T, P such that $R \subseteq T \subseteq P$ and $R = Cl_P(R)$. T is consistent if and only if R is consistent and $T \setminus R$ is consistent.

Proof. Since $\mathcal{A}(R) \cap \mathcal{A}(T) = \emptyset$ $\mathcal{A}(R)$ is a splitting set for T and the lemma follows directly form the splitting set theorem [168]. □

Weak Disjointness$_P$:

We show now that $\{P_1 \cup Q, P_2 \cup Q\}$ is a topicalization of P. By definition it holds that $(P_1 \cup Q) \cup (P_2 \cup Q) = P$. We have to show that for each $R \subset P$ it holds that R is consistent if $R \cap (P_1 \cup Q)$ is consistent and $R \cap (P_1 \cup Q)$ is consistent. Now assume that $R \cap (P_1 \cup Q)$ is consistent and $R \cap (P_1 \cup Q)$ is consistent. We can partition $P_1 \cup Q$ into $Cl_R(P_1)$ and $(P_1 \cup Q) \setminus Cl_R(P_1)$. From Lemma 1 follows that $Cl_R(P_1)$ is consistent and $Q_1 =_{def} (P_1 \cup Q) \setminus Cl_R(P_1)$ is consistent. Analogously we obtain $Cl_R(P_2)$ is consistent and $Q_2 =_{def} (P_2 \cup Q) \setminus Cl_R(P_2)$ is consistent.

For $Q_1 \cap Q_2$ it holds that $Cl_R(Q_1 \cap Q_2) = Q_1 \cap Q_2$ and from $Q_1 \cap Q_2 \subseteq Q_1$ follows that $Q_1 \cap Q_2$ is consistent. Consequently also $Cl_R(P_1) \cup (Q_1 \cap Q_2)$ is consistent. Since $R = (Cl_R(P_1) \cup (Q_1 \cap Q_2)) \cup Cl_R(P_2)$ it follows from $(Cl_R(P_1) \cup (Q_1 \cap Q_2))$ being consistent and $Cl_R(P_2)$ being consistent by Lemma 1 that R is consistent. This proves the claim such that $\{P_1 \cup Q, P_2 \cup Q\}$ is a topicalization of P.

To proof *Weak Disjointness*$_P$ we need to show that $(P \cup Q)!_Q = (P_1 \cup Q)!_Q \cup (P_2 \cup Q)!_Q$. From the above considerations we know that $(P_1 \cup Q)!_Q \cup (P_2 \cup Q)!_Q = (P_1 \cup Q)!_Q \cup (P_2 \cup Q)!_Q$ since $Q \cup P_2$ and $Q \cup P_1$ are consistent and $\{P_1 \cup Q, P_2 \cup Q\}$ being a topicalization of $P \cup Q$. From *Inclusion*$_!$ follows that $(P \cup Q)!_Q \subseteq P \cup Q$ such that we can write $(P \cup Q)!_Q = (P \cup Q) \setminus X$. From Screen$_!$ follows that $X \subseteq P$. Applying the same arguments to the right hand side yields

$$(P \cup Q) \setminus X = ((P_1 \cup Q) \setminus X_1) \cup ((P_2 \cup Q) \setminus X_2)$$

and $X_1 \subseteq P_1$ and $X_2 \subseteq P_2$. From P_1 and P_2 being disjoint follows $X_1 \cap P_2 = \emptyset$ and $X_2 \cap P_1 = \emptyset$. Hence

$$\begin{aligned}&((P_1 \cup Q) \setminus X_1) \cup ((P_2 \cup Q) \setminus X_2) =\\&((P_1 \cup Q) \setminus (X_1 \cup X_2)) \cup ((P_2 \cup Q) \setminus (X_1 \cup X_2) =\\&((P_1 \cup Q) \cup (P_2 \cup Q)) \setminus (X_1 \cup X_2) =\\&(P \cup Q) \setminus (X_1 \cup X_2).\end{aligned}$$

This leaves to show that $(P \cup Q) \setminus X = (P \cup Q) \setminus (X_1 \cup X_2)$. From *Topic-Independence*$_!$ it follows that $(P \cup Q) \setminus (P \cup Q)!_Q = ((P_1 \cup Q) \setminus (P_1 \cup Q)!_Q$ and we have that $X = (P \cup Q) \setminus (P \cup Q)!_Q$, $X_1 = (P_1 \cup Q) \setminus (P_1 \cup Q)!_Q$ and $X_2 = (P_2 \cup Q) \setminus (P_2 \cup Q)!_Q$. Thus *Weak Disjointness*$_P$ is satisfied.

Weak Parallelism$_P$: We define $P_1 = P \setminus Cl_{P \cup Q_2}(Q_2)$ and $P_2 = P \setminus Cl_{P \cup Q_1}(Q_1)$. Clearly it holds that $P = P_1 \cup P_2$.

Following same argumentation as for *Weak Disjointness*$_P$ we can show that $\{P_1 \cup Q_1, P_2 \cup Q_2\}$ is a topicalization of $P \cup Q$.

We need to show that

$$(P \cup (Q_1 \cup Q_2))!_{(Q_1 \cup Q_2)} = (P \cup Q_1)!_{Q_1} \cup (P \cup Q_2)!_{Q_2}.$$

From *Inclusion*$_!$ follows that $(P \cup (Q_1 \cup Q_2))!_{(Q_1 \cup Q_2)} \subseteq P \cup (Q_1 \cup Q_2)$ such that we can write $(P \cup (Q_1 \cup Q_2))!_{(Q_1 \cup Q_2)} = (P \cup (Q_1 \cup Q_2)) \setminus X$. It is to show that

$$(P \cup (Q_1 \cup Q_2)) \setminus X = ((P \cup Q_1) \setminus X_1) \cup ((P \cup Q_2) \setminus X_2).$$

We show first that $(P_1 \cup Q_1) \setminus X_1 = (P_1 \cup Q_1) \setminus (X_1 \cup X_2)$ which holds if and only if $X_2 \cap ((P_1 \cup Q_1) \setminus X_1) = \emptyset$. From Screen$_!$ and *Inclusion*$_!$ it follows that $X_2 \subseteq P$ and $X_2 \cap (Q_1 \cup Q_2) = \emptyset$.

Hence it is left to show that $X_2 \cap (P_1 \setminus X_1) = \emptyset$. Let $r \in X_2 \cap P_1$. From NM-Fullness$_!$ follows that for each $r \in X_2$ it holds that $((P \cup Q_2) \setminus X_2) \cup \{r\}$ is inconsistent and therefore that $r \in Cl_{P \cup Q}((P \cup Q_2) \setminus X_2)$. Therefore $r \notin (P_1 \setminus P_2)$ by definition of P_1 and P_2 and consequently $r \in P_1 \cap P_2$.

Therefore $X_2 \cap (P_1 \setminus X_1) = \emptyset$ if and only if $X_1 \cap (P_1 \cap P_2) = X_2 \cap (P_1 \cap P_2)$. $\{(P_1 \setminus P_2) \cup Q_1, P_1 \cap P_2\}$ is a topicalization of $P_1 \cup Q_1$ and $\{(P_2 \setminus P_1) \cup Q_2, P_1 \cap P_2\}$ is a topicalization of $P_2 \cup Q_2$. From *Topic-Independence*$_!$ follows that $X_1 \cap (P_1 \cap P_2) = X_2 \cap (P_1 \cap P_2)$. That is we have shown that $(P_1 \cup Q_1) \setminus X_1 = (P_1 \cup Q_1) \setminus (X_1 \cup X_2)$. Analogously follows that $(P_2 \cup Q_2) \setminus X_2 = (P_2 \cup Q_2) \setminus (X_1 \cup X_2)$.

Having shown this we can use the same argumentation from the proof for Weak-Disjointness, yielding:

$$\begin{aligned}&((P_1 \cup Q_1) \setminus X_1) \cup ((P_2 \cup Q_2) \setminus X_2) =\\&((P_1 \cup Q_1) \setminus (X_1 \cup X_2)) \cup ((P_2 \cup Q_2) \setminus (X_1 \cup X_2) =\\&((P_1 \cup Q_1) \cup (P_2 \cup Q_2)) \setminus (X_1 \cup X_2) =\\&(P \cup Q) \setminus (X_1 \cup X_2).\end{aligned}$$

This leaves to show that $(P \cup Q) \setminus X = (P \cup Q) \setminus (X_1 \cup X_2)$. From *Topic-Independence*$_!$ it follows that $(P \cup Q) \setminus (P \cup Q)!_Q = ((P \cup Q_1) \setminus (P \cup Q_1)!_{Q_1} \cup ((P \cup Q_2) \setminus (P \cup Q_2)!_{Q_2}$ and we have that $X = (P \cup Q) \setminus (P \cup Q)!_Q$, $X_1 = (P_1 \cup Q) \setminus (P \cup Q_1)!_{Q_1}$ and $X_2 = (P \cup Q_2) \setminus (P \cup Q_2)!_{Q_2}$. Thus Weak Parallelism$_P$ is satisfied. □

Proposition 4.3.28. (On Page 98) An operation $!_R$ is an operation of screened maxichoice consolidation if and only if it satisfies *Inclusion*$_!$, Screen-Consistency$_!$, Screen$_!$, NM-Fullness$_!$ and C-Uniformity$_Q$.

Proof. Construction to postulates:

Screen$_!$ We need to show that for all $R \subseteq P$, $R \subseteq \gamma_P(P \perp_! R)$. If $P \perp_! R = \emptyset$ then $\gamma_P(P \perp_! R) = P$ and $R \subseteq P$ by definition. If $P \perp_! R \neq \emptyset$ then by definition of screened remainder sets $R \subseteq X$ for all $X \in (P \perp_! R)$ it directly follows that $R \subseteq \gamma_P(P \perp_! R)$.

Screen-Consistency$_!$ We need to show that for all $R \subseteq P$, if there exists some consistent X, $R \subseteq X \subseteq P$ then $\gamma_P(P \perp_! R)$ is consistent. In the case of $P \perp_! R = \emptyset$ there does not exists a consistent set X, $R \subseteq X \subseteq P$. In the case of $P \perp_! R \neq \emptyset$ by definition of screened remainder sets each $X \in (P \perp_! R)$ is consistent and by definition $\gamma(P \perp_! R)$ is consistent.

Inclusion$_!$ We need to show that for all $R \subseteq P$, $\gamma_P(P \perp_! R) \subseteq P$. If $P \perp_! R = \emptyset$ then $\gamma_P(P \perp_! R) = P$. If $P \perp_! R \neq \emptyset$ then by definition of screened remainder sets for all $X \in (P \perp_! R)$, $X \subseteq P$ and thus by definition $\gamma(P \perp_! R) \subseteq P$.

NM-Fullness$_!$ We need to show that for all R, if $r \in P$ and $r \notin \gamma(P \perp_! R)$ then $\gamma(P \perp_! R)$ is consistent and $\gamma(P \perp_! R) \cup \{r\}$ is inconsistent. If $P \perp_! R = \emptyset$ then $\gamma_P(P \perp_! R) = P$ and there is no $r \in R$ and $r \notin \gamma(P \perp_! R)$ such that NM-Fullness$_!$ is satisfied vacuously. If $P \perp_! R \neq \emptyset$ and $r \notin \gamma(P \perp_! R)$ then from $\gamma(P \perp_! R) = X$ for some $X \in P \perp_! R$ it follows that X is consistent and from condition 3 of Definition 4.3.24 and $X \subset X \cup \{r\} \subseteq P$ it follows that $X \cup \{r\}$ is inconsistent such that NM-Fullness$_!$ is satisfied.

Screen-Uniformity$_!$ We need to show that all R, R' and P, and a selection function for P it holds that if for all $X \subseteq P$, $R \cup X$ is consistent iff $R' \cup X$ is consistent then $\gamma(P \perp_! R) \cap P = \gamma(P \perp_! R') \cap P$. If for all $X \subseteq P$, $R \cup X$ is consistent iff $R' \cup X$ is consistent then it follows that $P \perp_! R = P \perp_! R'$ and by definition of a selection function it holds that $\gamma(P \perp_! R) = \gamma(P \perp_! R')$.

Postulates to construction: Let $!_R$ be an operation for P that satisfies Screen$_!$, Screen-Consistency$_!$, *Inclusion*$_!$, NM-Fullness$_!$
and C-Uniformity$_Q$.

Let γ be such that: $\gamma(P \perp_! R) = P!_R$ We need to show that (1) γ is a well-defined function, that (2) γ is a maxichoice selection function, and that (3) for all R, $\gamma(P \perp_! R) = P!_R$.

Part (1): γ is a well-defined function if for all $P \perp_! R = P \perp_! R'$ it holds that $\gamma(P \perp_! R) = \gamma(P \perp_! R')$. Suppose $P \perp_! R = P \perp_! R'$ then it follows from Screen-Uniformity$_!$ that $P!_R = P!_{R'}$. By definition of γ it follows that $\gamma(P \perp_! R) = \gamma(P \perp_! R')$.

Part (2): For γ to be a maxichoice selection function we have to show that if $P \perp_! R \neq \emptyset$ then $\gamma(P \perp_! R) = X$ for some $X \in P \perp_! R$ and that if $P \perp_! R = \emptyset$ then $\gamma(P \perp R_!) = P$.

If $P \perp_! R \neq \emptyset$ there exists some consistent X, $R \subseteq X \subseteq P$ and it follows from *Screen Consistency*$_!$ that $P\,!_R$ is consistent. From *Inclusion*$_!$ follows that $P!_R \subseteq P$ and from Screen$_!$ that $R \subseteq P!_R$. To show that $P!_R \in P \perp_! R$ it is left to show that there is no H such that $P!_R \subset H \subseteq P$ and H is consistent. Let H be such that $P!_R \subset H \subseteq P$. Then for $R' = H \setminus P!_R$ it holds that $R' \subseteq P \setminus P!_R$ such that it follows from NM-Fullness$_!$ that $P!_R \cup R' = H$ is inconsistent.

If $P \perp_! R = \emptyset$ then there does not exist a consistent set X, $R \subseteq X \subseteq P$. Suppose to the contrary that $P \not\subseteq P!_R$ and let $R = P \setminus P!_R$. Then it follows from C-Screen-Fullness$_Q$ that $P!_R$ is consistent. From Screen$_!$ it follows that $R \subseteq P!_R$ and from *Inclusion*$_!$ that $P!_R \subseteq P$. This is a contradiction. Therefore we have shown that $P \subseteq P!_R$ and since it follows from inclusion that $P!_R \subseteq P$ we have $P = P!_R$ such that by definition of γ we have $\gamma(P \perp_! R) = P$ which was to show.

Part (3): That for all R, $\gamma(P \perp_! R) = P!_R$ follows directly from the construction. $\square$

Proposition 4.3.30. (On Page 98) If a screened maxichoice consolidation operator is based on a monotone maxichoice selection function, then it satisfies *Topic-independence*.

Proof. Let $\mathcal{B}$ be a topicalization of P. Then for each $B \in \mathcal{B}$ it holds that for each $X \in B \perp_! \emptyset$ there exists some $X' \in P \perp_! \emptyset$ such that $B \setminus X = B \setminus X'$. By monotony of γ it follows that $B \setminus \gamma(B \perp_! \emptyset) = B \setminus \gamma(P \perp_! \emptyset)$ for each $B \in \mathcal{B}$. Since $P = \bigcup \mathcal{B}$ it follows that $P \setminus \gamma(P \perp_! \emptyset) = \bigcup_{B \in \mathcal{B}}(B \setminus \gamma(B \perp_! \emptyset))$ and therefore $P \setminus P! = \bigcup_{B \in \mathcal{B}}(B \setminus B!)$ such that *Topic-Independence*$_!$ is satisfied. $\square$

A.1.3 *Proofs for Chapter 5*

Observation 5.3.4. (Page 117) Let $V_W(\mathcal{K})$ and $V_{\mathcal{A}}(\mathcal{K})$ be belief bases, let $\circ^a$ be a change operator, let $*^a_{\mathcal{L}_W}, *^a_{\mathcal{L}_V}$ be multiple belief base change operators, and t^a_W and t^a_V interpretation functions such that Equations 5.3.1 and 5.3.2 are satisfied. If $*^a_{\mathcal{L}_W}, *^a_{\mathcal{L}_V}$ satisfy *Success*, then $\circ$ satisfies *Awareness*$_a$.

Proof. By Equations (5.3.1) and (5.3.2) (On Page 115) it is

$$V_W(\mathcal{K} \circ^a a) = V_W(\mathcal{K}) *^a_{\mathcal{L}_W} t^a_W(a)$$

and

$$V(\mathcal{K} \circ^a a) = V_{\mathcal{A}}(\mathcal{K}) *^a_{\mathcal{L}_V} t^a_V(\mathcal{A}, a)$$

such that $t^a_W(a) \in V_W(\mathcal{K} \circ^a a)$ if and only if $V_W(\mathcal{K}) *^a_{\mathcal{L}_W} t^a_W(a)$ and $t^a_V(a, \mathcal{A}) \in V_{\mathcal{A}}(\mathcal{K} \circ^a a)$ if and only if $t^a_V(a, \mathcal{A}) \in V_{\mathcal{A}}(\mathcal{K}) *^a_{\mathcal{L}_V} t^a_V(\mathcal{A}, a)$. Given that $*^a_{\mathcal{L}_W}$ and $*^a_{\mathcal{L}_V}$ satisfy Success it holds that

$$t^a_W(a) \in V_W(\mathcal{K}) *^a_{\mathcal{L}_W} t^a_W(a)$$

and

$$t^a_V(a, \mathcal{A}) \in V_{\mathcal{A}}(\mathcal{K}) *^a_{\mathcal{L}_V} t^a_V(\mathcal{A}, a)$$

and therefore also $t^a_W(a) \in V_W(\mathcal{K} \circ^a a)$ and $t^a_V(a, \mathcal{A}) \in V_{\mathcal{A}}(\mathcal{K} \circ^a a)$ such that *Awareness* is satisfied. □

Proposition 5.3.5. (Page 118) Let Ξ be a set of belief operators, V an agent-view and $(\phi, \mathsf{Bel}, \mathcal{A})$ a secret. If $\mathsf{Bel}' = \mathsf{maxbel}_{(\Xi, \preceq_{\mathsf{bel}})}(V, \phi, \mathsf{Bel})$, $\mathsf{Bel}' \neq \perp$, then $\mathsf{Bel} \not\prec_{\mathsf{bel}} \mathsf{Bel}'$.

Proof. Assume to the contrary that $\mathsf{Bel}' = \mathsf{maxbel}_{(\Xi, \preceq_{\mathsf{bel}})}(V, \phi, \mathsf{Bel})$, $\mathsf{Bel}' \neq \perp$ and $\mathsf{Bel} \prec_{\mathsf{bel}} \mathsf{Bel}'$. We have to distinguish two cases:

1. If $\phi \notin \mathsf{Bel}(V)$, then $\mathsf{Bel}' = \mathsf{Bel}$ by definition of $\mathsf{maxbel}_{(\Xi, \preceq_{\mathsf{bel}})}$. From the reflexivity of $\preceq_{\mathsf{bel}}$ it follows that $\mathsf{Bel} \not\prec_{\mathsf{bel}} \mathsf{Bel}$, in contradiction to the assumption.
2. If $\phi \in \mathsf{Bel}(V)$, then it follows from our assumption and by *credulity* of $\preceq_{\mathsf{bel}}$ that $\mathsf{Bel}(V) \subseteq \mathsf{Bel}'(V)$. It follows that $\phi \in \mathsf{Bel}'(V)$ in contradiction to the definition of $\mathsf{maxbel}_{(\Xi, \preceq_{\mathsf{bel}})}$.

