UNIVERSITÄT DORTMUND
Fachbereich Informatik

Lehrstuhl VIII
Künstliche Intelligenz

CORA — A Knowledge–Based System for the Analysis of Case–Control Studies

LS–8 Report 21

Ursula Robers

Dortmund, May 19, 1996

Universität Dortmund
Fachbereich Informatik

University of Dortmund
Computer Science Department

Forschungsberichte des Lehrstuhls VIII (KI)
Fachbereich Informatik
der Universität Dortmund

ISSN 0943-4135

Anforderungen an:

Universität Dortmund
Fachbereich Informatik
Lehrstuhl VIII
D-44221 Dortmund

Research Reports of the unit no. VIII (AI)
Computer Science Department
of the University of Dortmund

ISSN 0943-4135

Requests to:

University of Dortmund
Fachbereich Informatik
Lehrstuhl VIII
D-44221 Dortmund

e-mail: reports@ls8.informatik.uni-dortmund.de
ftp: ftp-ai.informatik.uni-dortmund.de:pub/Reports
www: http://www-ai.informatik.uni-dortmund.de/ls8-reports.html

CORA — A Knowledge–Based System for the Analysis of Case–Control Studies

LS–8 Report 21

Ursula Robers

Dortmund, May 19, 1996

Universität Dortmund
Fachbereich Informatik

Abstract

Carrying out a statistical analysis of empirical data the researcher is typically concerned with the problem of choosing an appropriate statistical technique from a large number of competing methods. Most of the common statistical software only allows to analyse the data by applying certain methods that are implemented in this software without giving any support to the researcher with respect to the adequacy of a method for a particular data set.

This paper outlines the main features of the computer system CORA which provides a statistical analysis of stratified contingency tables and additionally supports the researcher at the different steps of this analysis. The support given by the system consists of two different aspects. On the one hand the help system of CORA contains general information on the implemented statistical methods which can be obtained on request by the user. On the other hand an advice tool recommends an adequate statistical method. This advice depends on the actual empirical case–control data that the user wants to analyse. To build up the advice tool a set of rules being discovered by machine learning is integrated into the system CORA. Simulation studies that investigate the finite–sample behaviour of estimators serve as knowledge sources for this discovery process.

The presented way of constructing such systems can be seen as a general approach which is applicable in many fields of research: with the means of machine learning you can discover knowledge in simulation studies and then integrate this knowledge into a system which supports the user by recommendations (guidelines) how to proceed.

1 Introduction

The aim of epidemiological case–control studies is to investigate possible associations between a potential risk factor and a certain disease. Carrying out such a study the numbers of persons are recorded having the disease or not and being exposed or not. For further statistical analyses, these four absolute frequencies are usually ordered in a 2×2 table. A quantifying statistical measure for the investigated association is the so–called odds ratio. It can be interpreted as the factor by which the risk of disease increases if a person is exposed to the risk factor of interest. Let us denote the probability for a case being exposed by p_1 and for a control p_0. Then, the odds ratio is defined as $\frac{p_1(1-p_0)}{p_0(1-p_1)}$. Typically, there are additional confounding variables being associated with the risk factor and also having an influence on the disease of interest. These confounders have to be controlled to ensure that the odds ratio reflects the only influence of the risk factor. One possibility for controlling such confounders consists in a stratification of the data according to the categories of the confounder where a 2×2 table is contructed for each category. If the confounder is controlled, i.e. all individual odds ratios are equal (homogeneity), a so–called common odds ratio ψ is to be estimated from the data. A great variety of estimators of the common odds ratio with different statistical properties is available, see e.g. [1], [2], [3], [4] and [5]. The behaviour of the estimators heavily depends on the characteristics of the case–control data to analyse.

The researcher is thus confronted with the problem of choosing an adequate estimator out of the large pool of competing techniques. To cope with this problem, the computer system CORA (**C**ombined **O**dds **R**atio **A**nalysis) was developed which assists the researcher in analysing the data. This support is achieved by two system components: an enlarged hypertext help system and a knowledge–based advice tool. The help system, which offers information about CORA and how to use it, is extended by some general aspects concerning a stratified contingency table analysis as well as the statistical properties of the implemented estimators and some other statistical methods used in the analysis. The advice tool suggests an appropriate estimator for the common odds ratio. This recommendation depends on the characteristics of the underlying case–control data entered into the system. In contrast, the information given by the help system, although it is context–sensitive, does not consider the actual characteristics of the data to analyse.