□

Proposition 5.3.6. (On Page 118) Let $\mathcal{D}$ be an agent, with $\circ$ and $\circ_a$ defined as in Equations (5.3.1) to (5.3.6) (Page 115) with $*_{\mathcal{S}}$ as defined in Equation (5.3.9) on Page 118 above and inner revision operators $*_{\mathcal{L}_W}$ and $*_{\mathcal{L}_V}$ satisfying *Success* (defined in Section 2.5). The change operator $\circ$ satisfies *Acknowledgment*$_\circ$, *Min-Secrecy-Weakening*$_\circ$ and the change operator for actions $\circ_a$ satisfies *Secrets-Invariance*$_{\circ_a}$, *Awareness*$_{\circ_a}$.

Proof. From Equation (5.3.3) (On Page 115) follows directly that for all $a \in \mathsf{Act}$ it holds that $\mathcal{S}(\mathcal{K} \circ^a a) = \mathcal{S}(\mathcal{K})$ implies that *Secrets-Invariance* is satisfied. To show the satisfaction of *Acknowledgement* we show that for all $p \in \mathsf{Per}$ and for any secret $(\Phi, \mathsf{Bel}, \mathcal{A}) \in \mathcal{S}(\mathcal{K} \circ p)$ it holds that $\phi \notin \mathsf{Bel}(V(\mathcal{K} \circ p))$. By definition of $*_{\mathcal{S}}$ it holds that each secret of $\mathcal{S}(\mathcal{K} \circ p)$ is of the form

$$(\Phi, \mathsf{maxbel}_{(\Xi, \preceq_{\mathsf{bel}})}(V(\mathcal{K} \circ p), \Phi, \mathsf{Bel}), \mathcal{A}).$$

By definition of $\mathsf{maxbel}_{(\Xi, \preceq_{\mathsf{bel}})}$ it holds that if for some $\mathsf{Bel} \in \Xi$ it holds that $\Phi \notin V(\mathcal{K} \circ p)$, then

$$\Phi \notin \mathsf{maxbel}_{(\Xi, \preceq_{\mathsf{bel}})}(V, \Phi, \mathsf{Bel})(V(\mathcal{K} \circ p)).$$

It is left to show that there exists $\mathsf{Bel} \in \Xi$ such that $\Phi \notin V(\mathcal{K} \circ p)$. This is guaranteed by our assumption that $\mathsf{Bel}_{\emptyset} \in \Xi$ with $\mathsf{Bel}_{\emptyset}(E) = \emptyset$. The satisfaction of *Awareness* follows from the satisfaction of the *Success* postulate by $*$ and Proposition 5.3.4. $\square$

Proposition 5.4.3. (Page 120) Let $(\Lambda, \mathfrak{F})$ be a setting such that $\Lambda \subseteq \mathcal{L}_{ES}$ and all $(\circ, \circ^a, \mathsf{act}) \in \mathfrak{F}$ are of the type $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, $\circ^a : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$.

If $(\Lambda, \mathfrak{F})$ is plain, then any agent with an initial agent state $(\mathcal{K}^0, \xi) \in (\Lambda, \mathfrak{F})$ that has a safe initial epistemic state $\mathcal{K}^0$ and that selects a safe action if possible, i. e., for all $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}^0, \mathsf{Per})$ and all $p \in \mathsf{Per}$ if there is an $a \in \mathsf{Act}$ such that $(\mathcal{K} \circ p) \circ^a a$ is safe, then $\mathcal{K} \circ p \circ^a \mathsf{act}(\mathcal{K} \circ p)$ is safe, is secrecy preserving.

Proof. We prove the proposition by induction over the construction of $\mathcal{K} \in \Omega_{\xi}(\Lambda, \mathsf{Per})$.

Assume that $(\circ, \circ^a, \mathsf{act})$ with $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, $\circ^a : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$ satisfies that $\mathcal{K}^0$ is safe and for all $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}^0, \mathsf{Per})$ and all $p \in \mathsf{Per}$ if there is a safe $a \in \mathsf{Act}$ wrt. $\mathcal{K} \circ p$, then $\mathsf{act}(\mathcal{K} \circ p)$ is safe.

We have to show that all epistemic states in $\Omega_{\xi}(\mathcal{K}^0, \mathsf{Per})$ are safe. By definition of the possible states, Definition 5.2.6, for each $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}^0, \mathsf{Per})$ it holds that $\mathcal{K} = \mathcal{K}^0$ or $\mathcal{K} = \mathcal{K}' \circ p \circ^a \mathsf{act}(\mathcal{K}' \circ p)$ for some $\mathcal{K}' \in \Omega_{\xi}(\mathcal{K}^0, \mathsf{Per})$ and $p \in \mathsf{Per}$. Since $\mathcal{K}^0$ is safe by assumption that if some $\mathcal{K} \in \Omega_{\xi}(\Lambda, \mathsf{Per})$ is safe, then for all $p \in \mathsf{Per}$, $\mathcal{K} \circ p \circ^a \mathsf{act}(\mathcal{K} \circ p)$ is safe.

Since $\circ$ satisfies *Acknowledgement*$_{\circ}$ it holds that if $\mathcal{K} \in \Omega_{\xi}(\Lambda, \mathsf{Per})$ is safe, then for all $p \in \mathsf{Per}$, $\mathcal{K} \circ p$ is safe. From the setting $(\Lambda, \mathfrak{F}, \mathsf{Per})$ being plain it follows that for all $\mathcal{K} \in \Omega_{\xi}(\Lambda, \mathsf{Per}) \subseteq \mathcal{L}_{ES}$ and for all $p \in \mathsf{Per}$ there exists some $a \in \mathsf{Act}$ that is safe wrt. $\mathcal{K} \circ p$.

For all $\mathcal{K}^0 \in \Lambda$ it follows that $\mathcal{K}^0 \circ p \circ^a \mathsf{act}(\mathcal{K} \circ p)$ is safe and by our assumption it follows that if $\mathcal{K} \in \Omega_{\xi}(\Lambda, \mathsf{Per})$ is safe, then $\mathcal{K} \circ p \circ^a \mathsf{act}(\mathcal{K} \circ p)\}$ is safe. It follows by induction that all $\mathcal{K} \in \Omega_{\xi}(\Lambda, \mathsf{Per})$ are safe. $\square$

Lemma 5.4.8. (Page 122 Let $\mathcal{K} \in \mathcal{L}_{ES}$ be an epistemic state, Per a set of percepts, Act a set of actions, and $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$ a change operator. If $\mathcal{K}$ is sound with respect to $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$, then it is safe.

Proof. Let $\mathcal{K}$ be a sound state wrt. some $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$. For any strategy $s_{\mathcal{A}}$ of $\mathcal{A}$ all paths $\pi = (\mathcal{K}_0, \ldots, \mathcal{K}_n) \in W$ with $\mathcal{K}_0 = \mathcal{K}$ that are compliant with $s_{\mathcal{A}}$ the utility $u_{\mathcal{A}}(\pi) = 1$, that means that all $\mathcal{K} \in \pi$ are safe. Therefore, $\mathcal{K}$ is safe. $\square$

Proposition 5.4.9. (Page 123) Let $(\Lambda, \mathfrak{F})$ be a setting for which $\Lambda \subseteq \mathcal{L}_{ES}$ and all $(\circ, \circ^a, \mathsf{act}) \in \mathfrak{F}$ are of the type $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, $\circ^a : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$. An agent with initial agent state $(\mathcal{K}^0, \xi) \in (\Lambda, \mathfrak{F})$ is secrecy preserving if and only if all states $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ are sound.

Proof.
$\Rightarrow$: From Lemma 5.4.8 follows that if all states $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ are sound, then they are all safe and by Definition 5.2.7 it follows that $(\mathcal{K}^0, \xi)$ is secrecy preserving.

$\Leftarrow$: Let $(\mathcal{K}^0, \xi)$ be secrecy preserving. Assume that there is some $\mathcal{K}' \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ that is not sound wrt. $(\mathcal{L}_{ES}, \mathcal{K}', \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$. Then $\mathcal{A}$ has a winning strategy $s_\mathcal{A}$ in $g_{\mathcal{L}_{ES},\mathcal{K}',\mathsf{Per},\mathsf{Act},\circ,\circ^a}$ and in any path $\pi = (\mathcal{K}', \dots, \mathcal{K}_n)$ that complies with $s_\mathcal{A}$ there is some $\mathcal{K}'' \in \pi$ that is not safe, since $u_\mathcal{A} = 1$.

As $\circ$ satisfies *Acknowledgement* for all $\mathcal{K}_i \in \pi$ with $P(\mathcal{K}_i) = \mathcal{A}$ it is the case that $\mathcal{K}_{i+1} = \mathcal{K}_i \circ s_\mathcal{A}(\pi[i])$ is safe. Consequently there is some $\mathcal{K}_i \in \pi$ with $P(\mathcal{K}_i) = \mathcal{A}$ such that $\mathcal{K}_{i+1} = \mathcal{K}_i \circ s_\mathcal{A}(\pi[i])$ is not safe. Any such $\mathcal{K}_{i+1}$ is contained in $\Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ since it contains all $\{\mathcal{K} \mid \mathcal{K} = \mathcal{K}^0 \circ p_0 \circ^a \mathsf{act}_\mathcal{D}(\mathcal{K}^0 \circ p_0) \circ_\mathcal{D} \dots, p_0, \dots, p_i \subseteq \mathsf{Per}, i \in \mathbb{N}_0\}$. Hence, there is at least one state in $\Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ that is not safe. Consequently $(\mathcal{K}^0, \xi)$ is not secrecy preserving. □

Proposition 5.4.10. (Page: 123) Let $(\Lambda, \mathfrak{F})$ be a setting that is plain. For any $(\mathcal{K}^0, \xi) \in (\Lambda, \mathfrak{F})$ any $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ that is safe, is sound wrt. $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$.

Proof. Let $(\mathcal{K}^0, \xi) \in (\Lambda, \mathfrak{F})$, $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ and $\mathcal{K}$ be safe. Since $(\Lambda, \mathfrak{F})$ is plain for all $(\mathcal{K}^0, \xi) \in (\Lambda, \mathfrak{F})$ for all epistemic states $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ that are safe it holds that for all $p \in \mathsf{Per}$ there is some action $a \in \mathsf{Act}$ such that a is safe with respect to $\mathcal{K} \circ p$. Therefore, for any strategy $s_\mathcal{A}$ in $g_{\mathcal{L}_{ES},\mathcal{K},\mathsf{Per},\mathsf{Act},\circ,\circ^a}$ there is a strategy $s_\mathcal{D}$ and a path $\pi = (\mathcal{K}_0, \dots, \mathcal{K}_n)$ with $\mathcal{K}_0 = \mathcal{K}$ that complies with $s_\mathcal{A}$ and $s_\mathcal{D}$ and

1. for all $\mathcal{K}_i \in \pi$ with $P(\mathcal{K}_i) = \mathcal{A}$ it is the case that $\mathcal{K}_{i+1} = \mathcal{K}_i \circ s_\mathcal{A}(\pi[i])$ is safe since $\circ$ satisfies *Acknowledgement*,

2. for all $\mathcal{K}_i \in \pi$ with $P(\mathcal{K}_i) = \mathcal{D}$ it is the case that $\mathcal{K}_{i+1} = \mathcal{K}_i \circ s_\mathcal{D}(\pi[i])$ is safe, since the setting $(\Lambda, \mathfrak{F})$ is plain.

Hence, for all strategies $s_\mathcal{A}$ there is a path $\pi = (\mathcal{K}, \dots, \mathcal{K}_n)$ that is compliant with $s_\mathcal{A}$ and the utility $u_\mathcal{A}(\pi) = 0$ such that $\mathcal{A}$ does not have a winning strategy in $g_{\mathcal{L}_{ES},\mathcal{K},\mathsf{Per},\mathsf{Act},\circ,\circ^a}$ and consequently $\mathcal{K}$ is sound wrt. $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$. □

Proposition 5.4.11. (Page 123) Let $(\mathcal{K}^0, \xi)$ be an agent with $\xi = (\circ, \mathsf{act})$, $\circ : \mathsf{Per} \cup \mathsf{Act} \times \mathcal{L}_{ES} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$. If $\mathcal{K}^0$ is sound wrt. $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$, then if $(\mathcal{K}^0, \xi)$ selects sound actions, i. e., for all $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ and all $p \in \mathsf{Per}$ if there is some $a \in \mathsf{Act}$ such that $(\mathcal{K} \circ p) \circ^a a$ is sound wrt. $(\mathcal{L}_{ES}, ((\mathcal{K} \circ p) \circ^a a), \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$, then

$\mathsf{act}(\mathcal{K} \circ p)$ is such that $\mathcal{K} \circ^a \mathsf{act}(\mathcal{K} \circ p)$ is sound wrt. $(\mathcal{L}_{ES}, \mathcal{K} \circ^a \mathsf{act}(\mathcal{K} \circ p), \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$, then $(\mathcal{K}^0, \xi)$ is secrecy preserving.

Proof. In Proposition 5.4.9 (Page 123) we showed that an agent with initial epistemic state $(\mathcal{K}^0, \xi)$ is secrecy preserving if all states $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ are sound wrt. $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$. What is left to show is that for all $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ for all $p \in \mathsf{Per}$ there exists an action ι such that $(\mathcal{K} \circ p) \circ^a a$ is sound wrt. $(\mathcal{L}_{ES}, (\mathcal{K} \circ^a a), \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$. By assumption $\mathcal{K}^0$ is sound wrt. $(\mathcal{L}_{ES}, \mathcal{K}^0, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$. We show that if an epistemic state $\mathcal{K}$ is sound wrt. $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$, then there exists an action $\iota \in \mathsf{Act}$ such that $\mathcal{K} \circ^a a$ is sound wrt. $(\mathcal{L}_{ES}, (\mathcal{K} \circ^a a), \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$.

If $\mathcal{K}$ is sound wrt. $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a)$, then $\mathcal{A}$ does not have a winning strategy in $g_{\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a}$ which means that for all strategies $s_\mathcal{A}$ there exists a path

$$\pi = (\mathcal{K}, \mathcal{K}_1, \mathcal{K}_2 \ldots, \mathcal{K}_n) \in W$$

that is compliant with $s_\mathcal{A}$ and $u_\mathcal{A}(\pi) = 0$. Hence for each $s_\mathcal{A}$ there exists a path $\pi = (\mathcal{K}, \mathcal{K}_1, \mathcal{K}_2 \ldots, \mathcal{K}_n) \in W$ such that for some $p \in \mathsf{Per}$ it holds $p = s_\mathcal{A}((\mathcal{K}))$ and there exists an action $a \in \mathsf{Act}$ such that $\mathcal{K}_2 = \mathcal{K}_1 \circ^a a$ and $u_\mathcal{A}(\pi) = 0$. The paths of the game $g_{\mathcal{L}_{ES}, \mathcal{K}_2, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a}$ are

$$W' = \{(\mathcal{K}_2 \ldots, \mathcal{K}_n) \mid (\mathcal{K}, \mathcal{K}_1, \mathcal{K}_2 \ldots, \mathcal{K}_n) \in W\},$$

further, if $u_\mathcal{A}((\mathcal{K}, \mathcal{K}_1, \mathcal{K}_2 \ldots, \mathcal{K}_n)) = 0$ then $u_\mathcal{A}((\mathcal{K}_2 \ldots, \mathcal{K}_n)) = 0$. Since, for all strategies $s_\mathcal{A}$ there exists a path

$$\pi = (\mathcal{K}, \mathcal{K}_1, \mathcal{K}_2 \ldots, \mathcal{K}_n) \in W$$

that is compliant with $s_\mathcal{A}$ and $u_\mathcal{A}(\pi) = 0$ also for all strategies $s'_\mathcal{A}$ in $g_{\mathcal{L}_{ES}, \mathcal{K}_2, \mathsf{Per}, \mathsf{Act}, \circ, \circ^a}$ there exists a path π' compliant with $s'_\mathcal{A}$ such that $u_\mathcal{A}(\pi') = 0$. Therefore, $\mathcal{K}_2$ is sound, which was to be shown. □

Observation 5.4.12. (On Page 124) Let $(\Lambda, \mathfrak{F})$ be a setting with $\xi = (\circ, \circ^a, \mathsf{act})$, $\circ : \mathcal{L}_{ES} \times \mathsf{Per} \to \mathcal{L}_{ES}$, $\circ^a : \mathcal{L}_{ES} \times \mathsf{Act} \to \mathcal{L}_{ES}$ and $\mathsf{act} : \mathcal{L}_{ES} \to \mathsf{Act}$. Let $\epsilon \in \mathsf{Act}$ be the empty action. If for all $\mathcal{K} \in \Omega_\xi s(\Lambda, \mathsf{Per})$ and all $p \in \mathsf{Per}$ it holds that $(\mathcal{K} \circ p) \circ^a \epsilon$ is safe, then $(\Lambda, \mathfrak{F})$ is plain.

Proof. From the definition of a plain setting in, see Definition 5.4.2 (Page 120), follows that in each possible state there exists a safe action. Here the empty action is assumed to be always secrecy preserving, which proofs the existence of such an action. Formally the definition is satisfied since for all $(\mathcal{K}, \xi) \in (\Lambda, \mathfrak{F})$ for all safe epistemic states $\mathcal{K} \in \Omega_\xi(\mathcal{K}, \mathsf{Per})$ it holds that for all $p \in \mathsf{Per}$ the epistemic state $(\mathcal{K} \circ p) \circ^a \epsilon$ is safe. □

Lemma 5.4.14. (On Page 126) For each $(\mathcal{K}, \xi) \in \mathcal{T}^{QA}$ it holds that for all $\mathcal{K}' \in \Omega_\xi(\mathcal{K}, \mathsf{Per}^{QA})$ that $\mathcal{S}(\mathcal{K}') = \mathcal{S}(\mathcal{K})$.

Proof. For all $\mathcal{K}' \in \Omega_\xi(\mathcal{K}, \mathsf{Per}^{QA})$, $\mathcal{K}' \neq \mathcal{K}$ it holds by Definition 5.2.6 that

$$\mathcal{K}' = \mathcal{K}'' \circ p \circ^{a} \mathsf{act}(\mathcal{K}'' \circ p)$$

for some $\mathcal{K}'' \in \Omega_\xi(\mathcal{K}, \mathsf{Per}^{QA})$ and $p \in \mathsf{Per}^{QA}$. Hence we use induction to show the lemma.

Apparently for $\mathcal{K}' = \mathcal{K}$ it holds that $\mathcal{S}(\mathcal{K}') = \mathcal{S}(\mathcal{K})$. Now, assume that it holds for $\mathcal{K}'' \in \Omega_\xi(\mathcal{K}, \mathsf{Per}^{QA})$ that $\mathcal{S}(\mathcal{K}'') = \mathcal{S}(\mathcal{K})$. Since $\mathsf{Per}^{QA} = \{\langle \mathcal{A}, \{\mathcal{D}\}, \mathsf{query}, \Phi\rangle \mid \Phi \in \mathcal{L}_{\mathsf{Base}}\}$ and by assumption 3.a) it holds that $t_V(\langle \mathcal{D}, \{\mathcal{A}\}, \mathsf{query}, \Phi\rangle) = \emptyset$ it follows from Equation (5.3.2) (On Page 115) that

$$V(\mathcal{K}'' \circ p) = V(\mathcal{K}'') *_{\mathcal{L}_V} \emptyset.$$

By satisfaction of assumption 4. and 6. it follows that

$$V(\mathcal{K}'') *_{\mathcal{L}_V} \emptyset = V(\mathcal{K}'') \cup \emptyset = V(\mathcal{K}'').$$

Therefore it holds for all $\phi \in F(\mathcal{S}(\mathcal{K}''))$ that $\phi \in Cn^{\mathsf{prop}}(V(\mathcal{K}'))$ if and only if $\phi \in Cn^{\mathsf{prop}}(V(\mathcal{K}''))$. From the satisfaction of *Min-Secrecy-Weakening* by $\circ$ follows that $\mathcal{S}(\mathcal{K}'' \circ p) = \mathcal{S}(\mathcal{K}'')$.

From the satisfaction of *Secrets action invariance* it follows that

$$\mathcal{S}(\mathcal{K}'' \circ p) = \mathcal{S}(\mathcal{K}'' \circ p \circ^{a} \mathsf{act}(\mathcal{K}'' \circ p)),$$

which completes the proof. □

Proposition 5.4.15. (On Page 126) For each $(\mathcal{K}^0, \xi) \in \mathcal{T}^{QA}$ it holds for all $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per}^{QA})$ that
$\mathcal{K}$ is not sound wrt. $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}^{QA}, \mathsf{Act}^{QA}, \circ, \circ^{a})$ if and only if

$$(\bigvee_{\phi \in F(\mathcal{S}(\mathcal{K}))} \phi) \in \mathsf{Bel}(V_{\mathcal{A}}(\mathcal{K})).$$

Proof.
"$\Leftarrow$": Suppose that $(\bigvee_{\phi \in F(\mathcal{S}(\mathcal{K}))} \phi) \in Cn^{\mathsf{prop}}(V_{\mathcal{A}}(\mathcal{K}))$. We show that in this setting it is a winning strategy for $\mathcal{A}$ to query all sensitive formulae in a sequence of queries.

We define a change operator $\odot$ for the change of an epistemic state $\mathcal{K}$ by a sequence of speech acts $\sigma = (\iota_{Q_1}, \iota_{A_1} \ldots \iota_{Q_n}, \iota_{A_n})$ as:

$$\mathcal{K} \odot \sigma = ((\ldots((((\mathcal{K} \circ \iota_{Q_1}) \circ^{a} \iota_{A_1}) \circ \iota_{Q_2}) \circ^{a} \iota_{A_1}) \cdots \circ \iota_{Q_n}) \circ^{a} \iota_{A_n})$$

We denote the sub-sequence of a sequence σ of the first i ι_Q, ι_A pairs by $\sigma[i]$.

We consider a speech act sequence $\sigma_{\mathsf{sec}} = (\iota_{Q_1}, \iota_{A_1} \ldots \iota_{Q_n}, \iota_{A_n})$ of queries of $\mathcal{A}$ and answers of $\mathcal{D}$ with

$$(\iota_{Q_i}, \iota_{A_i}) \in \{(\langle \mathcal{A}, \{\mathcal{D}\}, \mathsf{query}, \phi_i\rangle, \langle \mathcal{D}, \{\mathcal{A}\}, \mathsf{answer}, \phi_i'\rangle) \mid \\ \phi_i \in F(\mathcal{S}(\mathcal{K})), \phi_i' \in \{\phi_i, \neg\phi_i\}\}$$

such that the following two conditions are satisfied:

a) $\{\phi_1, \dots, \phi_n\} = \mathsf{F}(\mathcal{S}(\mathcal{K}))$,

b) for all epistemic states $\mathcal{K}'$ of the path generated by the action sequence

$$\pi_{\sigma_{\mathsf{sec}}} =_{def} (\mathcal{K} \odot \sigma_{\mathsf{sec}}[0], \mathcal{K} \odot \sigma_{\mathsf{sec}}[1], \dots, \mathcal{K} \odot \sigma_{\mathsf{sec}}[n])$$

it holds that $V(\mathcal{K}')$ is consistent.

We show that in this setting the strategy of $\mathcal{A}$ that always asks the queries of the given sequence in the given order is a winning strategy in the game $\mathcal{G}_{\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}^{QA}, \mathsf{Act}^{QA}, \circ, \circ^a}$. Formally this strategy can be defined as $s_a((\mathcal{K}''_1, \dots, \mathcal{K}''_m)) = \iota_{Q_j}$ with $j = (\frac{m+1}{2}) \mod n$.

For this it suffices to show that the epistemic state

$\mathcal{K} \odot \sigma_{\mathsf{sec}}$ is not safe

since we use an arbitrary sequence σ_{sec} of the form defined above.

By condition b) it holds for all $\mathcal{K} = \mathcal{K}_{i-1} \circ^a \iota_{A_i}$ with $\mathcal{K}_{i-1} \in \pi_{\sigma_{\mathsf{sec}}}$ and $\langle \mathcal{D}, \{\mathcal{A}\}, \mathsf{answer}, \phi'_i \rangle \in \sigma_{\mathsf{sec}}$ that $V(\mathcal{K})$ is consistent. By Equation (5.3.2) (On Page 115)

$$V(\mathcal{K}_{i-1}) \circ \langle \mathcal{D}, \{\mathcal{A}\}, answer, \phi'_i \rangle = V(\mathcal{K}_{i-1}) *_{\mathcal{L}_V} t_V(\mathcal{A}, \langle \mathcal{D}, \{\mathcal{A}\}, \mathsf{answer}, \phi'_i \rangle).$$

By Assumption 3.b) (On Page 125) we get that

$$t_V(\mathcal{A}, \langle \mathcal{D}, \{\mathcal{A}\}, \mathsf{answer}, \phi'_i \rangle) = \phi'_i$$

and by Assumption 4. (On Page 125) and the fact that $V(\mathcal{K}_{i-1}) \cup \{\phi'_i\}$ is consistent (by condition b)) it holds that

$$V(\mathcal{K}_{i-1}) *_{\mathcal{L}_V} \phi'_i = V(\mathcal{K}_{i-1}) \cup \{\phi'_i\}.$$

Therefore it holds that $\phi'_i \in V(\mathcal{K}_{i-1})$. By condition a) it follows that $V(\mathcal{K} \odot \sigma_{\mathsf{sec}})$ contains for all $\phi \in \mathsf{F}(\mathcal{S}(\mathcal{K}))$ either ϕ or $\neg\phi$. By assumption

$$(\bigvee_{\phi \in \mathsf{F}(\mathcal{S}(\mathcal{K}))} \phi) \in V(\mathcal{K} \odot \sigma_{\mathsf{sec}}).$$

Further, by Assumption 6. it holds that it follows that

$V(\mathcal{K} \odot \sigma_{\mathsf{sec}})$ is consistent.

From the last three statements follows that for some $\phi \in \mathsf{F}(\mathcal{S}(\mathcal{K}))$ it holds that

$$\phi \in Cn^{\mathsf{prop}}(V(\mathcal{K} \odot \sigma_{\mathsf{sec}})).$$

From Lemma 5.4.14 (On Page 126) follows that

$$\mathcal{S}(\mathcal{K} \odot \sigma_{\mathsf{sec}}) = \mathcal{S}(\mathcal{K}).$$

Hence $\mathcal{K} \odot \sigma_{\text{sec}}$ is not safe.

"$\Rightarrow$": Suppose $\mathcal{K}$ is not sound wrt. $(\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}^{QA}, \mathsf{Act}^{QA}, \circ, \circ^{a})$.

From the assumption it follows by definition that there is a winning strategy for $\mathcal{A}$. Let this be $s_{\mathcal{A}}$. We consider the finite sub-tree T' of the game tree of $g_{\mathcal{L}_{ES}, \mathcal{K}, \mathsf{Per}^{QA}, \mathsf{Act}^{QA}, \circ, \circ^{a}}$ that contains exactly the paths that are compliant with $s_{\mathcal{A}}$, truncated after the first node that is not safe. Hence all leaf nodes are not safe and all inner nodes are safe. We show by structural induction from the leafs to the root that for all nodes $\mathcal{K}'$ of T' it holds that $(\bigvee_{\phi \in F(\mathcal{S}(\mathcal{K}))} \phi) \in Cn^{\mathsf{prop}}(V(\mathcal{K}'))$.

BASIS: LEAF NODES Let $\mathcal{K}'$ be a leaf node. Since $\mathcal{K}'$ is not safe it holds for some $\phi \in F(\mathcal{S}(\mathcal{K}'))$ that $\phi \in Cn^{\mathsf{prop}}(V(\mathcal{K}'))$. By Lemma 5.4.14 (On Page 126) it holds that $\mathcal{S}(\mathcal{K}') = \mathcal{S}(\mathcal{K})$. If follows that

$$(\bigvee_{\phi \in F(\mathcal{S}(\mathcal{K}))} \phi) \in Cn^{\mathsf{prop}}(V(\mathcal{K}')).$$

INDUCTIVE STEP: Let $\mathcal{K}'$ be an inner node. The children of $\mathcal{K}'$ are denoted by $ch(\mathcal{K}')$. By induction assumption it holds for all $\mathcal{K}'' \in ch(\mathcal{K}')$ that $(\bigvee_{\phi \in F(\mathcal{S}(\mathcal{K}))} \phi) \in Cn^{\mathsf{prop}}(V(\mathcal{K}''))$. We have to distinguish if we are in a node in which $\mathcal{A}$ or $\mathcal{D}$ is acting. This is indicated by the player function P as defined in Equation (5.4.1) (Page 121).

CASE 1: $P(\mathcal{K}') = \mathcal{A}$ It holds that $\mathcal{K}'' = \mathcal{K}' \circ \langle \mathcal{A}, \{\mathcal{D}\}, \mathsf{query}, \phi \rangle$. By Equation (5.3.2) (On Page 115) it is

$$\begin{aligned} &V(\mathcal{K}') \circ \langle \mathcal{A}, \{\mathcal{D}\}, \mathsf{query}, \phi \rangle = \\ &\qquad V(\mathcal{K}') *_{\mathcal{L}_V} t_V(\mathcal{A}, \langle \mathcal{A}, \{\mathcal{D}\}, \mathsf{query}, \phi \rangle). \end{aligned}$$

By assumption 3.a) $t_V(\mathcal{A}, \langle \mathcal{A}, \{\mathcal{D}\}, \mathsf{query}, \phi \rangle) = \emptyset$. By assumption 4. $*_{\mathcal{L}_V}$ satisfies *Inclusion* and *Vacuity* such that, since $V(\mathcal{K})$ is consistent by condition (2), it holds that

$$V(\mathcal{K}'') = V(\mathcal{K}') *_{\mathcal{L}_V} \emptyset = V(\mathcal{K}') \cup \emptyset.$$

CASE 2: $P(\mathcal{K}') = \mathcal{D}$ It holds that $\mathcal{K}'' = \mathcal{K}' \circ \langle \mathcal{D}, \{\mathcal{A}\}, \mathsf{answer}, \phi' \rangle$. By Equation (5.3.2)

$$\begin{aligned} &V(\mathcal{K}) \circ \langle \mathcal{D}, \{\mathcal{A}\}, \mathsf{answer}, \phi' \rangle = \\ &\qquad\qquad V(\mathcal{K}) *_{\mathcal{L}_V} t_V(\mathcal{A}, \langle \mathcal{D}, \{\mathcal{A}\}, \mathsf{answer}, \phi' \rangle). \end{aligned}$$

By assumption 3.b) $t_V(\mathcal{A}, \langle \mathcal{D}, \{\mathcal{A}\}, \mathsf{answer}, \phi' \rangle) = \{\phi'\}$. By assumption 4. $*_{\mathcal{L}_V}$ satisfies *Inclusion* and *Vacuity* such that, since $V(\mathcal{K}') \cup \{\phi'\}$ is consistent by condition b), it holds that

$$V(\mathcal{K}') *_{\mathcal{L}_V} \phi' = V(\mathcal{K}') \cup \{\phi'\}.$$

By assumption 5. the children of $\mathcal{K}'$ result from $\mathcal{D}$'s two possible answer-values. Assume that these are $\{\phi, \neg\phi\}$. Then, taking assumption 6. into consideration, $V(\mathcal{K}') \cup \{\phi\}$ is a child of $\mathcal{K}'$ if it is consistent, and $V(\mathcal{K}') \cup \{\neg\phi\}$ is a child of $\mathcal{K}'$ if it is consistent. Now we consider two cases:

1. *$\mathcal{K}'$ has two children:*
   Then, there is another child $\mathcal{K}''' \in ch(\mathcal{K}')$, $\mathcal{K}''' \neq \mathcal{K}''$ with $\mathcal{K}''' = \mathcal{K}' \circ \langle \mathcal{D}, \{\mathcal{A}\}, \mathsf{answer}, \neg\phi' \rangle$. By the same argumentation as for $\mathcal{K}''$ it follows that $V(\mathcal{K}') *_{\mathcal{L}_V} \neg\phi' = V(\mathcal{K}') \cup \{\neg\phi'\}$. Since the induction assumption holds for both children if follows that
   $$\begin{aligned}(\textstyle\bigvee_{\phi \in F(\mathcal{S}(\mathcal{K}))} \phi) \in Cn^{\mathsf{prop}}(V(\mathcal{K}') \cup \{\phi'\}) \\ \cap\, Cn^{\mathsf{prop}}(V(\mathcal{K}') \cup \{\neg\phi'\}) \\ = Cn^{\mathsf{prop}}(V(\mathcal{K}')).\end{aligned}$$

2. *$\mathcal{K}'$ has one child:*
   There is no other child, from which follows that
   $$V(\mathcal{K}') *_{\mathcal{L}_V} \neg\phi' = V(\mathcal{K}') \cup \{\neg\phi'\}$$
   is inconsistent. By definition of Cn^{prop} it follows that
   $$\phi' \in Cn^{\mathsf{prop}}(V(\mathcal{K}')).$$
   Consequently we have that
   $$Cn^{\mathsf{prop}}(V(\mathcal{K}')) = Cn^{\mathsf{prop}}(V(\mathcal{K}') \cup \{\phi'\}).$$
   Since by induction assumption
   $$\bigvee_{\phi \in F(\mathcal{S}(\mathcal{K}'))} \phi \in Cn^{\mathsf{prop}}(V(\mathcal{K}') \cup \{\phi'\})$$
   it follows directly that
   $$\bigvee_{\phi \in F(\mathcal{S}(\mathcal{K}'))} \phi \in Cn^{\mathsf{prop}}(V(\mathcal{K}')).$$

□

Corollary 5.4.17. (On Page 127) Let $\mathcal{T}^{QA}_{\bigvee} \subseteq \mathcal{T}^{QA}$ be the setting for which it holds that for all $(\mathcal{K}, \xi) \in \mathcal{T}^{QA}_{\bigvee}$ there is some $\phi \in F(\mathcal{S}(\mathcal{K}))$ such that $\phi \equiv^{p} (\bigvee_{\phi \in F(\mathcal{S}(\mathcal{K}))} \phi)$. The setting $\mathcal{T}^{QA}_{\bigvee}$ is *plain*.

Proof. We have to show that for all agents $(\mathcal{K}^0, \xi) \in \mathcal{T}^{QA}_{\bigvee}$ for all safe epistemic states $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}^0, \mathsf{Per}^{QA})$ it holds that for all $p \in \mathsf{Per}^{QA}$ there is some action $a \in \mathsf{Act}^{QA}$ such that $(\mathcal{K} \circ p) \circ^{a} a$ is safe.

Let $(\mathcal{K}^0, \xi) \in \mathcal{T}^{QA}_{\bigvee}$ and $\mathcal{K} \in \Omega_{\xi}(\mathcal{K}^0, \mathsf{Per}^{QA})$ from Proposition 5.4.15 follows directly that if $\mathcal{K}$ is safe, then it is sound. By definition of a sound epistemic state (Definition 5.4.7, Page 122) follows that $\mathcal{A}$ does not have a winning strategy in the game $\mathcal{G}_{\mathcal{L}_{ES},\mathcal{K},\mathsf{Per}^{QA},\mathsf{Act}^{QA},\circ,\circ^a}$.

Assume to the contrary that for some $p \in \mathsf{Per}^{QA}$, which represents a move of $\mathcal{A}$, for all actions $a \in \mathsf{Act}^{QA}$, moves of $\mathcal{D}$, $(\mathcal{K} \circ p) \circ^a a$ is unsafe. Then, any strategy of the attacker $s_{\mathcal{A}}$ such that $s_{\mathcal{A}}(\mathcal{K}) = p$ is a winning strategy. □

Proposition 5.5.2. (On Page 133) Let $(\Lambda, \mathfrak{F})$ be a setting. An agent with initial agent state $(\mathcal{K}^0, \xi) \in (\Lambda, \mathfrak{F})$ is secrecy preserving if for all $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ and all $p \in \mathsf{Per}$:

a) $\preceq_{\mathcal{K} \circ p}$ satisfies sound preference

b) if there is a sound action $a \in \mathsf{Act}$, then there is a sound action $a' \in \mathsf{Act}'_{(\mathcal{K},\xi)}$

Proof. By construction of Ω all states $\mathcal{K} \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$, except for $(\mathcal{K}^0, \xi)$, are successors of some other state $\mathcal{K}' \in \Omega_\xi(\mathcal{K}^0, \mathsf{Per})$ such that $\mathcal{K} = \mathcal{K}' \circ p \circ^a \mathsf{act}(\mathcal{K}' \circ p)$. We proof the result by induction over this successor relation. The initial agent state $(\mathcal{K}^0, \xi)$ is sound by assumption. That is, it is left to show that if $\mathcal{K}'$ is sound, then $\mathcal{K}' \circ p \circ^a \mathsf{act}(\mathcal{K}' \circ p)$ is sound.

If $\mathcal{K}'$ is sound, then by definition for all $p \in \mathsf{Per}$ there is an action $a \in \mathsf{Act}$ such that a is sound. It follows from ii) that some $a' \in \mathsf{Act}'_{(\mathcal{K},\xi)}$ is sound. Then it, follows from i) that $\mathsf{act}(\mathcal{K}' \circ p)$ is sound. □

Proposition 5.5.12. (On Page 146) Any function $\mathsf{sr} : \mathcal{P}_{\mathsf{fin}}(\mathsf{act}) \times \Omega \to \mathcal{C}l$ that satisfies Principles I.1, I.2 and II also satisfies Principle III and Principle IV.

Proof. Assume a secrecy reasoner sr which satisfies Principle I.1, Principle I.2 and Principle II. Let Act'_{inf} be a finite set of actions and $\mathcal{K}$ an epistemic state. Let $\mathsf{cl} = \mathsf{sr}(\mathsf{Act}'_{inf}, \mathcal{K})$ be the classification determined by sr.

We first prove the following lemma.

Lemma A.1.3. If there exists an action $b \in \mathsf{Act}'_{inf}$ with $\mathsf{vioAfter}(\mathcal{K}, b) = \{\emptyset\}$, then for all actions $a \in \mathsf{Act}'_{inf}$ it holds $\mathsf{cl}(a) = 0$ if and only if $\mathsf{vioAfter}(\mathcal{K}, a) = \{\emptyset\}$.

Proof. "$\Leftarrow$": If $\mathsf{vioAfter}(\mathcal{K}, a) = \{\emptyset\}$, then $\mathsf{cl}(a) = 0$:
Let $a \in \mathsf{Act}'_{inf}$ be an action with $\mathsf{vioAfter}(\mathcal{K}, a) = \{\emptyset\}$.

1. By the definitions of $\mathsf{vioAfter}$ and $\sqsupset$, there cannot exist an action $a' \in \mathsf{Act}'_{inf}$ such that $\mathsf{vioAfter}(\mathcal{K}, a') \sqsubset \mathsf{vioAfter}(\mathcal{K}, a)$ holds.

2. Hence, for no $a' \in \mathsf{Act}'_{inf}$ the precondition of Principle I.1 is satisfied. Therefore, there does not exist an action $a' \in \mathsf{Act}'_{inf}$ such that $\mathsf{cl}(a') < \mathsf{cl}(a)$ is demanded by this principle.