In order to build up the advice tool we have applied machine learning. The finite–sample behaviour of statistical methods can be determined by simulation studies, which are valuable, if an analytical investigation would be too complicated or even impossible. From the results of these studies, represented in a knowledge base, a characterization of estimators can be learned. The resulting set of rules then underlies the advice tool.

Knowledge–based techniques have been widely used when developing intelligent statistical software. But most of the statistical expert systems that have been worked out since the mid 80th fell back to a transfer view of knowledge acquisition. This transfer view is based on the assumption that with the help of tools — namely expert system shells — knowledge can be easily carried over from the expert to the system. Almost all of these systems failed in practice. This false assessment of the knowledge acquisition process may be one reason for their failure.

In [6] Morik proposed to focus on modelling the knowledge and has created a different view of knowledge acquisition. With the help of tools assisting this modelling process knowledge can be discovered and revised. Thus it can be made available for knowledge–based systems (KBS) in a more appropriate way. Hence, KBS can now be developed that essentially differ from the first generation of expert systems regarding the quality of the implemented knowledge.

The outline of this paper is as follows. First we describe some basic aspects of the system CORA: the domain of the system, the expertise required by the user as well as the overall system architecture of CORA. In the third section we take a closer look at the advice tool. We present the particular steps of the above mentioned modelling process needed to build up this tool. Section 4 illustrates the implementation of CORA, that is the design of the advice tool, which uses the results of the modelling process, the design of the analysis tool, which involves all statistical procedures to handle the data, and the help system. Some screendumps are presented to show the layout of the user interface. In Section 5 we discuss our approach and the lessons learned while developing CORA. Besides we suggest some main directions for future research to improve and to enlarge our approach.

2 Some basic features of CORA

In this section we describe the domain of the system CORA. For developing an appropriate system it is also necessary to take a look at the potential users and their statistical knowledge. Finally, we present the overall architecture of the system.

CORA is restricted to a rather small domain, i.e. to the analysis of stratified 2×2 contingency tables for the purpose of evaluating case–control studies as outlined in the indroduction. However, the domain is hard to handle for a researcher, because different kinds of expertise are required. He/she needs the medical and epidemiological background to design the study and to collect the data, but additionally statistical knowledge is needed especially while analysing the data. In practice, the co–operation between physicians or epidemiologists on the one hand and statisticians on the other hand is often difficult. Sometimes physicians totally disclaim the help of statisticians, because they themselves are equipped with statistical knowledge. But in case this statistical qualification is not sufficient severe problems may arise.

In a contingency table analysis there are various decisions to be made by the researcher. Here, the choice of an estimator for the common odds ratio is of particular importance. For this, the researcher should not only know the different types of estimators, but also their asymptotical and finite properties. Especially, the latter strongly depends on the characteristics of the data he/she wants to analyse. If the researcher does not know the relationships between the characteristics of the data and the properties of the estimators, unfavourable selections may be the consequence. Here, CORA should assist this selection process by providing the user with this kind of expertise. We integrated this expertise into the help system and especially into the advice tool of CORA such that the system can give a recommendation regarding the choice of an estimator.

As mentioned before, the advice tool contains a set of learned rules. The system examines the characteristics of the present data and, if possible, fires a suitable rule that supposes an estimator. That means, there is no complex inference process, but a single stage decision process: only one rule is applied. We should emphasize that this tool has only an advisory capacity, i.e. the user of the system is not restricted to follow the recommendation of the tool, the final selection is the responsibility of the user.

We have developed a uniform user–friendly graphical interface for all components of CORA. The statistical procedures are also part of the system: they form the analysis tool. Thus, CORA is not an intelligent interface for an existing statistical software package (a so–called front–end) but a stand alone system. The lower part of Figure 1 illustrates the overall system architecture of CORA.

3 Knowledge acquisition: a modelling process

The approach presented here emphasizes the process of modelling the expertise. The simulation studies as well as the experts themselves serve as knowledge sources for this process. The acquisition of the expertise is supported by MOBAL (see [7]). This system combines knowledge acquisition and machine learning, hence additional knowledge can be derived. First, we have to build up a model, the knowledge base, which represents the results of the simulation studies and thus makes them available for MOBAL. From this model represented in the knowledge base a characterization of the examined point estimators can be learned. This characterization, consisting of a set of rules, relates the characteristics of the data to the behaviour of the estimates. The modelling process and the integration of its results into the system architecture of CORA is depicted in Figure 1. Let us stress that the main goal of using machine learning is not to objectify the knowledge acquisition process, but it is seen as a chance to discover new knowledge and thus to improve the knowledge base.

A closer look at the cyclical modelling process reveals different steps. It starts with outlining a framework for the model. In this framework, the relevant characteristics of the data, i.e. the parameter constellations, and the criteria for the assessment of the estimators have to be determined and to be classified. Such a classification is necessary since it would not be useful to look for rules which are only applicable in very special situations. After evaluating the model built up so far, possible revisions can be made.

From the classified assessment criteria a suitability of an estimator in a certain parameter constellation can be derived. In addition to this suitability, we ascertain a ranking for each estimator with respect to the assessment criteria. Based on this ranking, it is possible to state which estimator is best in a given constellation. Thus it can later be recommended for a data situation with similar characteristics. According to the suitability of the estimators, the recommendations can be classified. Such a classification is necessary since there may be for instance data constellations in which even the best estimator behaves poorly.

The next step of the modelling process concerns the representation of the model. Here, MOBAL provides a first order logic, strictly speaking a function free Horn clause logic. Thus, the items in the knowledge base are mainly facts and rules, apart from some other representational structures like rule models (see Section 3.3) as well as sorts and a topology of predicates (see [7]). Because the latter two are not relevant for the representation of our model they are not further discussed.

The steps of the modelling process are described below in more detail.

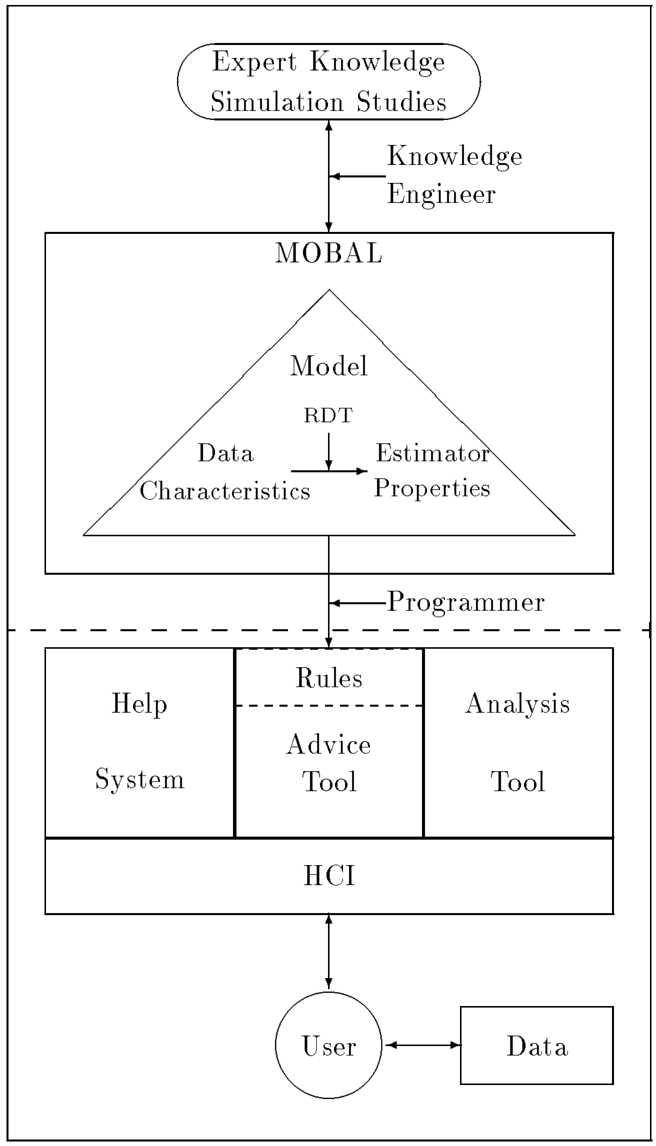

Figure 1: Integration of the modelling process into the system architecture of CORA