3. There are no secret $(\phi, \mathsf{Bel}) \in \mathcal{S}(\mathcal{K})$ and no belief operator $\mathsf{Bel}' \in \Xi$ with $\mathsf{Bel}' \prec_{\mathsf{bel}} \mathsf{Bel}$ such that the secret (ϕ, Bel') is potentially violated after action a.
   The reason is that, if there were a potential violation of (ϕ, Bel') after a, then by definition there exists $V_W \in V(\mathcal{K}) \oplus a$ such that $\phi \in \mathsf{Bel}'(V_W)$. Since $\mathsf{Bel}' \prec_{\mathsf{bel}} \mathsf{Bel}$ holds, the credulity property of the order $\preceq_{\mathsf{bel}}$ implies that $\mathsf{Bel}(V_W) \supseteq \mathsf{Bel}'(V_W) \ni \phi$. Hence, it holds $(\phi, \mathsf{Bel}) \in \mathsf{vio}(\mathcal{K}, V_W)$ and $\mathsf{vio}(\mathcal{K}, V_W) \in \mathsf{vioAfter}(\mathcal{K}, a)$ by Definition 5.5.4. This cannot happen since $\mathsf{vioAfter}(\mathcal{K}, a) = \{\emptyset\}$ holds.

4. Therefore, for each action $a' \in \mathsf{Act}'_{inf}$ the precondition of Principle I.2 cannot be satisfied so that for each action $a' \in \mathsf{Act}'_{inf}$ the relation $a' \prec_{vio} a$ does hold. Hence, for no $a' \in \mathsf{Act}'_{inf}$ $\mathsf{cl}(a') < \mathsf{cl}(a)$ is demanded by this Principle I.2.

5. By Points 2 and 4, there exists a classification cl' with $\mathsf{cl}'(a) = 0$ which satisfies Principles I.1 and I.2.

6. From the assumed satisfaction of Principle II by the secrecy reasoner sr, it outputs a classification cl such that $\mathsf{cl}(a) = 0$.

"$\Rightarrow$" If $\mathsf{cl}(a) = 0$, then $\mathsf{vioAfter}(\mathcal{K}, a) = \{\emptyset\}$:
We show the implication by contraposition. Let $a \in \mathsf{Act}'_{inf}$ be an action such that $\mathsf{vioAfter}(\mathcal{K}, a) \neq \{\emptyset\}$ holds.

1. By the assumption of Lemma A.1.3 there exists $b \in \mathsf{Act}'_{inf}$ with $\mathsf{vioAfter}(\mathcal{K}, b) = \{\emptyset\}$.

2. Hence, it holds $\mathsf{vioAfter}(\mathcal{K}, a) \sqsubset \mathsf{vioAfter}(\mathcal{K}, b)$ by definition.

3. By Principle I.1, it follows that $\mathsf{cl}(a) > \mathsf{cl}(b) \geqslant 0$.

$\square$

SATISFACTION OF PRINCIPLE III
Let $\mathcal{K}$ and $\mathcal{K}'$ be epistemic states with equal components possibly except for the secrecy policies which are of the following form:

$$\begin{aligned} & \mathcal{S}(\mathcal{K}) = \{(\phi_1, \mathsf{Bel}_1), \ldots (\phi_n, \mathsf{Bel}_n)\}, \\ & \mathcal{S}(\mathcal{K}') = \{(\phi_1, \mathsf{Bel}_1{}'), \ldots (\phi_n, \mathsf{Bel}_n{}')\} \\ \text{with} \quad & \mathsf{Bel}_i{}' \preceq_{\mathsf{bel}} \mathsf{Bel}_i \text{ for all } i \in \{1, \ldots, n\}. \end{aligned}$$

Assume there exist actions $a, b \in \mathsf{Act}'_{inf}$ such that $\mathsf{vioAfter}(\mathcal{K}, a) = \{\emptyset\}$ and $\mathsf{vioAfter}(\mathcal{K}', b) = \{\emptyset\}$.
We show that for all actions $c \in \mathsf{Act}'_{inf}$, if $\mathsf{cl}(c) = 0$ holds, then $\mathsf{cl}'(c) = 0$ holds with $\mathsf{cl} = \mathsf{sr}(\mathsf{Act}'_{inf}, \mathcal{K})$ and $\mathsf{cl}' = \mathsf{sr}(\mathsf{Act}'_{inf}, \mathcal{K}')$.
Let c be an action $c \in \mathsf{Act}'_{inf}$ such that $\mathsf{cl}(c) = 0$.

1. Since by assumption action $a \in \mathsf{Act}'_{inf}$ satisfies $\mathsf{vioAfter}(\mathcal{K}, a) = \{\emptyset\}$ it holds that $\mathsf{vioAfter}(\mathcal{K}, c) = \{\emptyset\}$ by Lemma A.1.3.

2. From the definition of vioAfter it follows directly that for all $V_W \in \oplus(V(\mathcal{K}), c)$ and for all $(\phi, \mathsf{Bel}) \in \mathcal{S}(\mathcal{K})$ it holds that $\phi \notin \mathsf{Bel}(V_W)$.

3. By presupposition, the secrecy policies are of the form

$$\begin{aligned} & \mathcal{S}(\mathcal{K}) = \{(\phi_1, \mathsf{Bel}_1), \dots (\phi_n, \mathsf{Bel}_n)\}, \\ & \mathcal{S}(\mathcal{K}') = \{(\phi_1, \mathsf{Bel}_1{}'), \dots (\phi_n, \mathsf{Bel}_n{}')\} \\ \text{with} \quad & \mathsf{Bel}_i{}' \preceq_{\mathsf{bel}} \mathsf{Bel}_i \text{ for all } i \in \{1, \dots, n\}. \end{aligned}$$

   so that for all $V_W \in \mathcal{P}(\mathcal{L}_{BB}) \times \mathcal{L}_{BS}{}^*$ it holds

$$\mathsf{Bel}'_i(V_W) \subseteq \mathsf{Bel}_i(V_W)$$

   by the credulity property of the order $\preceq_{\mathsf{bel}}$. Hence, it follows that for all $V_W \in \oplus(V(\mathcal{K}), c)$ and for all $(\phi, \mathsf{Bel}') \in \mathcal{S}(\mathcal{K}')$ it holds that $\phi \notin \mathsf{Bel}'(V_W)$ by Point 2.

4. It follows that

$$\{\emptyset\} = \mathsf{vioAfter}(\mathcal{S}(\mathcal{K}'), V(\mathcal{K}), c) = \mathsf{vioAfter}(\mathcal{S}(\mathcal{K}'), V(\mathcal{K}'), c).$$

5. Since by assumption, action $b \in \mathsf{Act}'_{inf}$ satisfies that

$$\mathsf{vioAfter}(\mathcal{K}', b) = \{\emptyset\}$$

   it follows that $\mathsf{cl}'(c) = 0$ by Lemma A.1.3. □

SATISFACTION OF PRINCIPLE IV

Let $\mathcal{K}$ and $\mathcal{K}'$ be epistemic states with equal components possibly except for $V_{\mathcal{A}}(\mathcal{K}) \supseteq V_{\mathcal{A}}(\mathcal{K}')$. Assume there exist actions $a, b \in \mathsf{Act}'_{inf}$ such that $\mathsf{vioAfter}(\mathcal{K}, a) = \{\emptyset\}$ and $\mathsf{vioAfter}(\mathcal{K}', b) = \{\emptyset\}$.
We show that for each $c \in \mathsf{Act}'_{inf}$ if $\mathsf{cl}(c) = 0$ holds, then $\mathsf{cl}'(c) = 0$ holds with $\mathsf{cl} = \mathsf{sr}(\mathsf{Act}'_{inf}, \mathcal{K})$ and $\mathsf{cl}' = \mathsf{sr}(\mathsf{Act}'_{inf}, \mathcal{K}')$. Let c be an action $c \in \mathsf{Act}'_{inf}$ such that $\mathsf{cl}(c) = 0$.

1. Since $a \in \mathsf{Act}'_{inf}$ satisfies $\mathsf{vioAfter}(\mathcal{K}, a) = \{\emptyset\}$ it holds that $\mathsf{vioAfter}(\mathcal{K}, c) = \{\emptyset\}$ by Lemma A.1.3.

2. From the definition of vioAfter it follows directly that for $V(\mathcal{K}) \supseteq V(\mathcal{K}')$ it holds that

$$\begin{aligned} \mathsf{vioAfter}(\mathcal{S}(\mathcal{K}), V(\mathcal{K}), c) \supseteq \mathsf{vioAfter}(\mathcal{S}(\mathcal{K}), V(\mathcal{K}'), c) = \\ \mathsf{vioAfter}(\mathcal{S}(\mathcal{K}'), V(\mathcal{K}'), c). \end{aligned}$$

3. It follows that $\mathsf{vioAfter}(\mathcal{S}, V(\mathcal{K}'), c) = \{\emptyset\}$.

4. Since action $b \in \mathsf{Act}'_{inf}$ satisfies $\mathsf{vioAfter}(\mathcal{K}', b) = \{\emptyset\}$ it follows that $\mathsf{cl}'(c) = 0$ by Lemma A.1.3.

$\square$

Lemma 5.5.13. (On Page 147) Given some set of actions A and an epistemic state $\mathcal{K}$. The set $\mathsf{conflictSets}(A, \mathcal{K})$ is a partition of A, i. e., it holds that

a) $\bigcup \mathsf{conflictSets}(A, \mathcal{K}) = A$ and

b) for all $CS, CS' \in \mathsf{conflictSets}(A, \mathcal{K})$, $CS \neq CS'$ it holds that $CS \cap CS' = \emptyset$.

Proof. Let A be some set of actions, $\mathcal{K}$ an epistemic state and $\mathsf{conflictSets}(A, \mathcal{K})$ the set of conflict sets for A and $\mathcal{K}$.

a) For each singleton set $A' = \{a\}$ with $a \in A$ it holds trivially that

$$\begin{aligned} &\text{for all } a, b \in A',\ a \neq b \\ &\quad \text{exist } a_1, \dots, a_n \in A' \text{ with } a_1 = a,\ a_n = b \text{ and} \\ &\qquad \text{for all } i \in \{1, \dots, n-1\}\text{: } a_i \prec_{vio} a_{i+1}. \end{aligned} \tag{A.1.1}$$

Since $\mathsf{conflictSets}(A, \mathcal{K})$ is a set-inclusion maximal set of sets A' satisfying (A.1.1) it follows that for all $a \in A$ that $a \in CS$ for some $CS \in \mathsf{conflictSets}(A)$. Thus $\bigcup \mathsf{conflictSets}(A, \mathcal{K}) = A$.

b) Assume to the contrary that there exist

$$CS, CS' \in \mathsf{conflictSets}(A, \mathcal{K}), CS \neq CS' \text{ such that } CS \cap CS' \neq \emptyset.$$

For all $a \in CS \cap CS'$ and all $b \in CS \cup CS'$ $a \neq b$ exist

$$a_1, \dots, a_n \in A' \text{ with } a_1 = a,\ a_n = b$$

and for all $i \in \{1, \dots, n-1\}$: $a_i \prec_{vio} a_{i+1}$., and therefore for $CS \cup CS' = A'$ (A.1.1) is satisfied. Therefore CS and CS' are not set-inclusion maximal subsets of A with respect to property (A.1.1), in contradiction to the assumption that $CS, CS' \in \mathsf{conflictSets}(A, \mathcal{K})$.

$\square$

Proposition 5.5.15. (On Page 150) If all elementary operations of Procedure 5.5.1 are computable, the algorithm always terminates and returns a complete classification.

Proof.
Termination:
We show that all loops terminate after a finite number of steps.

1. The for loop from Line 2 to Line 4 iterates over the set of considered actions Act'_{inf}, which is finite by definition.

2. The repeat-until loop from Line 5 to Line 26 has the termination condition $\mathsf{unclass} = \emptyset$. Before the loop, the set $\mathsf{unclass}$ is initialized in Line 1 with the finite set of considered actions Act'_{inf}. Within the loop, $\mathsf{unclass}$ is only modified in Line 25, by setting $\mathsf{unclass} := \mathsf{unclass} \setminus \mathsf{best}$.

   We show that always in Line 25 if $\mathsf{unclass} \neq \emptyset$, then $\mathsf{unclass} \cap \mathsf{best} \neq \emptyset$. The set best is defined in Line 6 and not modified in any other line. Apparently it holds that $\mathsf{best} \subseteq \mathsf{unclass}$. It is left to show that if $\mathsf{unclass} \neq \emptyset$, then $\mathsf{best} \neq \emptyset$.

   The set best is non-empty if and only if there exists $a \in \mathsf{unclass}$ such that there does not exists $b \in \mathsf{unclass}$, and $\mathsf{vioAfter}(\mathcal{K}, a) \sqsupset \mathsf{vioAfter}(\mathcal{K}, b)$. We show that this is the case for all possible sets $\mathsf{unclass}$.

   a) Assume to the contrary that for all $a \in \mathsf{unclass}$ there exists $b \in \mathsf{unclass}$ such that $\mathsf{vioAfter}(\mathcal{K}, a) \sqsupset \mathsf{vioAfter}(\mathcal{K}, b)$. This is only possible if there is a cycle with respect to $\sqsupset$, i.e., there exists a sequence $(a_1, \dots, a_n)$ with $\{a_1, \dots, a_n\} = \mathsf{unclass}$ such that

   $$\mathsf{vioAfter}(\mathcal{K}, a_1) \sqsupset \cdots \sqsupset \mathsf{vioAfter}(\mathcal{K}, a_n) \sqsupset \mathsf{vioAfter}(\mathcal{K}, a_1).$$

   We show that this is impossible.

   b) For any pair of actions a_1, $a_2 \in \mathsf{unclass}$ such that

   $$\mathsf{vioAfter}(\mathcal{K}, a_1) \sqsupset \mathsf{vioAfter}(\mathcal{K}, a_2)$$

   it holds by Equation (5.5.11) Condition 2. (On Page 141) that there exists some

   $$S_{a_2} \in \max_{\subseteq} \mathsf{vioAfter}(\mathcal{K}, a_2) \text{ and some } S_{a_1} \in \mathsf{vioAfter}(\mathcal{K}, a_1)$$

   such that $S_{a_2} \subset S_{a_1}$. This holds also for the first two elements of our considered sequence.

   c) By Equation (5.5.11) Condition 1 (On Page 141) it follows for our considered sequence that for each pair a_i, a_{i+1}, $1 \leqslant i < n$, it holds that for all $S_{a_{i+1}} \in \mathsf{vioAfter}(\mathcal{K}, a_{i+1})$ there exists $S_{a_i} \in \mathsf{vioAfter}(\mathcal{K}, a_i)$ such that $S_{a_{i+1}} \subseteq S_{a_i}$ and for all $S_{a_1} \in \mathsf{vioAfter}(\mathcal{K}, a_1)$ there exists $S_{a_n} \in \mathsf{vioAfter}(\mathcal{K}, a_n)$ such that $S_{a_1} \subseteq S_{a_n}$. By the transitivity of the $\subseteq$ relation it follows that for a_2 it holds that for all $S_{a_1} \in \mathsf{vioAfter}(\mathcal{K}, a_1)$ there exists $S_{a_2} \in \mathsf{vioAfter}(\mathcal{K}, a_2)$ such that $S_{a_1} \subseteq S_{a_2}$.

   d) From Points 2b and 2c follows that there exists some $S_{a_2} \in \max_{\subseteq} \mathsf{vioAfter}(\mathcal{K}, a_2)$ and some $S'_{a_2} \in \mathsf{vioAfter}(\mathcal{K}, a_2)$ such that $S_{a_2} \subset S_{a_1} \subseteq S_{a_2}$, which cannot be the case since S_{a_2} is set-inclusion maximal.

3. The for loop from Line 10 to Line 10 is a for loop over the set of equivalence classes of the set $\mathsf{best} \subseteq \mathsf{unclass}$. Since $\mathsf{unclass}$ is a finite set, the set of equivalence classes of a subset of it is finite as well.

4. The for loop from Line 11 to Line 24 is a for loop over the set of equivalence classes of the set $\mathsf{best} \subseteq \mathsf{unclass}$. Since $\mathsf{unclass}$ is a finite set, the set of equivalence classes of a subset of it is finite as well.

5. The repeat-until loop from Line 13 to Line 23 terminates if $\mathsf{conflictSets} = \emptyset$. The set $\mathsf{conflictSets}$ is initialized before the loop in Line 12 by $\mathsf{conflictSets}(A, \mathcal{K})$. In Line 21 it is set to

   $$\mathsf{conflictSets} := \mathsf{conflictSets} \setminus \mathsf{classSets}.$$

   We have to show that if $\mathsf{conflictSets} \neq \emptyset$, then

   $$\mathsf{classSets} \cap \mathsf{conflictSets} \neq \emptyset$$

   in Line 21. The set $\mathsf{classSets}$ is initialized by the empty set in Line 14 and only modified in Line 18 in which a conflict set from the set $\mathsf{conflictSets}$ is added to it. Hence, $\mathsf{classSets} \subseteq \mathsf{conflictSets}$ such that it is left to show that $\mathsf{classSets} \neq \emptyset$.

   a) If $\mathsf{conflictSets} \neq \emptyset$ in Line 15, then the for loop in the same line iterates over all its elements.

   b) We show that for any set $\mathsf{conflictSets}$ there is at least one $CS \in \mathsf{conflictSets}$ that satisfies the condition in Line 16 such that $\mathsf{classSets}$ is never empty in Line 21.

   c) Consider to the contrary that for some $\mathsf{conflictSets}$ it holds that for all CS there is some $CS' \in \mathsf{conflictSets}$ with $CS' \neq CS$ such that $a' \in CS'$ and $a \in CS$ exist with $a' \prec_{vio} a$.

   d) Then, there is a sequence of conflict sets $(CS_1, \ldots, CS_n) \subseteq \mathsf{conflictSets}$ and $CS_n = CS_1$ such that for each CS_i, CS_{i+1}, $1 \leqslant i < n$ there exist $a_i \in CS_i$ and $a'_{i+1} \in CS_{i+1}$ with $a_i \prec_{vio} a'_{i+1}$.

   e) By definition of conflict sets for each pair a_i, a'_i there exists a sequence $(a_{i1}, \ldots, a_{ik_i})$ with $a_{i1} = a_i$, $a_{ik_i} = a'_i$ and for $1 \leqslant j < k_i$ it holds that $a_{ij} \prec_{vio} a_{i(j+1)}$.

   f) Hence, the set

   $$A' = \{a_{i1}, \ldots, a_{ik_i}; \ldots; a_{n1}, \ldots, a_{nk_n}\} \subseteq CS_1 \cup \cdots \cup CS_n$$

   satisfies Condition A.1.1.

   g) The set $\mathsf{conflictSets}$ is the set of set-inclusion maximal sets A' that satisfy A.1.1. Since $CS_1 \subset CS_1 \cup \cdots \cup CS_n$ the set CS_1 cannot be set-inclusion maximal such that it follows that $CS_1 \notin \mathsf{conflictSets}$, in contradiction to the assumption.

6. The for loop from Line 15 to Line 20 iterates over the set of conflict sets which is, as shown in Lemma 5.5.13, a partition of an equivalence class of actions. Since the set of actions is finite, the set of conflict sets is finite.

Complete Classification:
It is left to show that the algorithm returns a complete classification. The set of unclassified actions $\mathsf{unclass}$ is initialized in Line 1, $\mathsf{unclass} := \mathsf{Act}'_{inf}$. The set is only modified in Line 25, where the set best is subtracted from it, i. e., $\mathsf{unclass} := \mathsf{unclass} \setminus \mathsf{best}$. The algorithm terminates only if $\mathsf{unclass} = \emptyset$, Line 26. Therefore it suffices to show that always in Line 25 all actions in best are classified.