3.1 Modelling the domain knowledge

Monte–Carlo studies are carried out to investigate and compare the performance of some estimators. In the simulation study to be evaluated (see e.g. [8]), the following six estimators of the common odds ratio have been examined: the Mantel–Haenszel estimator $\hat{\psi}_{MH}$, the Woolf estimator $\hat{\psi}_W$, and corresponding jackknifed versions, that is the jackknifed (type I) logarithm of the Mantel–Haenszel estimator $J^I_{\ln(MH)}$, the jackknifed (type I) Woolf estimator J^I_W, the jackknifed (type II) Mantel–Haenszel estimator J^{II}_{MH}, and the jackknifed (type II) logarithm of the Mantel–Haenszel estimator $J^{II}_{\ln(MH)}$. For a description of the different jackknife approaches see for instance [9].

For the design of the simulation study values for the involved parameters have to be fixed such as the number of tables, the number of cases and controls, and the probability for a control to be exposed. Since we assumed homogeneity, we only have to fix a value for the common odds ratio. This is of special importance because the difference between this value and the estimate obtained from the simulated data can be used to rank the estimators.

Each choice for the values of the parameters characterizes a certain constellation. From these characteristics, others can easily be derived, as for instance the ratio of cases and controls or the differences between the extreme values of the exposure probabilities in a certain parameter constellation. Note that most of the mentioned characteristics refer to the single strata. That means for instance there are ten different numbers of cases in a parameter constellation with ten contingency tables. Thus, the description of the data characteristics including all tables is very complex. To reduce this complexity, we calculated averages of the values of the single strata. As additional characteristic, we calculated the well–known Gini ratio. This measures how balanced the characteristics are across the strata.

As essential criteria for the assessment of the estimators we calculated the means of the estimated biases and mean squared errors (MSE) from 1000 simulation runs for each parameter constellation. The bias measures the deviation of the estimate from the true parameter value, whereas the MSE can be regarded as a measure for the variability. Based on the bias and the MSE a ranking of the estimators can be given where these two criteria have to be combined appropriately. This aspect is addressed in the following section.

3.2 Classification of the data properties and the assessment criteria

The rules to be learned for characterizing the estimators should not be too detailed as for instance:

> IF the true, but unknown common odds ratio is 3.5 and there are 30 cases,
> THEN estimator X has a bias of 0.034.

This means, we should use qualitative criteria instead of quantitative measures. This can be realized by dividing the initial criteria into appropriate categories. Fixing the bounds of the categories was the most difficult but also one of the most important tasks in modelling this domain. The definition of the categories is arbitrary, but of course the chosen categories strongly influence the rules, which are discovered, as well as the complexity of the knowledge base.

With the categories presented in Table 1, the above mentioned rule is subsumed by the following one:

IF the common odds ratio is moderate and there are many cases,
THEN estimator X has small bias.

The conclusion contained in this rule, however, is not a recommendation. For obtaining the recommendations, we have to come back to the ranking and to the suitability of each estimator. The ranking is based on an appropriate combination of the ranks of an estimator regarding bias and MSE. The suitability is derived by combining the two categorized criteria bias and MSE. (For clarity, we omit the details of these approaches for combining bias and MSE.) Then, we rate the ranking with respect to the suitability. Hence, if estimator Y is placed first according to its ranking and if in addition its suitability is very good (i.e. very small values for bias and MSE), the above rule could be formulated like as follows:

IF the common odds ratio is moderate and there are many cases,
THEN we recommend estimator Y which is supposed to perform very well for
data with the above characteristics.