1. The set best is initialized in Line 6 and not altered before it is subtracted from $\mathsf{unclass}$ in Line 25.
2. The set best partitioned into the set $\sim$ equivalence classes eqbest in Line 7.
3. Each of these equivalence classes is partitioned, as shown in Lemma 5.5.13, into their conflict sets $\mathsf{conflictSets}$ in Line 12. The set $\mathsf{conflictSets}$ is only modified in Line 21 where the set $\mathsf{classSets}$ is subtracted.
4. The set $\mathsf{classSets}$ is initialized in Line 14 by the empty set and only modified in Line 18 by adding the set CS. All actions in CS have been classified in Line 17. Hence, in Line 21 only sets of classified actions are removed from $\mathsf{conflictSets}$.
5. The repeat-until loop from Line 13 to 23 only terminates if $\mathsf{conflictSets} = \emptyset$, which implies, since $\mathsf{conflictSets}$ is a partition of A, that all actions in A are classified. This is repeated for all $A \in \mathsf{eqbest}$ by the for loop from Line 11 to 24 which implies that in Line 25 all actions in best are classified.

$\square$

Proposition 5.5.16. (On Page 150) Procedure 5.5.1 satisfies the Principles I.1, I.2 and II, as given in Definition 5.5.11.

Proof.
Principle I.1:
Given $(\Xi, \preceq_{\mathsf{bel}})$ as parameter, let cl be the classification function as defined by Algorithm 5.5.1. We prove that, for all $\mathsf{Act}'_{inf} \subseteq \mathsf{Act}$, for all $\mathcal{K} \in \Omega$ and for all $a, b \in \mathsf{Act}'_{inf}$ such that $\mathsf{vioAfter}(\mathcal{K}, b) \sqsupset \mathsf{vioAfter}(\mathcal{K}, a)$ it holds $\mathsf{cl}(b) > \mathsf{cl}(a)$.

1. From Proposition 5.5.15 follows that for some iteration of the repeat-until loop starting in Line 5 it is $b \in \mathsf{best}$ in Line 6. In

this iteration $b \in \mathsf{unclass}$ and there is no $a' \in \mathsf{unclass}$ such that $\mathsf{vioAfter}(\mathcal{K}, b) \sqsupset \mathsf{vioAfter}(\mathcal{K}, a')$.

2. Hence, in particular $a \notin \mathsf{unclass}$ and consequently $a \notin \mathsf{best}$. Since $\mathsf{unclass} = \mathsf{Act}'_{inf}$ initially and then only classified actions are removed from it, as shown in the proof of Proposition 5.5.15 it follows that at this point a is already classified.

3. Action b will be classified in a following iteration of Line 17 with the value of $\mathsf{rank}[A]$ with $b \in A$.

4. The value of $\mathsf{rank}[A]$ that is determined in Line 9. By assumption $\mathsf{vioAfter}(\mathcal{K}, b) \sqsupset \mathsf{vioAfter}(\mathcal{K}, a)$, and $b \in A$ and $a \in \mathsf{Act}'_{inf} \setminus \mathsf{unclass}$ such that by Line 9 it follows that

$$\mathsf{rank}[A] = \mathsf{cl}(b) > \mathsf{cl}(a).$$

□

Principle I.2:
Given $(\Xi, \preceq_{\mathsf{bel}})$ as parameter, let cl be the classification function as defined by Algorithm 5.5.1. We prove that, for all $\mathsf{Act}'_{inf} \subseteq \mathsf{Act}$, for all $\mathcal{K} \in \Omega$ and for all $a, b \in \mathsf{Act}'_{inf}$ with $\mathsf{vioAfter}(\mathcal{K}, b) \sim \mathsf{vioAfter}(\mathcal{K}, a)$ and $a \prec_{vio} b$:

1. *Conflict Free*: If there do not exist actions $a_1, \dots, a_n \in \mathsf{Act}'_{inf}$ such that $a_1 = b$, $a_n = a$ and $a_i \prec_{vio} a_{i+1}$ for all $i \in \{1, \dots, n-1\}$, then it follows that $\mathsf{cl}(b) > \mathsf{cl}(a)$.

2. *Conflicting*: Otherwise, it follows $\mathsf{cl}(b) \geqslant \mathsf{cl}(a)$.

From the assumption $\mathsf{vioAfter}(\mathcal{K}, b) \sim \mathsf{vioAfter}(\mathcal{K}, a)$ follows by Definition of $\sqsupset$ that for all $c \in \mathsf{Act}'_{inf}$, if $\mathsf{vioAfter}(\mathcal{K}, a) \sqsupset \mathsf{vioAfter}(\mathcal{K}, c)$, then $\mathsf{vioAfter}(\mathcal{K}, b) \sqsupset \mathsf{vioAfter}(\mathcal{K}, c)$. Therefore, if in some iteration $a \in \mathsf{best}$, then also $b \in \mathsf{best}$ according to the definition of best in Line 6.

We consider the iteration of the repeat loop starting in Line 5 for which this is the case in the following. By definition of eqbest in Line 7 there exists $A \in \mathsf{eqbest}$ such that $a \in A$ and $b \in A$, since $\mathsf{vioAfter}(\mathcal{K}, b) \sim \mathsf{vioAfter}(\mathcal{K}, a)$.

1. By definition of $\mathsf{conflictSets}$ in Line 12 and the satisfaction of the condition of conflict freeness there exist

   $$\begin{aligned} & CS_a, CS_b \in \mathsf{conflictSets},\ CS_a \neq CS_b \text{ such that} \\ & \quad a \in CS_a \text{ and } b \in CS_b. \end{aligned}$$

   From Lemma 5.5.13 follows that for all $CS'' \in \mathsf{conflictSets}$ if $CS'' \neq CS_a$, then $a \notin CS''$; and if $CS'' \neq CS_b$, then $b \notin CS''$.

   We consider the iteration of the repeat-until loop starting in Line 13 and of the for loop starting in Line 15, in which a is classified in Line 17.

a) In Line 18 $\mathsf{classSets} := \mathsf{classSets} \cup \{CS_a\}$.

b) Action b is not classified yet in this iteration of the repeat-until loop starting in Line 13 since the condition in Line 16 is not satisfied for the CS_b iteration of the for loop starting in Line 15 as long as $CS_a \in \mathsf{conflictSets}$.

c) In the next execution of Line 21 $\mathsf{conflictSets} := \mathsf{conflictSets} \setminus \mathsf{classSets}$ with $CS_a \in \mathsf{classSets}$ and in Line 22 $\mathsf{rank}[A] := \mathsf{rank}[A] + 1$.

d) From Proposition 5.5.15 follows that b will be classified in the following. Since $b \in CS_b \subseteq A$ it will be classified in the same iteration of the for loop starting in Line 11 with the then current value of $\mathsf{rank}[A]$. Since $\mathsf{rank}[A]$ is only modified in Line 22, $\mathsf{rank}[A] := \mathsf{rank}[A] + 1$, it follows that $cl(b) > cl(a)$. □

2. By definition of $\mathsf{conflictSets}$ in Line 12 it follows that there exists some $CS \in \mathsf{conflictSets}$ such that $a \in CS$ and $b \in CS$.

   From Lemma 5.5.13 follows that for all $CS' \in \mathsf{conflictSets}$ if $CS' \neq CS$, then $a \notin CS'$ and $b \notin CS'$.

   From Line 17 follows that all $a' \in CS$ are assigned the same classification rank. Consequently $cl(a) = cl(b)$. □

Principle II:
Given $(\Xi, \preceq_{\mathsf{bel}})$ as parameter, let cl be the classification function as defined by Algorithm 5.5.1. Further, let sr' be another function $\mathsf{sr}' : \mathcal{P}_{\mathsf{fin}}(\mathsf{Act}) \times \Omega \to Cl$ fulfilling Principles I.1 and I.2. We prove that, for all $\mathsf{Act}'_{inf} \subseteq \mathsf{Act}$, for all $\mathcal{K} \in \Omega$ and for all $a \in \mathsf{Act}'_{inf}$ it holds $\mathsf{cl}'(a) \geqslant \mathsf{cl}(a)$ with $\mathsf{cl}' = \mathsf{sr}'(\mathsf{Act}'_{inf}, \mathcal{K})$.

We proceed by induction on the classification rank r in the range of cl. Thus, we consider the following induction hypothesis:
For all $i \leqslant r - 1$ it holds for all actions $a \in \mathsf{Act}'_{inf}$ with $\mathsf{cl}(a) = i$ that $\mathsf{cl}(a) \leqslant \mathsf{cl}'(a)$.

Let $a \in \mathsf{Act}'_{inf}$ be an action with $\mathsf{cl}(a) = r$.

– Base case: $r = 0$.
  By definition no lower classification rank is possible. Thus, it follows $\mathsf{cl}(a) \leqslant \mathsf{cl}'(a)$.

– Inductive case: $r > 0$.

  1. Assume indirectly that $\mathsf{cl}'(a) < r$ holds.
  2. Consider the classification of a by Algorithm 5.5.1. Action a is treated in one and only one iteration of the repeat-until loop in Line 5. Thus, for later reference, let A denote the equivalence class with $a \in A$ and $CS_a \subseteq A$ the conflict set of a as determined by the algorithm in this iteration.

We distinguish two cases how the algorithm computes the value of the classification rank of action a.

Case 1: The value is set in Line 9. Then, there exist $b \in \mathsf{Act}'_{inf}$ and $c \in A$ such that

$$\mathsf{vioAfter}(\mathcal{K}, c) \sqsupset \mathsf{vioAfter}(\mathcal{K}, b) \text{ and } \mathsf{cl}(b) = r - 1. \quad \text{(A.1.2)}$$

Since actions a, c are in the same equivalence class A, it follows that $\mathsf{vioAfter}(\mathcal{K}, a) \sim \mathsf{vioAfter}(\mathcal{K}, c)$. From the definitions of $\sqsupset$ and $\sim$, we can show that together with (A.1.2) the latter implies $\mathsf{vioAfter}(\mathcal{K}, a) \sqsupset \mathsf{vioAfter}(\mathcal{K}, b)$. Due to the relation $\sqsupset$ and Principle I.1, it holds

$$\mathsf{cl}'(a) > \mathsf{cl}'(b). \quad \text{(A.1.3)}$$

Now, we apply the induction hypothesis on $\mathsf{cl}(b)$, since $\mathsf{cl}(b) = r - 1$ holds by (A.1.2) and obtain that $\mathsf{cl}'(b) \geqslant \mathsf{cl}(b)$. By assumption, it holds $r > \mathsf{cl}'(a)$ and thus $\mathsf{cl}'(b) \geqslant \mathsf{cl}(b) = r - 1 \geqslant \mathsf{cl}'(a)$. This contradicts that $\mathsf{cl}'(a) > \mathsf{cl}'(b)$ must hold by (A.1.3).

Case 2: If Case 1 does not apply, the value for the classification of a in Line 17 is set in Line 22. Thus, there exists $b \in A$ with $\mathsf{cl}(b) = r - 1$ and $b \in CS_b \subset A$ with $CS_b \neq CS_a$ such that there exists $c \in CS_a$ with $b \prec_{vio} c$. More precisely, in the previous iteration of the for loop in Line 15 the conflict set CS_b must still be in $\mathsf{conflictSets}$ as one reason why a has been not classified with $r - 1$, but then CS_b is removed from $\mathsf{conflictSets}$ in Line 21.

Since b, c are in the same equivalence class A, it holds $\mathsf{vioAfter}(\mathcal{K}, c) \sim \mathsf{vioAfter}(\mathcal{K}, b)$. Further, since $c \in CS_a$ and $b \in CS_b$ and $CS_a \neq CS_b$, by definition of $\mathsf{conflictSets}$, there do not exist actions $a_1, \dots, a_n \in \mathsf{Act}'_{inf}$ such that $a_1 = c$, $a_n = b$ and for all $i \in \{1, \dots, n-1\}$ it holds $a_i \prec_{vio} a_{i+1}$. Hence, the premises of Principle I.2 are satisfied which implies

$$\mathsf{cl}'(c) > \mathsf{cl}'(b). \quad \text{(A.1.4)}$$

Next, we argue that the fact that $a \in CS_a$ and $c \in CS_a$ implies $\mathsf{cl}'(a) = \mathsf{cl}'(c)$. This follows from an inductive argument using the local mitigation requirement of Principle I.2 and the definition of CS_a. By (A.1.4), it holds $\mathsf{cl}'(a) > \mathsf{cl}'(b)$.

Now, we apply the induction hypothesis on $\mathsf{cl}(b) = r - 1$ so that it follows $\mathsf{cl}'(b) \geqslant \mathsf{cl}(b)$. By assumption, it holds $r > \mathsf{cl}'(a)$ and thus $\mathsf{cl}'(b) \geqslant \mathsf{cl}(b) = r - 1 \geqslant \mathsf{cl}'(a)$. This contradicts that $\mathsf{cl}'(a) > \mathsf{cl}'(b)$ must hold. □

□

BIBLIOGRAPHY

[1] Agentprogramming.com. Last checked: 2014-09-19. *Agent Platforms*. `http://agentprogramming.com/agent-platforms/`.

[2] Ahlstrom, Kristoffer. 2010. *On Epistemic Agency*. Ph.D. thesis, University of Massachusetts at Amherst.

[3] Albrecht, Ella. 2012. *Abhängigkeitsgraphen und Erklärungen für die Antwortmengen Programmierung (Dependency Graphs and Explanations for Answerset Programming)*. Bachelor's Thesis, Computer Science, Chair 1 - Information Engineering, Technische Universität Dortmund.

[4] Albrecht, Ella, Krümpelmann, Patrick, & Kern-Isberner, Gabriele. 2014. Construction of Explanation Graphs from Extended Dependency Graphs for Answer Set Programs. *Pages 1–16 of:* Hanus, Michael, & Rocha, Ricardo (eds), *Post-proceedings of the 27th Workshop on Functional and Logic Programming (WFLP 2013)*. LNAI, no. 8439. Springer.

[5] Alchourron, Carlos E., Gärdenfors, Peter, & Makinson, David. 1985. On the Logic of Theory Change: Partial Meet Contraction and Revision Functions. *The Journal of Symbolic Logic*, **50**(2), 510–530.

[6] Alechina, Natasha, Jago, Mark, & Logan, Brian. 2008. Preference-based belief revision for rule-based agents. *Synthese*, **165**(2), 159–177.

[7] Alferes, José Júlio, Leite, João Alexandre, Pereira, Luís Moniz, Przymusinska, Halina, & Przymusinski, Teodor C. 1998. Dynamic Logic Programming. *Pages 98–111 of:* Cohn, A., Schubert, L., & Shapiro, S. (eds), *Proceedings of the 6th International Conference on Principles of Knowledge Representation and Reasoning (KR'98)*. San Francisco: Morgan Kaufmann Publishers.

[8] Alferes, José Júlio, Banti, Federico, Brogi, Antonio, & Leite, João Alexandre. 2004. Semantics for dynamic logic programming: A principled-based approach. *In: Proceedings of the 7th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'04)*, vol. 1730. Springer.

[9] Alferes, José Júlio, Banti, Federico, Brogi, Antonio, & Leite, João Alexandre. 2005. The Refined Extension Principle for Semantics of Dynamic Logic Programming. *Studia Logica: An International Journal for Symbolic Logic*, **79**(1), 7–32.

[10] Arrow, Kenneth J. 2012. *Social choice and individual values*. Vol. 12. Yale university press.

[11] Aucher, Guillaume, Boella, Guido, & van der Torre, Leendert. 2011. A dynamic logic for privacy compliance. *Artificial Intelligence and Law*, **19**, 187–231. 10.1007/s10506-011-9114-3.

[12] Baader, Franz. 2003. *The description logic handbook: theory, implementation, and applications*. Cambridge university press.

[13] Bacchus, Fahiem, Grove, Adam J., Halpern, Joseph Y., & Koller, Daphne. 1996. From statistical knowledge bases to degrees of belief. *Artificial Intelligence*, **87**, 75 – 143.

[14] Bach, Joscha. 2011. A Motivational System for Cognitive AI. *Pages 232–242 of:* Schmidhuber, Jürgen, Thórisson, Kristinn, & Looks, Moshe (eds), *Artificial General Intelligence*. Lecture Notes in Computer Science, vol. 6830. Springer Berlin / Heidelberg.

[15] Balduccini, Marcello, & Gelfond, M. 2003. Logic Programs with Consistency-Restoring Rules. *In:* P. Doherty, J. McCarthy, M.-A. Williams (ed), *Proceedings of the International Symposium on Logical Formalization of Commonsense Reasoning*. AAAI 2003 Spring Symposium Series.

[16] Baral, Chitta, & Gelfond, Michael. 2000. Reasoning Agents in Dynamic Domains. *Pages 257–279 of:* Minker, Jack (ed), *Logic-Based Artificial Intelligence*. The Springer International Series in Engineering and Computer Science, vol. 597. Springer US.

[17] Baral, Chitta, Eiter, Thomas, Bjäreland, Marcus, & Nakamura, Mutsumi. 2008a. Maintenance goals of agents in a dynamic environment: Formulation and policy construction. *Artificial Intelligence*, **172**(12), 1429–1469.

[18] Baral, Chitta, Gelfond, Michael, & Rushton, Nelson. 2008b. Probabilistic reasoning with answer sets. *Theory and Practice of Logic Programming*, 12.

[19] Barbi, Manuel. 2014. *Implementierung einer Motivationskomponente für wissensbasierte Agenten (Implementation of a Motivation Component for Knowledge-based Agents)*. Bachelor's Thesis, Computer Science, Chair 1 - Information Engineering, Technische Universität Dortmund.

[20] Baroni, Pietro, & Giacomin, Massimiliano. 2009. Skepticism relations for comparing argumentation semantics. *International Journal of Approximate Reasoning*, **50**(6), 854 – 866.

[21] Baroni, Pietro, Caminada, Martin, & Giacomin, Massimiliano. 2011. Review: An Introduction to Argumentation Semantics. *Knowledge Engineering Review*, **26**(4), 365–410.

[22] Bell, David E., Raiffa, Howard, & Tversky, Amos (eds). 1988. *Decision Making*. Cambridge University Press. Cambridge Books Online.

[23] Bellifemine, Fabio, Poggi, Agostino, & Rimassa, Giovanni. 2001. Developing Multi-agent Systems with JADE. *Pages 89–103 of:* Castelfranchi, Cristiano, & Lespérance, Yves (eds), *Intelligent Agents VII Agent Theories Architectures and Languages*. Lecture Notes in Computer Science, vol. 1986. Springer Berlin Heidelberg.

[24] Bench-Capon, Trevor J. M., & Dunne, Paul E. 2007. Argumentation in artificial intelligence. *Artificial Intelligence*, **171**(10-15), 619–641.

[25] Benerecetti, Massimo, Giunchiglia, Fausto, & Serafini, Luciano. 1998. Model checking multiagent systems. *Journal of Logic and Computation*, **8**(3), 401–423.

[26] Besnard, P., & Hunter, A. 2001. A logic-based theory of deductive arguments. *Artificial Intelligence*, **128**(1-2), 203–235.

[27] Besnard, Philippe, & Hunter, Anthony. 2006. Knowledgebase Compilation for Efficient Logical Argumentation. *Pages 123–133 of: Proceedings of the 10th International Conference on Knowledge Representation (KR'06)*. AAAI Press.

[28] Besnard, Philippe, & Hunter, Anthony. 2008. *Elements of Argumentation*. The MIT Press.

[29] Billington, D., Antoniou, G., Governatori, G., & Maher, M. 1999. Revising Nonmonotonic Theories: The Case of Defeasible Logic. *Pages 101–112 of:* Burgard, Wolfram, Cremers, Armin, & Cristaller, Thomas (eds), *KI-99: Advances in Artificial Intelligence*, vol. 1701. Springer.

[30] Binnewies, Sebastian, Zhuang, Zhiqiang, & Wang, Kewen. 2015. Partial Meet Revision and Contraction in Logic Programs. *In: Proceedings of the 29th AAAI Conference on Artificial Intelligence*. AAAI Publications.

[31] Biskup, J., & Bonatti, P.A. 2004. Controlled query evaluation for enforcing confidentiality in complete information systems. *International Journal of Information Security*, **3**(1), 14–27.

[32] Biskup, Joachim. 2010. Usability Confinement of Server Reactions: Maintaining Inference-Proof Client Views by Controlled Interaction Execution. *Pages 80–106 of:* Kikuchi, Shinji, Sachdeva, Shelly, & Bhalla, Subhash (eds), *Databases in Networked Information Systems*. Lecture Notes in Computer Science, vol. 5999. Springer Berlin / Heidelberg.

[33] Biskup, Joachim. 2011. History-dependent inference control of queries by dynamic policy adaption. *Pages 106–121 of: Data and applications security and privacy XXV*. Springer.

[34] Biskup, Joachim. 2012. Inference-usability confinement by maintaining inference-proof views of an information system. *International Journal of Computational Science and Engineering*, 7(1), 17–37.

[35] Biskup, Joachim, & Bonatti, Piero. 2007. Controlled query evaluation with open queries for a decidable relational submodel. *Annals of Mathematics and Artificial Intelligence*, **50**, 39–77. 10.1007/s10472-007-9070-5.

[36] Biskup, Joachim, & Tadros, Cornelia. 2010. Policy-Based Secrecy in the Runs & Systems Framework and Controlled Query Evaluation. *Pages 60–77 of:* Echizen, Isao, Kunihiro, Noboru, & Sasaki, Ryôichi (eds), *Short paper of the 5th International Workshop on Security (IWSEC'10)*. Information Processing Society of Japan (IPSJ).

[37] Biskup, Joachim, & Tadros, Cornelia. 2012. Inference-Proof View Update Transactions with Minimal Refusals. *Pages 104–121 of:* Garcia-Alfaro, Joaquin, Navarro-Arribas, Guillermo, Cuppens-Boulahia, Nora, & De Capitani di Vimercati, Sabrina (eds), *6th International Workshop on Data Privacy Management (DPM 2011)*. LNCS, vol. 7122. Springer.

[38] Biskup, Joachim, & Weibert, Torben. 2008. Keeping Secrets in Incomplete Databases. *International Journal of Information Security*, 7(3), 199–217.

[39] Biskup, Joachim, Kern-Isberner, Gabriele, & Thimm, Matthias. 2008. Towards Enforcement of Confidentiality in Agent Interactions. *Pages 104–112 of:* Pagnucco, Maurice, & Thielscher, Michael (eds), *Proceedings of the 12th International Workshop on Non-Monotonic Reasoning (NMR'08)*. Sydney, Australia: University of New South Wales, Technical Report No. UNSW-CSE-TR-0819.

[40] Biskup, Joachim, Tadros, Cornelia, & Wiese, Lena. 2010. Towards controlled query evaluation for incomplete first-order databases. *Pages 230–247 of: Foundations of Information and Knowledge Systems*. Springer.

[41] Biskup, Joachim, Kern-Isberner, Gabriele, Krümpelmann, Patrick, & Tadros, Cornelia. 2014. Reasoning on Secrecy Constraints under Uncertainty to Classify Possible Actions. *Pages 97–116 of:* Beierle, Christoph, & Meghini, Carlo (eds), *8th Proceedings of the 8th International Symposium on Foundations of Information and Knowledge Systems (FoIKS2014)*. LNCS, vol. 8367. Springer.

[42] Blackburn, Patrick, de Rijke, Maarten, & Venema, Yde. 2001. *Modal Logic*. New York, NY, USA: Cambridge University Press.

[43] Bochman, Alexander. 2001. *A Logical Theory of Nonmonotonic Inference and Belief Change*. New York, NY, USA: Springer-Verlag New York, Inc.

[44] Boenn, Georg, Brain, Martin, De Vos, Marina, & Ffitch, John. 2011. Automatic music composition using answer set programming. *Theory and practice of logic programming*, **11**(2-3), 397–427.

[45] Böhmer, Mirja. 2011. *Evaluation von Ansätzen zur Wissensänderung von ASP Wissensbasen (Evaluation of approaches to belief change in ASP knowldege bases)*. Diplomarbeit, Computer Science, Chair 1 - Information Engineering, Technische Universität Dortmund.

[46] Bonatti, P. A., Kraus, S., & Subrahmanian, V. S. 1995. Foundations of secure deductive databases. *IEEE Transactions on Knowledge and Data Engineering*, **7**, 406–422.

[47] Bonatti, Piero, Calimeri, Francesco, Leone, Nicola, & Ricca, Francesco. 2010. Answer Set Programming. *Pages 159–182 of:* Dovier, Agostino, & Pontelli, Enrico (eds), *A 25-Year Perspective on Logic Programming*. Lecture Notes in Computer Science, vol. 6125. Springer Berlin / Heidelberg.

[48] Booth, Richard. 2001. A negotiation-style framework for non-prioritised revision. *Pages 137–150 of: Proceedings of the 8th Conference on Theoretical Aspects of Rationality and Knowledge (TARK'01)*.

[49] Booth, Richard, & Nittka, Alexander. 2008. Reconstructing an Agent's Epistemic State from Observations about its Beliefs and Non-beliefs. *The Journal of Logic Programming*, **18**(5), 755–782.

[50] Booth, Richard, Meyer, Thomas, & Varzinczak, Ivan José. 2009. Next steps in propositional horn contraction. *Pages 702–707 of: Proceedings of the 21st international jont conference on Artifical intelligence (IJCAI'09)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

[51] Bordini, Rafael H., Braubach, Lars, Dastani, Mehdi, Seghrouchni, Amal El Fallah, Gomez-Sanz, Jorge J., Leite, João, O'Hare, Gregory, Pokahr, Alexander, & Ricci, Alessandro. 2006. A survey of programming languages and platforms for multiagent systems. *Informatica*, **30**, 33–44.

[52] Börger, Benedict. 2014. *Modellierung und Evaluation von vertraulichkeitsbewahrenden Agenten in einem E-Markt Szenario mittels Antwortmengenprogrammierung (Conceptualization and evaluation of secrecy preserving agents in an E-Market scenario with answerset programming)*. Bachelor's Thesis, Computer Science, Chair 1 - Information Engineering, Technische Universität Dortmund.

[53] Börger, Egon, & Stärk, Robert F. 2003. *Abstract State Machines: A Method for High-level System Design and Analysis*. Springer.

[54] Bouyssou, Denis, Pirlot, Marc, & Vincke, Ph. 1997. A general model of preference aggregation. *Pages 120–134 of: Essays in decision making*. Springer.

[55] Brachman, Ronald J., & Levesque, Hector J. 2004. *Knowledge Representation and Reasoning*. Elsevier and Morgan Kaufmann Publishers.

[56] Bratman, Michael. 1987. *Intention, plans, and practical reason*. Harvard University Press (Cambridge, Mass.).

[57] Bratman, Michael E., Israel, David J., & Pollack, Martha E. 1988. Plans and resource-bounded practical reasoning. *Computational Intelligence*, **4**(3), 349–355.

[58] Brewka, G., & Eiter, T. 1999a. Prioritizing default logic: Abridged report. *Festschrift on the occasion of Prof. Dr. W. Bibel's 60th birthday. Kluwer, Dordrecht*.

[59] Brewka, Gerhard, & Eiter, Thomas. 1999b. Preferred answer sets for extended logic programs. *Artifical Intelligence*, **109**(1-2), 297–356.

[60] Brewka, Gerhard, Dix, Jürgen, & Konolige, Kurt. 1997. *Nonmonotonic Reasoning: An Overview*. CSLI Lecture Notes, vol. 73. CSLI Publications, Stanford, CA.

[61] Brown, Donald J. 1975. Aggregation of preferences. *The Quarterly Journal of Economics*, 456–469.

[62] Calimeri, Francesco, Cozza, Susanna, Ianni, Giovambattista, & Leone, Nicola. 2008. Computable Functions in ASP: Theory and Implementation. *Pages 407–424 of:* Garcia de la Banda, Maria, & Pontelli, Enrico (eds), *Logic Programming*. Lecture Notes in Computer Science, vol. 5366. Springer Berlin Heidelberg.

[63] Calimeri, Francesco, Ianni, Giovambattista, & Ricca, Francesco. 2014. The third open answer set programming competition. *Theory and Practice of Logic Programming*, **14**(01), 117–135.

[64] Cohen, Philip R, & Levesque, Hector J. 1997. Communicative actions for artificial agents. *Pages 419–436 of: Software agents*. MIT Press.

[65] Darwiche, Adnan, & Pearl, Judea. 1994. On the Logic of Iterated Belief Revision. *Pages 5–23 of:* Fagin, Ronald (ed), *Proceedings of the 5th Conference on Theoretical Aspects of Reasoning about Knowledge*. Pacific Grove, CA: Morgan Kaufmann.

[66] Dastani, Mehdi, Birna Riemsdijk, M., & Meyer, John-JulesCh. 2005. Programming Multi-Agent Systems in 3APL. *Pages 39–67 of:* Bordini, RafaelH., Dastani, Mehdi, Dix, Jürgen, & Fallah Seghrouchni, Amal (eds), *Multi-Agent Programming*. Multiagent Systems, Artificial Societies, and Simulated Organizations, vol. 15. Springer US.

[67] Dastani, Mehdi, van Riemsdijk, M. Birna, & Winikoff, Michael. 2011. Rich goal types in agent programming. *Pages 405–412 of: The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*. AAMAS '11. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

[68] Delgrande, J., Schaub, T., Tompits, H., & Woltran, S. 2008. Belief Revision of Logic Programs under Answer Set Semantics. *Pages 411–421 of:* Brewka, G., & Lang, J. (eds), *Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning (KR'08)*. AAAI Press.

[69] Delgrande, James, & Jin, Yi. 2012. Parallel belief revision: Revising by sets of formulas. *Artifical Intelligence*, **176**(1), 2223–2245.

[70] Delgrande, James, & Wassermann, Renata. 2010. Horn Clause Contraction Functions: Belief Set and Belief Base Approaches. *In: Knowledge Representation and Reasoning Conference (KR 2010)*. AAAI.

[71] Delgrande, James, Schaub, Torsten, Tompits, Hans, & Woltran, Stefan. 2009a. Merging Logic Programs under Answer Set Semantics. *Pages 160–174 of: Proceedings of the 25th International Conference on Logic Programming (ICLP'09)*. ICLP '09. Berlin, Heidelberg: Springer-Verlag.

[72] Delgrande, James, Peppas, Pavlos, & Woltran, Stefan. 2013. AGM-Style Belief Revision of Logic Programs under Answer Set Semantics. *Pages 264–276 of:* Cabalar, Pedro, & Son, TranCao (eds), *Logic Programming and Nonmonotonic Reasoning*. Lecture Notes in Computer Science, vol. 8148. Springer.

[73] Delgrande, James P. 2008. Horn Clause Belief Change: Contraction Functions. *Pages 156–165 of:* Brewka, Gerhard, & Lang, Jérôme (eds), *Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning (KR'08)*. AAAI Press.

[74] Delgrande, James P. 2010. An Approach to Revising Logic Programs under the Answer Set Semantics. *Proceedings of the 13th International Workshop on Non-Monotonic Reasoning (NMR'10)*.

[75] Delgrande, James P., Schaub, Torsten, & Tompits, Hans. 2007. A Preference-Based Framework for Updating Logic Programs. *Pages 71–83 of:* Baral, Chitta, Brewka, Gerhard, & Schlipf, John S. (eds),

Proceedings of the 9th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'07), vol. 4483. Springer.

[76] Delgrande, James P., Schaub, Torsten, Tompits, Hans, & Woltran, Stefan. 2009b. A general approach to belief change in answer set programming. *Computing Research Repository (CoRR)*, **abs/0912.5511**.

[77] Dell'Armi, T., Faber, W., Ielpa, G., Leone, N., & Pfeifer, G. 2004. System Description: DLV with Aggregates. *Pages 326–330 of:* V. Lifschitz, I. Niemelä (ed), *Proceedings of the 7th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'04)*. LNAI, vol. 2923. Springer.

[78] Dilger, Daniel, Krümpelmann, Patrick, & Tadros, Cornelia. 2013. *Preserving Confidentiality in Multiagent Systems - An Internship Project within the DAAD RISE Program*. Tech. rept. 1. Technische Universität Dortmund.

[79] Dix, Jürgen, Faber, Wolfgang, & Subrahmanian, VS. 2012. Privacy preservation using multi-context systems and default logic. *Pages 195–210 of: Correct Reasoning*. Springer.

[80] Doyle, Jon. 1987. *A truth maintenance system*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. Pages 259–279.

[81] Duff, Simon, Harland, James, & Thangarajah, John. 2006. On proactivity and maintenance goals. *Pages 1033–1040 of:* Nakashima, Hideyuki, Wellman, Michael P., Weiss, Gerhard, & Stone, Peter (eds), *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, May 8-12*. ACM.

[82] Duff, Simon, Thangarajah, John, & Harland, James. 2014. Maintenance goals in intelligent agents. *Computational Intelligence*, **30**(1), 71–114.

[83] Dung, Phan Minh. 1995. On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games. *Artificial Intelligence*, **77**(2), 321–358.

[84] Eiter, Thomas, Faber, Wolfgang, Leone, Nicola, & Pfeifer, Gerald. 2000a. Declarative problem-solving using the DLV system. *Logic-based artificial intelligence*, 79–103.

[85] Eiter, Thomas, Faber, W., Koch, C., Leone, N., & Pfeifer, G. 2000b. DLV - A system for declarative problem solving. *In:* Baral, C., & Truszczynski, M. (eds), *Proceedings of the 8th International Workshop on Non-Monotonic Reasoning*.

[86] Eiter, Thomas, Faber, Wolfgang, Leone, Nicola, Pfeifer, Gerald, & Polleres, Axel. 2000c. Planning under incomplete knowledge. *Pages 807–821 of: Computational Logic—CL 2000*. Springer.

[87] Eiter, Thomas, Fink, Michael, Sabbatini, Giuliana, & Tompits, Hans. 2002. On properties of update sequences based on causal rejection. *Theory and Practice of Logic Programming*, **2**(6), 711–767.

[88] Eiter, Thomas, Ianni, Giovambattista, Schindlauer, Roman, & Tompits, Hans. 2006. Towards Efficient Evaluation of HEX Programs. *Pages 40–46 of: Proceedings of 11th International Workshop on Non-Monotonic Reasoning (NMR'06)*. Technical Report, vol. IfI-06-04. KR Inc.

[89] Fagin, Ronald, Halpern, Joseph Y., Moses, Yoram, & Vardi, Moshe Y. 1995. *Reasoning about Knowledge*. MIT Press.

[90] Falappa, Marcelo A., Kern-Isberner, Gabriele, & Simari, Guillermo R. 2002. Explanations, belief revision and defeasible reasoning. *Artificial Intelligence*, **141**(1), 1–28.

[91] Falappa, Marcelo Alejandro, Kern-Isberner, Gabriele, & Simari, Guillermo Ricardo. 2009. Belief Revision and Argumentation Theory. *Pages 341–360 of: Argumentation in Artificial Intelligence*. Springer.

[92] Falappa, Marcelo Alejandro, García, Alejandro Javier, Kern-Isberner, Gabriele, & Simari, Guillermo Ricardo. 2013. Stratified belief bases revision with argumentative inference. *Journal of Philosophical Logic*, **42**(1), 161–193.

[93] Farkas, C., & Jajodia, S. 2002. The inference problem: a survey. *ACM Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD) Explorations Newsletter*, **4**, 6–11.

[94] Fermé, Eduardo, & Hansson, Sven. 2011. AGM 25 Years. *Journal of Philosophical Logic*, **40**, 295–331. 10.1007/s10992-011-9171-9.

[95] Fermé, Eduardo, Saez, Karina, & Sanz, Pablo. 2003. Multiple Kernel Contraction. *Studia Logica: An International Journal for Symbolic Logic*, **73**(2), 183–195.

[96] Fermé, Eduardo L., & Hansson, Sven Ove. 1999. Selective Revision. *Studia Logica*, **63**(3), 331–342.

[97] Fischer, M. 1994. A survey of concurrent MetateM - the language and its applications. *Pages 480–505 of:* Gabey, D.M., & Ohlbach, H.J. (eds), *Proceedings of the First International Conference on Temporal Logic*. Lecture Notes in Computer Science, vol. 827. Springer.

[98] Fishburn, P.C. 1970. *Utility Theory for Decision Making*. Wiley, New York.

[99] Fitting, Melvin. 2002. Fixpoint semantics for logic programming a survey. *Theoretical computer science*, **278**(1), 25–51.

[100] Foundation for Intelligent Physical Agents. 2002a (12). *FIPA ACL Message Structure Specification*. `http://www.fipa.org/specs/fipa00061/SC00061G.html`.

[101] Foundation for Intelligent Physical Agents. 2002b (12). *FIPA Communicative Act Library Specification*. `http://www.fipa.org/specs/fipa00061/SC00061G.html`.

[102] Freuder, Eugene C, Minca, Marius, & Wallace, Richard J. 2001. Privacy/efficiency tradeoffs in distributed meeting scheduling by constraint-based agents. *Pages 63–72 of: Proc. IJCAI DCR*.

[103] Fuhrmann, André, & Hansson, SvenOve. 1994. A survey of multiple contractions. *Journal of Logic, Language and Information*, **3**(1), 39–75.

[104] Gabbay, D.M. 1985. Theoretical Foundations for Non-Monotonic Reasoning in Expert Systems. *Pages 439–457 of:* Apt, KrzysztofR. (ed), *Logics and Models of Concurrent Systems*. NATO ASI Series, vol. 13. Springer Berlin Heidelberg.

[105] García, A., & Simari, G. 2004. Defeasible Logic Programming: An Argumentative Approach. *Theory and Practice of Logic Programming*, **4**(1+2), 95–138.

[106] Gärdenfors, Peter, & Rott, Hans. 1995. *Belief revision*. Oxford University Press.

[107] Garey, Michael R, & Johnson, David S. 1979. *Computers and intractability*. Vol. 174. Freeman New York.

[108] Gebser, M., Kaminski, R., Kaufmann, B., Ostrowski, M., Schaub, T., & Schneider, M. 2011. Potassco: The Potsdam Answer Set Solving Collection. *AI Communications*, **24**(2), 105–124.

[109] Gebser, M., Kaminski, R., Kaufmann, B., & Schaub, T. 2012. *Answer Set Solving in Practice*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool Publishers.

[110] Gebser, Martin, Guziolowski, Carito, Ivanchev, Mihail, Schaub, Torsten, Siegel, Anne, Thiele, Sven, & Veber, Philippe. 2010. Repair and Prediction (under Inconsistency) in Large Biological Networks with Answer Set Programming. *In: KR*.

[111] Gelfond, Michael. 2004. Answer Set Programming and the Design of Deliberative Agents. *Pages 19–26 of:* Demoen, Bart, & Lifschitz, Vladimir (eds), *ICLP*. Lecture Notes in Computer Science, vol. 3132. Springer.

[112] Gelfond, Michael, & Leone, Nicola. 2002. Logic programming and knowledge representation: the A-Prolog perspective. *Artificial Intelligence*, **138**(1), 3–38.