3.3 Representation of the model

3.3.1 Knowledge representation in MOBAL

The system MOBAL (cf. [7]) is an environment for building up, inspecting and changing a knowledge base. The items in the knowledge base are represented within a restricted higher–order predicate logic. The domain knowledge consists mainly of

- **facts**, expressing relations, properties and concept membership of objects. They are represented as function–free literals without variables:

$$pred(Term_1, Term_2, ..., Term_n).$$

- **rules**, expressing inferential relations between predicates, necessary and sufficient conditions of concepts, hierarchies of properties. They are represented as Horn clauses in which the premises and the conclusion literal may be negated:

$$pred_1(Term_1^{(1)}, \ldots, Term_{n_1}^{(1)}) \,\&\ldots\&\, pred_m(Term_{p_1}^{(m)}, \ldots, Term_{p_m}^{(m)})$$
$$\rightarrow pred_{concl}(Term_1^{(concl)}, \ldots, Term_{q_{concl}}^{(concl)}).$$

  By forward chaining, new facts can be inferred from rules.

- **rule models** (metapredicates), expressing the structure of the rules to be learned. A rule model is a rule in which predicate variables are used instead of actual predicates of an application domain. A predicate variable can be instantiated by a predicate symbol of the same arity. A full instantiated rule model, where all predicate variables have been replaced by actual predicates, is a rule.

Further, the domain knowledge can include some other items like a topology of predicates and sorts, which are not considered here in detail.

number of tables	**S**	**odds ratio**	ψ
small	$S < 5$	exactly 1	$\psi = 1$
moderate	$5 \leq S < 10$	small	$1 < \psi \leq 2$
large	$10 \leq S < 50$	moderate	$2 < \psi \leq 7$
very large	$50 \leq S$	large	$7 < \psi$
number of cases	**CA**	**number of controls**	**CO**
small	$CA \leq 5$	small	$CO \leq 5$
moderate	$5 < CA \leq 20$	moderate	$5 < CO \leq 20$
large	$20 < CA \leq 100$	large	$20 < CO \leq 100$
very large	$100 < CA$	very large	$100 < CO$
ratio CO/CA	**R**	**Gini ratio**	**GR**
balanced	$R \leq 1.25$	balanced	$GR = 0$
medium balanced	$1.25 < R \leq 3$	medium balanced	$1 < GR \leq 0.5$
unbalanced	$3 < R$	unbalanced	$0.5 < GR \leq 1$
probability of a control to be exposed	**P**	**differences of the probabilities of a control to be exposed**	**D**
low	$P \leq 0.3$	small	$D \leq 0.2$
centered	$0.3 < P \leq 0.7$	large	$0.2 < D$
high	$0.7 < P$		
bias	**B**	**MSE**	**M**
very small	$B < 0.005$	very small MSE	$M < 0.01$
small	$0.005 \leq B < 0.05$	small MSE	$0.01 \leq M < 0.1$
moderate	$0.05 \leq B < 0.5$	moderate MSE	$0.1 \leq M < 1$
large	$0.5 \leq B$	large MSE	$1 \leq M$

Table 1: Categories for the data characteristics and the assessment criteria

3.3.2 Domain model

The characteristics of the data as well as the assessment of the estimators are represented as facts. The following example describes the properties of the first parameter constellation. It contains 18 facts with two or three arguments. The third place is necessary for those data characteristics varying across the strata.

```
oddsratio(sit_1,1.7).
number_of_strata(sit_1,2).
diff_prob(sit_1,0.1).
gini_ratio_cases(sit_1,1).
gini_ratio_ratio(sit_1,0).
gini_ratio_prob(sit_1,1).
number_of_cases(sit_1,1,20).
number_of_cases(sit_1,2,30).
number_of_controls(sit_1,1,60).
number_of_controls(sit_1,2,90).
prob(sit_1,1,0.2).
prob(sit_1,2,0.3).
ratio_cc(sit_1,1,3).
ratio_cc(sit_1,2,3).
mean_cases(sit_1,25).
mean_controls(sit_1,75).
mean_prob(sit_1,0.25).
mean_ratio(sit_1,3).
```

`oddsratio, number_of_strata, diff_prob,` ... are predicates concerning the data characteristics. The arguments `sit_1, sit_2`, ... label the parameter constellation of the simulation study. They are the first arguments in all predicates. `1.7, 1.0, 2,` ... are the arguments in the last place. They denote the values for the corresponding data characteristics, e.g. the common odds ratio or the number of strata. The arguments `1, 2, 3,` ... in the second place of the three–place predicates specify to which strata the corresponding values for the data characteristics belong.