[113] Gelfond, Michael, & Lifschitz, Vladimir. 1988. The Stable Model Semantics For Logic Programming. *Pages 1070–1080 of: Proceedings of the 5th International Conference and Symposium on Logic Programming (ICLP/SLP 1988).* MIT Press.

[114] Gelfond, Michael, & Lifschitz, Vladimir. 1991. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, **9**(3/4), 365–386.

[115] Genesereth, Michael, & Nilsson, Nils. 1987a. *Logical Foundations of Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann.

[116] Genesereth, Michael R, & Nilsson, Nils J. 1987b. *Logical foundations of artificial intelligence*. Vol. 9. Morgan Kaufmann Los Altos, CA.

[117] Georgeff, Michael P., & Lansky, Amy L. 1987. Reactive Reasoning and Planning. *Pages 677–682 of:* Forbus, Kenneth D., & Shrobe, Howard E. (eds), *AAAI*. Morgan Kaufmann.

[118] Georgeff, Michael P., & Rao, Anand S. 1996. A profile of the Australian Artificial Intelligence Institute. *IEEE Expert*, **11**(6), 89–92.

[119] Giusto, Paolo Di, & Governatori, Guido. 1999. A New Approach to Base Revision. *Pages 327–341 of: Proceedings of the 9th Portuguese Conference on Artificial Intelligence: Progress in Artificial Intelligence (EPIA '99).* EPIA '99. London, UK: Springer-Verlag.

[120] Gray III, James W, & Syverson, Paul F. 1998. A logical approach to multilevel security of probabilistic systems. *Distributed Computing*, **11**(2), 73–90.

[121] Halpern, Joseph Y., & O'Neill, Kevin R. 2008. Secrecy in Multiagent Systems. *ACM Transactions on Information and System Security*, **12**(October), 5:1–5:47.

[122] Hansson, Sven Ove. 1991. *Belief base dynamics*. Ph.D. thesis, Uppsala University, Faculty of Arts.

[123] Hansson, Sven Ove. 1992. In Defense of Base Contraction. *Synthese*, **91**(3), 239–245.

[124] Hansson, Sven Ove. 1994. Kernel Contraction. *The Journal of Symbolic Logic*, **59**(3), 845–859.

[125] Hansson, Sven Ove. 1997. Semi-Revision. *Journal of Applied Non-Classical Logics*, **7**(2).

[126] Hansson, Sven Ove. 1999. A Survey of Non-Prioritized Belief Revision. *Erkenntnis*, **50**(2-3), 413–427.

[127] Hansson, Sven Ove. 2001. *A Textbook of Belief Dynamics*. Norwell, MA, USA: Kluwer Academic Publishers.

[128] Hansson, Sven Ove. 2010. Multiple and iterated contraction reduced to single-step single-sentence contraction. *Synthese*, **173**(2).

[129] Hindriks, Koen, & van Riemsdijk, M. 2008. Satisfying Maintenance Goals. *Pages 86–103 of:* Baldoni, Matteo, Son, Tran, van Riemsdijk, M., & Winikoff, Michael (eds), *Declarative Agent Languages and Technologies V*. Lecture Notes in Computer Science, vol. 4897. Springer Berlin / Heidelberg.

[130] Hindriks, KoenV., de Boer, FrankS., van der Hoek, Wiebe, & Meyer, John-JulesCh. 2001. Agent Programming with Declarative Goals. *Pages 228–243 of:* Castelfranchi, Cristiano, & Lespérance, Yves (eds), *Intelligent Agents VII Agent Theories Architectures and Languages*. Lecture Notes in Computer Science, vol. 1986. Springer Berlin Heidelberg.

[131] Homann, Sebastian. 2013. *Argumentationsbasierte selektive Revision erweiterter logischer Programme (Argumentation based selective revision of extended logic programs)*. Master's thesis, Computer Science, Chair 1 - Information Engineering, Technische Universität Dortmund.

[132] Hué, Julien, Würbel, Eric, & Papini, Odile. 2008. Removed Sets Fusion: Performing Off The Shelf. *Pages 94–98 of: Proceedings of the 2008 conference on ECAI 2008: 18th European Conference on Artificial Intelligence*. IOS Press.

[133] Hué, Julien, Papini, Odile, & Würbel, Eric. 2009. Merging Belief Bases Represented by Logic Programs. *Pages 371–382 of:* Sossai, Claudio, & Chemello, Gaetano (eds), *Proceedings of the 10th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'09)*. Lecture Notes in Computer Science, vol. 5590. Springer.

[134] Hughes, Dominic, & Shmatikov, Vitaly. 2004. Information hiding, anonymity and privacy: a modular approach. *Journal of Computer Security*, **12**(January), 3–36.

[135] Inoue, Katsumi, Sakama, Chiaki, & Wiese, Lena. 2011. Confidentiality-Preserving Data Publishing for Credulous Users by Extended Abduction. *Computing Research Repository (CoRR)*, **abs/1108.5825**.

[136] Jamroga, Wojciech, & Ågotnes, Thomas. 2007. Modular Interpreted Systems. *Pages 131:1–131:8 of: Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*. AAMAS '07. New York, NY, USA: ACM.

[137] Janus, Tim. 2013. *Resource-bounded Planning of Communication under Confidentiality Constraints for BDI Agents*.

[138] Kakas, A., Amgoud, L., Kern-Isberner, G., Maudet, N., & Moraitis, P. 2012. ABA: Argumentation Based Agents. *Pages 9–27 of:* McBurney, Peter, Parsons, Simon, & Rahwan, Iyad (eds), *Argumentation in Multi-Agent Systems*. Lecture Notes in Computer Science, vol. 7543. Springer Berlin Heidelberg.

[139] Kakas, Antonis C, Mancarella, Paolo, Sadri, Fariba, Stathis, Kostas, & Toni, Francesca. 2004. The KGP model of agency. *In: Proceedings of 16th European Conference on Artificial Intelligence (ECAI)*. IOS Press.

[140] Kakas, Antonis C., Mancarella, Paolo, Sadri, Fariba, Stathis, Kostas, & Toni, Francesca. 2008. Computational Logic Foundations of KGP Agents. *Journal of Artificial Intelligence Research (JAIR)*, **33**, 285–348.

[141] Katsuno, Hirofumi, & Mendelzon, Alberto. 1994. On the Difference Between Updating a Knowledge Base and Revising It. *Pages 387–394 of:* Allen, James F., Fikes, Richard, & Sandewall, Erik (eds), *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*. San Mateo, California: Morgan Kaufmann.

[142] Katsuno, Hirofumi, & Mendelzon, Alberto O. 1991. Propositional knowledge base revision and minimal change. *Artifical Intelligence*, **52**(December), 263–294.

[143] Kern-Isberner, Gabriele. 2001. *Conditionals in nonmonotonic reasoning and belief revision: considering conditionals as agents*. Berlin, Heidelberg: Springer-Verlag.

[144] Kern-Isberner, Gabriele, & Krümpelmann, Patrick. 2011. A constructive approach to independent and evidence retaining belief revision by general information sets. *Pages 937–942 of: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI'11)*. AAAI Press.

[145] Kern-Isberner, Gabriele, & Thimm, Matthias. 2010. Novel Semantical Approaches to Relational Probabilistic Conditionals. *In: Proceedings of the Twelfth International Conference on the Principles of Knowledge Representation and Reasoning (KR 2010)*.

[146] Kern-Isberner, Gabriele, & Thimm, Matthias. 2012. A Ranking Semantics for First-Order Conditionals. *Pages 456–461 of:* L. De Raedt, C. Bessiere, D. Dubois (ed), *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI)*. IOS Press.

[147] Kodaganallur, V. 2004. Incorporating language processing into java applications: A JavaCC tutorial. *Software, IEEE*, **21**(4), 70–77.

[148] Koen V. Hindriks, Wouter Pasman. 2014. *GOAL User Manual*. Delft University of Technology.

[149] Konolige, Kurt. 1986. *A Deduction Model of Belief*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

[150] Kowalski, Robert, & Sadri, Fariba. 1999. From logic programming towards multi-agent systems. *Annals of Mathematics and Artificial Intelligence*, **25**(3-4), 391–419.

[151] Krümpelmann, Patrick. 2012. Dependency Semantics for Sequences of Extended Logic Programs. *Logic Journal of the IGPL*, **20**(5), 943–966.

[152] Krümpelmann, Patrick, & Kern-Isberner, Gabriele. 2008. Propagating Credibility in Answer Set Programs. *In:* Schwarz, Sibylle (ed), *Proceedings of the 22nd Workshop on (Constraint) Logic Programming (WLP'08)*. Technische Berichte. Martin-Luther-Universität Halle-Wittenberg, Germany.

[153] Krümpelmann, Patrick, & Kern-Isberner, Gabriele. 2010. On belief dynamics of dependency relations for extended logic programs. *In: Proceedings of the 13th International Workshop on Non-Monotonic Reasoning (NMR'10)*.

[154] Krümpelmann, Patrick, & Kern-Isberner, Gabriele. 2012a. Belief Base Change Operations for Answer Set Programming. *In: Proceedings of the 13th European Conference on Logics in Artificial Intelligence (JELIA'12)*. Lecture Notes in Artificial Intelligence, vol. 7519. Springer.

[155] Krümpelmann, Patrick, & Kern-Isberner, Gabriele. 2012b. On Agent-based Epistemic Secrecy. *In:* Rossi, Riccardo, & Woltran, Stefan (eds), *Proceedings of the 14th International Workshop on Non-Monotonic Reasoning (NMR'12)*.

[156] Krümpelmann, Patrick, & Kern-Isberner, Gabriele. 2013. Secrecy preserving BDI Agents based on Answerset Programming. *Pages 124–137 of: Proceedings of the 11th German Conference on Multi-Agent System Technologies (MATES'13)*. Lecture Notes in Computer Science, vol. 8076. Springer.

[157] Krümpelmann, Patrick, & Thimm, Matthias. 2010. A Logic Programming Framework for Reasoning about Know-How. *In: Proceedings of the 13th International Workshop on Non-Monotonic Reasoning (NMR'10)*.

[158] Krümpelmann, Patrick, Tamargo, Luciano H., García, Alejandro J., & Falappa, Marcelo A. 2009. Forwarding Credible Information in Multi-agent Systems. *Pages 41–53 of:* Karagiannis, Dimitris, & Jin, Zhi (eds), *Proceedings of the 3rd International Conference on Knowledge Science, Engineering and Management (KSEM'09)*. Lecture Notes in Computer Science, vol. 5914. Springer.

[159] Krümpelmann, Patrick, Thimm, Matthias, Kern-Isberner, Gabriele, & Fritsch, Regina. 2011. Motivating Agents in Unreliable Environments: A Computational Model. *Pages 65–76 of:* Klügl, Franziska, & Ossowski, Sascha (eds), *Multiagent System Technologies - 9th German Conference, (MATES 2011), Berlin, Germany, October 6-7, 2011. Proceedings*. Lecture Notes in Computer Science, vol. 6973. Springer.

[160] Krümpelmann, Patrick, Thimm, Matthias, Falappa, Marcelo A., Garcia, Alejandro J., Kern-Isberner, Gabriele, & Simari, Guillermo R. 2012. Selective Revision by Deductive Argumentation. *Page 281 of:* Modgil, Sanjay, Oren, Nir, & Toni, Francesca (eds), *Formal Argumentation - First International Workshop on Theory and Application, (TAFA'11), Barcelona, Spain, July 16-17, 2011, Revised Selected Papers*. Lecture Notes in Computer Science, vol. 7132. Springer.

[161] Krümpelmann, Patrick, Janus, Tim, & Kern-Isberner, Gabriele. 2014a. Angerona - A flexible Multiagent Framework for Knowledge-based Agents. *Pages 35–50 of:* Bulling, Nils (ed), *Proceedings of the 12th European Conference on Multi-Agent Systems*. Lecture Notes in Artificial Intelligence, vol. 8953. Springer.

[162] Krümpelmann, Patrick, Janus, Tim, & Kern-Isberner, Gabriele. 2014b. *Angerona - A Multiagent Framework for Logic Based Agents*. Tech. rept. Technische Universität Dortmund, Department of Computer Science.

[163] Kudo, Yasuo, & Murai, Tetsuya. 2004. A Method of Belief Base Revision for Extended Logic Programs Based on State Transition Diagrams. *Pages 1079–1084 of:* Negoita, Mircea, Howlett, Robert, & Jain, Lakhmi (eds), *Knowledge-Based Intelligent Information and Engineering Systems*. Lecture Notes in Computer Science, vol. 3213. Springer Berlin / Heidelberg.

[164] Leite, JoãoAlexandre, Alferes, JoséJúlio, & Pereira, LuísMoniz. 2002. $\mathcal{MINERVA}$ - A Dynamic Logic Programming Agent Architecture. *Pages 141–157 of:* Meyer, John-JulesCh., & Tambe, Milind (eds), *Intelligent Agents VIII*. Lecture Notes in Computer Science, vol. 2333. Springer Berlin Heidelberg.

[165] Leone, Nicola, Pfeifer, Gerald, Faber, Wolfgang, Eiter, Thomas, Gottlob, Georg, Perri, Simona, & Scarcello, Francesco. 2006. The

DLV system for knowledge representation and reasoning. *ACM Transactions on Computational Logic (TOCL)*, **7**(3), 499–562.

[166] Lespérance, Yves, Levesque, Hector J., Lin, Fangzhen, Marcu, Daniel, Reiter, Raymond, & Scherl, Richard B. 1995. Foundations of a Logical Approach to Agent Programming. *Pages 331–346 of:* Wooldridge, Michael, Müller, Jörg P., & Tambe, Milind (eds), *ATAL*. Lecture Notes in Computer Science, vol. 1037. Springer.

[167] Levesque, Hector J., Reiter, Raymond, Lesperance, Yves, Lin, Fangzhen, & Scherl, Richard B. 1997. GOLOG: A Logic Programming Language for Dynamic Domains. *Journal of Logic Programming*, **31**(1-3), 59–83.

[168] Lifschitz, Vladimir, & Turner, Hudson. 1994. Splitting a logic program. *Pages 23–37 of: Proceedings of the 11th international conference on Logic programming*. Cambridge, MA, USA: MIT Press.

[169] Lifschitz, Vladimir, Tang, Lappoon R, & Turner, Hudson. 1999. Nested expressions in logic programs. *Annals of Mathematics and Artificial Intelligence*, **25(3-4)**, 369–389.

[170] Luck, Michael, & d'Inverno, Mark. 1995. A Formal Framework for Agency and Autonomy. *Pages 254–260 of: Proceedings of the 1st International Conference on Multi-Agent Systems (ICMAS'95)*. AAAI Press/MIT Press.

[171] Ma, Jiefei, Russo, Alessandra, Broda, Krysia, & Lupu, Emil. 2011. Multi-agent abductive reasoning with confidentiality. *In: Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011)*. International Foundation for Autonomous Agents and Multiagent Systems.

[172] Makinson, D. 1997. Screened Revision. *Theoria*, **63**(1-2), 14–23.

[173] Makinson, David. 1994. General Patterns in Nonmonotonic Reasoning. *In: Handbook of Logic in Artificial Intelligence and Logic Programming, Vol. Iii*. Clarendon Press.

[174] Mantel, Heiko. 2003. *A uniform framework for the formal specification and verification of information flow security*. Ph.D. thesis, Universität des Saarlandes.

[175] Mayer, Julia M., Schuler, Richard P., & Jones, Quentin. 2012. Towards an understanding of social inference opportunities in social computing. *Pages 239–248 of: Proceedings of the 17th ACM international conference on Supporting group work*. GROUP '12. New York, NY, USA: ACM.

[176] McCullough, Daryl. 1987. Specifications for Multi-Level Security and a Hook-Up Property. *Pages 161–166 of: IEEE Symposium on Security and Privacy*. IEEEE.

[177] McLean, John. 1992. Proving noninterference and functional correctness using traces. *Journal of Computer security*, **1**(1), 37–57.

[178] McLean, John. 1994. A General Theory of Composition for Trace Sets Closed under Selective Interleaving Functions. *In: Proceedings of the 1994 IEEE Symposium on Security and Privacy (SP'94)*. IEEE Computer Society.

[179] Mele, A. R. 2003. *Motivation and Agency*. Oxford University Press.

[180] Meneguzzi, Felipe, & Luck, Michael. 2007. Motivations as an Abstraction of Meta-level Reasoning. *Pages 204–214 of: Proceedings of the 5th international Central and Eastern European conference on Multi-Agent Systems and Applications V (CEEMAS'07)*. Berlin, Heidelberg: Springer-Verlag.

[181] Modgil, S., & Prakken, H. 2011. Revisiting preferences and argumentation. *Pages 1021–1026 of: Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Two*. AAAI Press.

[182] Móra, Michael da Costa, Lopes, José Gabriel Pereira, Vicari, Rosa Maria, & Coelho, Helder. 1999. BDI Models and Systems: Bridging the Gap. *Pages 11–27 of: Proceedings of the 5th International Workshop on Intelligent Agents V, Agent Theories, Architectures, and Languages (ATAL'98)*. ATAL '98. London, UK, UK: Springer-Verlag.

[183] Nebel, Bernhard. 1992. Syntax-based approaches to belief revision. *Pages 52–88 of:* Gärdenfors, P. (ed), *Belief Revision*, vol. 29. Cambridge, UK: Cambridge University Press.

[184] Niemelä, Ilkka, & Simons, Patrik. 1997. Smodels - An Implementation of the Stable Model and Well-Founded Semantics for Normal LP. *Pages 421–430 of: Proceedings of the 4th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'97)*. London, UK: Springer-Verlag.

[185] Nogueira, Monica, Balduccini, Marcello, Gelfond, Michael, Watson, Richard, & Barry, Matthew. 2001. An A-Prolog decision support system for the Space Shuttle. *Pages 169–183 of: Practical Aspects of Declarative Languages*. Springer.

[186] Novák, Peter. 2008. Jazzyk: A Programming Language for Hybrid Agents with Heterogeneous Knowledge Representations. *Pages 72–87 of:* Hindriks, Koen V., Pokahr, Alexander, & Sardiña, Sebastian (eds), *Programming Multi-Agent Systems, 6th International Workshop, ProMAS 2008, Estoril, Portugal, May 13, 2008. Revised Invited and Selected Papers*. Lecture Notes in Computer Science, vol. 5442. Springer.

[187] Osborne, Martin J, & Rubinstein, Ariel. 1994. *A course in game theory*. MIT press.

[188] Osorio, Mauricio, & Cuevas, Victor. 2007. Updates in answer set programming: An approach based on basic structural properties. *Theory and Practice of Logic Programming*, **7**(4), 451–479.

[189] Palamidessi, Catuscia (ed). 2003. *Uniform Equivalence of Logic Programs under the Stable Model Semantics*. Vol. 2916. Springer Berlin / Heidelberg.

[190] Pearl, Judea. 1990. System Z: A natural ordering of defaults with tractable applications to nonmonotonic reasoning. *Pages 121–135 of: Proceedings of the 3rd Conference on Theoretical Aspects of Reasoning about Knowledge*. Morgan Kaufmann Publishers Inc.

[191] Poole, D. 1988. A logical framework for default reasoning. *Artificial Intelligence*, **36**(1), 27–47.

[192] Prakken, Henry. 2010. An abstract framework for argumentation with structured arguments. *Argument and Computation*, **1**(2), 93–124.

[193] R. Biedert, N. Delsaux, T. Lottermann. 2012. *Java Simple Plugin Framework*. http://code.google.com/p/jspf/. [Online; accessed 10-December-2012].