The following four facts characterize the assessment of the estimators. As above the arguments in first place label the parameter constellation. The second place marks the names of the estimators, e.g. the abbreviation `mh` for the Mantel–Haenszel and `bl` for the Breslow–Liang estimator. The last place marks the values for the mean squared error (mse) and the bias.

```
mse(sit_1,mh,1.12436).
mse(sit_1,bl,1.34329).
bias(sit_1,mh,0.54718).
bias(sit_1,bl,0.23197).
```

For all data characteristics and both assessment criteria, the classification into categories is then achieved by rules like the following one:

```
number_of_strata(S,NS) & gt(NS,4) & le(NS,10) -->
moderate_number_of_strata(S).
```

The above mentioned ranking and the assessed suitabilities are derived and represented by similar rules. For further information concerning these rules see [10].

If we know the recommendations and the categorized suitabilities then we can assess the recommendations according to the suitabilities with rules like the following ones:

```
recommendation(S,E) & good_suitability(S,E) --> good_recommendation(S,E).
recommendation(S,E) & bad_suitability(S,E) --> bad_recommendation(S,E).
```

3.4 Learning the recommendation rules

3.4.1 RDT

The rule discorvery tool RDT (see [11]), which is included in MOBAL, helps the user to find regularities in facts. It is a model–based learning algorithm that induces rules from facts. New facts can be derived using the learned rules. The necessary input to RDT are facts and rule models (metapredicates). RDT defines a hypothesis subspace that is actually searched via a set of explicitly spelled out hypothesis templates, the rule models. Thus the hypothesis space consists of the set of all possible instantiations of rule models with domain predicates. For efficiently searching in this hypothesis space a generalization relation on the set of rule models is defined by suitably extending the Θ–subsumption for clauses (see [12]) or the generalized Θ–subsumption (see [13]), respectively. According to Buntine, a clause C Θ–subsumes a clause C' ($C \geq_\Theta C'$), if the more general clause C can be converted to the clause C' by repeatedly turning variables to constants or other terms, adding atoms to the body, or partially evaluating the body of C by resolving some clauses in the background knowledge.

This leads to the following definition of the generality relationship $\geq_{RS}$ between rule models:

> A rule model R is more general than R' ($R \geq_{RS}$), iff there exists a substitution σ applied to term variables, and a substitution Σ applied to predicate variables that does not unify different predicate variables such that $R\sigma\Sigma \subseteq R'$. The substitution σ turns term variables to constants or other terms and the substitution Σ renames or instantiates the predicate variables.

RDT searches this partial ordering top–down from the most general to the more specific hypotheses. Hypotheses are computed by instantiating the predicate variables of the rule models with predicate symbols. Then, these hypotheses are tested. There are three possible results for this test:

1. the hypothesis is too general, i.e. it covers too many negative or unknown instances,

2. the hypothesis is accepted, or

3. the hypothesis is too special, i.e. it covers too few positive instances.

A breadth first search strategy is used and those hypotheses that have already been accepted or pruned as too special are remembered to avoid exploring their specializations. The specializations of the former, i.e. accepted hypotheses, are disregarded because they are redundant, the specialisations of the latter are pruned because they never can be confirmed. Only in the case of a hypothesis being too general, the search is continued.

The premises of a rule model are incrementally instantiated. An order of the premises allows to further prune the search of hypotheses also within a single rule model. This order must take into account the bindings of the variables of the rule model. Based on the connection of a variable to the conclusion, a measure for the distance of this variable from the conclusion is defined. This measure is then used to determine the premise order. The connection of a variable X to the conclusion via the relation chain $rc(X)$ is defined as follows:

> A variable X occurring in the conclusion of a rule model is connected via the empty relation chain ($rc(X) = \emptyset$).
>
> A variable X_i ($1 \leq i \leq n$) occurring in a premise $R(X_1, X_2, \ldots, X_n)$ is connected via the relation chain $rc(X_i) = R \circ rc(X_j)$, iff a variable X_j ($1 \leq j \leq n$) $n, i \neq j$ of $R(X_1, X_2, \ldots, X_n)$ is connected via the relation chain $rc(X_j)$.

A variable can have more than one relation chain, but a rule model which contains an unconnected variable is not allowed. The distance of a variable X, denoted by $\delta(X)$, is then defined as the length of the minimal relation chain connecting it to the conclusion. Thus the order of premises can be defined as follows:

$$P \leq_P P', \text{iff } \min(\{\delta(X) | X \text{occurring in} P\}) \leq \min(\{\delta(X) | X \text{occurring in} P'\}).$$

While instantiating the premises of the rule schema with respect to this order we instantiate P before P', if $P \leq_P P'$. Then we can test all partially instantiated hypotheses in the same way as we test the fully instantiated rule model, if we drop the uninstantiated premises.

The threshold for too few instances and thus for pruning the search is computed from a user–specified acceptance criterion. Several primitives for this criterion are defined as the cardinalities of: **pos(H)** the positive instances of a hypothesis H, **neg(H)** the negative instances of H, **pred(H)** the unknown, i.e. neither provable true nor provable false instances of the conclusion which will be predicted by H, **total(H)** the total instances of H, **unc(H)** the instances of the conclusion which are uncovered by H, and **concl(H)** all instances of the conclusion. The acceptance criterion is a logical expression of conjunctions and disjunctions of arithmetical comparisons (i.e. $=, <, \leq, \geq, >$) involving arithmetical expressions (i.e. $+, -, *, /$) built from numbers and the above primitives, for example:

$$\text{pos(H)} > 4 \ \& \ \text{neg(H)} < 1 \ \& \ \text{unc(H)} < (0.9 * \text{total(H)}).$$

For all specializations of a hypothesis H the numbers of positive, negative, and predicted instances are smaller than those numbers for the hypothesis H itself, whereas the number of uncovered instances grows. The number of instances of the conclusion does not change while specializing H. Using these relations among the primitives, a pruning criterion is derived from the acceptance criterion. It only prunes hypotheses which cannot be accepted.

Further effectiveness of the algorithm RDT comes from the use of a many sorted logic and a topology of predicates (see [7]).

3.4.2 The learning task and its results

Here, the goal of learning is to gain a characterization of the estimators concerning the categorized data characteristics. From now on, we only consider those data characteristics that do not vary across the different strata. The remaining nine data characteristics are classified in a total number of 22 categories. The goal predicates to learn about are the four classified (assessed) recommendations and additionally the unassessed recommendation. We carried out further learning steps with some different predicates, e.g. those for the suitabilities, see [10]. To define the hypothesis space for learning, RDT needs suitable metapredicates. We used the following ones:

```
MP1(S,P1,R): S(Est) & P1(Sit) --> R(Sit,Est).
MP2(S,P1,P2,R): S(Est) & P1(Sit) & P2(Sit) --> R(Sit,Est).
MP3(S,P1,P2,P3,R): S(Est) & P1(Sit) & P2(Sit) & P3(Sit) --> R(Sit,Est).
MP4(S,P1,P2,P3,P4,R): S(Est) & P1(Sit) & P2(Sit) & P3(Sit) & P4(Sit)
--> R(Sit,Est).
```

While learning, `S(Est)` is instantiated with predicates that determine the variable `Est` in the conclusion, e.g. `mantel_haenszel(Est)`. Using the first metapredicate we search for data characteristics where a single property is sufficient to recommend an estimator. In the next steps we search for combinations of two, three and four characteristics.

In the presented knowledge base, we do not want to infer new facts from the learned rules. Hence, we consider two acceptance criteria. The first one does not allow predicted facts at all, where the second one allows a maximum number of 0.1*total for the predicted facts. The main results of the learning step are summarized in Table 2.