[194] Rao, A. S., & Georgeff, M. P. 1991a. Modeling rational agents within a BDI-architecture. *Pages 473–484 of: Principles of Knowledge Representation and Reasoning. Proceedings of the second International Conference*. San Mateo: Morgan Kaufmann.

[195] Rao, Anand S. 1996a. AgentSpeak(L): BDI Agents Speak Out in a Logical Computable Language. *Pages 42–55 of: Agents Breaking Away*. Springer.

[196] Rao, Anand S. 1996b. Decision Procedures for Propositional Linear-Time Belief-Desire-Intention Logics. *Pages 33–48 of:* Wooldridge, Michael, Müller, Jörg P., & Tambe, Milind (eds), *ATAL*. Lecture Notes in Computer Science, vol. 1037. Springer.

[197] Rao, Anand S., & Georgeff, Michael P. 1991b. Asymmetry Thesis and Side-Effect Problems in Linear-Time and Branching-Time Intention Logics. *Pages 498–505 of:* Myopoulos, John, & Reiter, Ray (eds), *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI-91)*. Sydney, Australia: Morgan Kaufmann.

[198] Rao, Anand S., & Georgeff, Michael P. 1992. An Abstract Architecture for Rational Agents. *Pages 439–449 of:* Nebel, Bernhard, Rich, Charles, & Swartout, William R. (eds), *Proceedings of Knowledge Representation and Reasoning (KR and R-91)*. Morgan Kaufmann.

[199] Rao, Anand S., & Georgeff, Michael P. 1993. A Model-Theoretic Approach to the Verification of Situated Reasoning Systems. *Pages 318–324 of:* Bajcsy, Ruzena (ed), *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*. Chambéry, France: Morgan Kaufmann.

[200] Rao, Anand S., & Georgeff, Michael P. 1995. BDI-agents: from theory to practice. *In: Proceedings of the 1st International Conference on Multiagent Systems (ICMAS'05)*. San Francisco: AAAI Press.

[201] Rao, Anand S., Georgeff, Michael P., & Sonenberg, Elizabeth A. 1992. Social Plans: A Preliminary Report (Abstract). *SIGOIS Bull.*, **13**(3), 10–.

[202] Reiter, Raymond. 1987. *A logic for default reasoning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. Pages 68–93.

[203] Ricca, Francesco, Grasso, Giovanni, Alviano, Mario, Manna, Marco, Lio, Vincenzino, Iiritano, Salvatore, & Leone, Nicola. 2012. Team-building with answer set programming in the Gioia-Tauro seaport. *Theory and Practice of Logic Programming*, **12**(03), 361–381.

[204] Richardson, Matthew, & Domingos, Pedro. 2006. Markov logic networks. *Machine learning*, **62**(1-2), 107–136.

[205] Rödder, Wilhelm. 2000. Conditional logic and the principle of entropy. *Artificial Intelligence*, **117**(1), 83–106.

[206] Rott, Hans. 1995. Just because. Taking belief bases very seriously. *Logic for a Change*, **9**, 106–124.

[207] Rumbaugh, James, Jacobson, Ivar, & Booch, Grady. 2004. *Unified Modeling Language Reference Manual, The*. Pearson Higher Education.

[208] Russell, Stuart Jonathan, & Wefald, Eric. 1991. *Do the right thing: studies in limited rationality*. MIT press.

[209] Sakama, Chiaki, Caminada, Martin, & Herzig, Andreas. 2010. A Logical Account of Lying. *Pages 286–299 of:* Janhunen, Tomi, & Niemelä, Ilkka (eds), *Logics in Artificial Intelligence*. Lecture Notes in Computer Science, vol. 6341. Springer Berlin Heidelberg.

[210] Schaefer, Thomas J. 1976. Complexity of decision problems based on finite two-person perfect-information games. *Pages 41–49 of: Proceedings of the eighth annual ACM symposium on Theory of computing*. STOC '76. New York, NY, USA: ACM.

[211] Schut, Martijn, & Wooldridge, Michael. 2001. Principles of intention reconsideration. *Pages 340–347 of: Proceedings of the fifth international conference on Autonomous agents*. AGENTS '01. New York, NY, USA: ACM.

[212] Searle, John R. 1969. *Speech acts: An essay in the philosophy of language*. Vol. 626. Cambridge university press.

[213] Sichermann, G. L., de Jonge, W., & van de Riet, R. P. 1983. Answering queries without revealing secrets. *ACM Transactions on Database Systems*, **8**, 41–59.

[214] Singh, Munindar P. 1991. A logic of situated know-how. *Pages 343–348 of: Proceedings of the 9th National Conference on Artificial Intelligence (AAAI'91)*.

[215] Singh, Munindar P. 1999. Know-how. *Pages 81–104 of:* Wooldridge, M., & Rao, A. (eds), *Foundations of Rational Agency*. Kluwer Academic, Dordrecht.

[216] Slota, M., & Leite, J. 2010. On Semantic Update Operators for Answer-Set Programs. *Pages 957–962 of:* Coelho, H., & Wooldridge, M. (eds), *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI'10)*. IOS Press.

[217] Slota, Martin, & Leite, João. 2012. Robust Equivalence Models for Semantic Updates of Answer-Set Programs. *In: Proceedings of the 13th International Conference on Principles of Knowledge Representation and Reasoning (KR'12)*. AAAI Press.

[218] Soininen, Timo, & Niemelä, Ilkka. 1998. Developing a declarative rule language for applications in product configuration. *Pages 305–319 of: practical aspects of declarative languages*. Springer.

[219] Son, Tran Cao, & Sakama, Chiaki. 2010. Reasoning and Planning with Cooperative Actions for Multiagents Using Answer Set Programming. *Pages 208–227 of: Proceedings of the 7th International Conference on Declarative Agent Languages and Technologies*. DALT'09. Berlin, Heidelberg: Springer-Verlag.

[220] Spohn, W. 1988. Ordinal conditional functions: a dynamic theory of epistemic states. *Pages 105–134 of:* Harper, W.L., & Skyrms, B. (eds), *Causation in Decision, Belief Change, and Statistics*, vol. 2. Kluwer Academic Publishers.

[221] Sutherland, David. 1986. A model of information. *Pages 175–183 of: Proceedings of the 9th National Computer Security Conference*. DTIC Document.

[222] Tadros, Cornelia. 2014. *Belief Change Operations under Confidentiality Requirements in Multiagent Systems*. Ph.D. thesis, Technische Universität Dortmund.

[223] Tamargo, Luciano H., Thimm, Matthias, Krümpelmann, Patrick, Garcia, Alejandro J., Falappa, Marcelo A., Simari, Guillermo R., & Kern-Isberner, Gabriele. 2013. Credibility-based

Selective Revision by Deductive Argumentation in Multi-agent Systems. *In:* E. Ferme, D. Gabbay, G.R. Simari (ed), *Trends in Belief Revision and Argumentation Dynamics*. College Publications.

[224] Tao, Jia, Slutzki, Giora, & Honavar, Vasant. 2014. A Conceptual Framework for Secrecy-preserving Reasoning in Knowledge Bases. *ACM Trans. Comput. Logic*, **16**(1), 3:1–3:32.

[225] Thimm, Matthias. 2012. A Probabilistic Semantics for abstract Argumentation. *Pages 750–755 of: Proceedings of the 20th European Conference on Artificial Intelligence (ECAI 2012)*. IOS Press.

[226] Thimm, Matthias. 2013. Dynamic Preference Aggregation under Preference Changes. *In: Proceedings of the Fourth Workshop on Dynamics of Knowledge and Belief (DKB'13)*.

[227] Thimm, Matthias. 2014. Tweety - A Comprehensive Collection of Java Libraries for Logical Aspects of Artificial Intelligence and Knowledge Representation. *In: Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR'14)*. AAAI Press.

[228] Thimm, Matthias, & García, Alejandro J. 2010. Classification and strategical issues of argumentation games on structured argumentation frameworks. *Pages 1247–1254 of: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems.

[229] Thimm, Matthias, & Krümpelmann, Patrick. 2009. Know-How for Motivated BDI Agents (Extended Abstract). *In:* Decker, Sichman, Sierra, & Castelfranchi (eds), *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'09)*. International Foundation for Autonomous Agents and Multiagent Systems.

[230] Van der Hoek, W., & Wooldridge, M. 2003. Towards a logic of rational agency. *Logic Journal of IGPL*, **11**(2), 135–159.

[231] van der Hoek, Wiebe. 2004. *Epistemic logic for AI and computer science*. Vol. 41. Cambridge University Press.

[232] van der Torre, Leendert. 2012. Logics for Security and Privacy. *Pages 1–7 of:* Cuppens-Boulahia, Nora, Cuppens, Frédéric, & Garcia-Alfaro, Joaquin (eds), *Data and Applications Security and Privacy XXVI*. Lecture Notes in Computer Science, vol. 7371. Springer Berlin / Heidelberg.

[233] van Ditmarsch, Hans P, & Kooi, Barteld P. 2006. Semantic results for ontic and epistemic change. *arXiv preprint cs/0610093*.

[234] van Harmelen, Frank, van Harmelen, Frank, Lifschitz, Vladimir, & Porter, Bruce. 2007. *Handbook of Knowledge Representation*. San Diego, USA: Elsevier Science.

[235] van Rooy, Robert. 2004. Utility, Informativity and Protocols. *Journal of Philosophical Logic*, **33**(4), 389–419.

[236] Vladimir, Lifschitz. 2002. Answer set programming and plan generation. *Artificial Intelligence*, **138**(1–2), 39–54.

[237] Šefránek, Ján, & Siika, Jozef. 2006. Irrelevant Updates of Nonmonotonic Knowledge Bases. *Pages 771–772 of: Proceedings of the 17th European Conference on Artificial Intelligence (ECAI'06)*. IOS Press.

[238] Wallace, Richard J., & Freuder, Eugene C. 2005. Constraint-based reasoning and privacy/efficiency tradeoffs in multi-agent problem solving. *Artificial Intelligence*, **161**(1–2), 209 – 227.

[239] Walsh, Toby. 2007. Representing and reasoning with preferences. *AI Magazine*, **28**(4), 59.

[240] Wassermann, Renata. 2001. *Ressource-Bounded Belief Revision*. Ph.D. thesis, ILLC, University of Amsterdam.

[241] Weiss, G. (ed). 1999. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. Cambridge, MA, USA: MIT Press.

[242] Weyns, Danny, Van Dyke Parunak, H., Michel, Fabien, Holvoet, Tom, & Ferber, Jacques. 2005. Environments for Multiagent Systems State-of-the-art and Research Challenges. *Pages 1–47 of: Proceedings of the First International Conference on Environments for Multi-Agent Systems*. E4MAS'04. Berlin, Heidelberg: Springer-Verlag.

[243] Weyns, Danny, Vizzari, Giuseppe, & Holvoet, Tom. 2006. Environments for Situated Multi-agent Systems: Beyond Infrastructure. *Pages 1–17 of:* Weyns, Danny, Dyke Parunak, H., & Michel, Fabien (eds), *Environments for Multi-Agent Systems II*. Lecture Notes in Computer Science, vol. 3830. Springer Berlin Heidelberg.

[244] Wierzoch, Pia. 2014. *Aktualisierung von Wissen unter Vertraulichkeitsanforderungen*. Bachelor's Thesis, Computer Science, Chair 6 - Information Systems and Security, Technische Universität Dortmund.

[245] Witteveen, Gees, & van der Hoek, Wiebe. 1997. A general framework for revising nonmonotonic theories. *Pages 258–272 of:* Dix, Jürgen, Furbach, Ulrich, & Nerode, Anil (eds), *Logic Programming And Nonmonotonic Reasoning*. Lecture Notes in Computer Science, vol. 1265. Springer Berlin / Heidelberg.

[246] Woltran, Stefan. 2008. A common view on strong, uniform, and other notions of equivalence in answer-set programming. *Theory and Practice of Logic Programming*, **8**(2), 217–234.

[247] Wooldridge, Michael. 1999. Intelligent Agents. *Chap. Intelligent Agents of:* Weiss, G. (ed), *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. Cambridge, MA, USA: MIT Press.

[248] Wooldridge, Michael J. 2009. *An Introduction to MultiAgent Systems*. 2 edn. Wiley.

[249] Yager, Ronald R. 2008. Prioritized aggregation operators. *International Journal of Approximate Reasoning*, **48**(1), 263 – 274. Special Section: Perception Based Data Mining and Decision Support Systems.

[250] Yager, Ronald R. 2010. Lexicographic ordinal OWA aggregation of multiple criteria. *Information Fusion*, **11**(4), 374 – 380.

[251] Zhang, Dongmo, Foo, Norman, Meyer, Thomas, & Kwok, Rex. 2004. Negotiation as mutual belief revision. *Pages 317–322 of: Proceedings of the 19th national conference on Artifical intelligence (AAAI'04)*. AAAI'04. AAAI Press.

[252] Zhang, Yan. 2006. Logic program-based updates. *ACM Transactions on Computational Logic (TOCL)*, **7**(3), 421–472.

LIST OF IMPORTANT SYMBOLS

Ag	Set of agents 38
$\mathcal{K}$	Epistemic state of an agent 29
$\mathcal{K}^0$	Initial epistemic state 41
Λ	Set of initial epistemic states 119
Ω	Set of epistemic states 112
ξ	Functional component of an agent 41
$\mathfrak{F}$	Set of functional components 57
a	Sction 44
ϵ	Symbol representing the empty action 38
Act	Set of actions 20
$\mathsf{Act}_{\mathsf{inf}}$	Set of inform actions 135
p	Percept 19
p_ϵ	Symbol representing the empty percept 38
Per	Set of percepts 19
ι	Speech act 75
Σ	Set of speech acts 39
$\hat{\phi}$	ϕ has an undetermined truth value 125
$\mathfrak{B}$	Set of beliefs 19
$\mathcal{L}_{\mathfrak{B}}$	Power set of the set of possible beliefs 19
D	Desire 51
$\mathfrak{D}$	Set of desires 19
$\mathcal{L}_{\mathfrak{D}}$	Power set of the set of possible desires 19
I	Intention 52
$\mathfrak{I}$	Set of desires 19
AtInt	Set of atomic intentions 57
$\mathcal{L}_{\mathfrak{I}}$	Power set of the set of possible intentions 19
kh	Know-how statement 52
$\mathfrak{KH}$	Set of know-how statements 52
$\mathcal{L}_{\mathfrak{KH}}$	Power set of the set of possible know-how statements 54
$\mathcal{L}_{\mathfrak{KH},b}$	Power set of the set of possible basic know-how statements 53
$\mathfrak{M}$	Set of motives 51
μ	Motivation value 51

$\mathcal{L}_{\mathfrak{M}}$	Power set of the set of possible motives 54
$\mathcal{L}_{\mathfrak{M},b}$	Power set of the set of possible basic motives 51
$\mathcal{X}$	Arbitrary agent 38
$\mathcal{A}$	Attacking agent 108
$\mathcal{D}$	Defending agent 108
$\mathcal{T}$	Setting 119
At	Set of all atoms 21
Lit	Set of literals 26
$\mathsf{U}(\mathsf{Lit})$	Set of grounded literals 26
$\mathcal{L}_{\mathsf{Base}}$	Common communication language 38
$\mathcal{L}^{\mathsf{asp}}_{\mathsf{At}}$	Language of extended logic programs 26
$\mathcal{L}^{\mathsf{prop}}_{\mathsf{At}}$	Language of propositional logic 21
$\mathcal{B}$	Belief base 30
$\mathcal{L}_{\mathsf{ES}}$	Language of an epistemic state 41
$\mathcal{L}_{\mathsf{E}}$	Epistemic component language 39
$\mathcal{L}_{\mathsf{BS}}$	Belief set language 40
$\mathsf{V}_{\mathcal{A}}$	View on the world view of agent $\mathcal{A}$ 109
$\mathsf{F}(\mathcal{S}(\mathcal{K}))$	Set of secret formulae 126
$\mathsf{V}_{\mathcal{W}}$	World view 109
$\mathcal{W}$	Set of world views 136
$\mathcal{S}$	Set of Secrets 109
$\mathcal{L}_{\mathcal{S}}$	Language of secrets 109
e	Extension 64
ext	Extension family function 64
$\mathcal{E}$	Set of extension family functions 66
comb	Combination operator for extensions 66
Γ	Set of combination operator for extensions 66
AS	Set of answer sets for a given program 27
body	Body of a rule 26
not	Default negation operator 26
$\mathsf{Lit}_{\mathsf{not}}$	Set of default negated literals 26
$\bullet$	Variable for a notion of ASP equivalence 89
head	Head of a rule 89
$\mathfrak{L}$	Set of literals and default negated literals 26

κ	Accumulator 24
γ	Categorizer 24
τ	Argument tree 23
$\mathcal{F}(At)$	All argument trees over a set of atoms At 24
$\mathcal{T}^-$	Contra argument trees 24
$\mathcal{T}^+$	Pro argument trees 24

Bel	Belief operator 40
$\mathsf{Bel}_{\mathcal{X}}$	Belief operator of agent $\mathcal{X}$ 40
$\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{cred}}$	Credulous ASP belief operator 66
$\mathsf{Bel}^{\mathsf{asp}}_{\mathsf{skep}}$	Skeptical ASP belief operator 66
Ξ	Set of belief operators 61
BS	Belief set 40
E	Epistemic component 39

$!_R$	Screened consolidation operator 94
$-$	Contraction operator 31
$\hat{+}$	World view append operator 136
$\oplus$	Agent view append operator 136
$*$	Prioritized change operator for epistemic components 28
$\circ$	Change operator for epistemic states and percepts 42
$\circ^a$	Change operator for epistemic states and actions 42
$\circ_E$	Change operator for epistemic components and percepts 75
$+$	Non-closing expansion operator 30
$\dot{+}$	Closing expansion operator 32
$*_{\mathcal{S}}$	Change operator for secrets 115
f	Selection function 74
$\star_{f_{\mathcal{B}}}$	Selective revision operator 75
t	Interpretation function for speech-acts 74

σ	General choice function 59
cl	Classification function 140
$\equiv^p$	Classical propositional equivalence 22
$\cong^p$	Pair-wise propositional equivalence 22
$\equiv_{SE}$	Strong equivalence for ASP 27
$\equiv_{UE}$	Uniform equivalence for ASP 27
$\equiv_{SE}$	Answer set equivalence for ASP 27

$\equiv_{SE}$	Syntactic equivalence for ASP 89
$\cdot$	Function composition 48
$\vdash$	Classical propositional entailment 21
α	Function returning the action for an atomic intention 57
$\mathfrak{P}$	Set of multi sets 22
$\mathcal{P}(S)$	Power set of the set S 18
$\mathcal{P}_{\mathsf{fin}}(S)$	Set of all finite subsets of set S 144
$s_{\mathcal{X}}$	Strategy of agent $\mathcal{X}$ 122
$\subset_{\mathsf{fin}}$	Finite subset 144
$Cn^{\mathsf{prop}}(\cdot)$	Deductive consequence operator 21
Mod	Model operator 64
r	Ratio of models 64
$\preceq_{\mathsf{bel}}$	Preference relation on belief operators 61
f_{agg}	Aggregation function for preferences 131
f_{agg}^{lex}	Lexicographic aggregation function for preferences 131
$\preceq_{(\mathcal{K},\xi)}$	Preference relation on options for action 58
$\preceq_{\mathcal{K}}^{a}$	Preference relation on options with respect to informativity 133
$\preceq_{(\mathcal{K},\xi)}^{s}$	Preference relation on options with respect to secrecy 130
$\prec_{vio}$	Preference relation on actions wrt. violation of secrets 142
vio	Violation set for a world view 135
$\mathsf{vioAfter}$	Violation set for a set of world views 135
sr	Secrecy reasoner 144

INDEX