3.4.3 Selecting a rule set

The goal now is to select a rule set from the learned rules, which is then integrated into the KBS. As criterium for this selection, we mainly consider the redundancy in the rule set. Frequently, different rules cover the same data constellations. The following example illustrates this:

> We consider two rules:
> 1. `mantel_haenszel(mh) & oddsratio=1(S) & large_cases(S)`
> `& large_strata(S)` → `medium_recommendation(S,mh).`
> 2. `mantel_haenszel(mh) & oddsratio=1(S) & large_cases(S)`
> `& balanced_gini_ratio_cases(S)` → `medium_recommendation(S,mh).`
>
> These two rules only differ from each other in the last premise. Both rules cover the first six parameter constellations of the simulation study. The reason for this is that there is a relation between the data characteristics `small_strata` and `balanced_gini_ratio_cases` in the simulation study at issue. This relation is fixed in the simulation design: in situations with few strata, the number of cases is always uniformly distributed over the strata.

Because of relations like this, redundant rules have been learned. Without going into detail we would like to mention that our choice of only one rule out of the set of redundant rules is based on certain data characteristics (see [10]).

Goal predicates	**MP**	**AC**	**Results / Remarks**
all	MP1	all	no learned rules
`very_good_recommendation`	MP2	pos>0.9*total pos=total	no learned rules
`good_recommendation`	MP2	pos=total	one learned rule time for learning: 3343 seconds rule for J_{MH}^{II}
`medium_recommendation`	MP2	pos>0.9*total pos=total	no learned rule
`bad_recommendation`	MP2	pos>0.9*total pos=total	no learned rule
`recommendation`	MP2	pos=total	eight learned rules time for learning: 7137 seconds rules for J_{MH}^{II}
`recommendation`	MP2	pos>0.9*total	10 rules time for learning: 8452 seconds rules for J_{MH}^{II}
`very_good_recommendation`	MP3	pos>0.9*total pos=total	no learned rules
`good_recommendation`	MP3	pos=total	nine learned rules time for learning: 30044 seconds rules for J_{MH}^{II}
`good_recommendation`	MP3	pos>0.9*total	18 learned rules time for learning: 31223 seconds rules for J_{MH}^{II}
`medium_recommendation`	MP3	pos=total	24 learned rules time for learning: 32678 seconds rules for JK_{MH}^{II}
`bad_recommendation`	MP3	pos=total	one learned rule time for learning: 29452 seconds rule for $\hat{\psi}_{MH}$
`very_good_recommendation`	MP4	pos=total	no learned rules
`good_recommendation`	MP4	pos=total	c. 80 rules rules for JK_{MH}^{II} and JK_{W}^{I}
`medium_recommendation`	MP4	pos=total	c. 100 rules learning process failed[a]
`bad_recommendation`	MP4	pos=total	12 rules rules for $\hat{\psi}_{MH}$

[a]The learning process was stopped after one week.

Table 2: Results obtained from the process of learning

For the second metapredicate the following five rules are selected:

```
   unbalanced_ratio_CoCa(S) & large_strata(S)
→ good_recommendation(S,jk)

   small_oddsratio(S) & unbalanced_ratio_CoCa(S)
→ recommendation(S,jk)

   small_prob(S) & small_strata(S)
→ recommendation(S,jk)

   gini_ratio_CoCa_balanced(S) & oddsratio_exactly_one(S)
→ recommendation(S,jk)

   gini_prob_unbalanced & small_oddsratio(S)
→ recommendation(S,jk)
```

We selected six rules for the third metapredicate:

```
   oddsratio_exactly_one(S) & large_cases(S) & small_strata
→ moderate_recommendation(S,jk)

   small_oddsratio(S) & large_cases(S) & small_strata
→ moderate_recommendation(S,jk)

   moderate_strata(S) & oddsratio_exactly_one(S) & large_cases(S)
→ good_recommendation(S,jk)

   large_strata(S) & oddsratio_exactly_one(S) & large_cases(S)
→ good_recommendation(S,jk)

   small_strata(S) & oddsratio_exactly_one(S) & very_large_cases(S)
→ good_recommendation(S,jk)

   gini_cases_medium_balanced(S) & large_difference_prob(S) & large_oddsratio(S)
→ bad_recommendation(S,mh)
```

Finally, there are seven rules which lead to a recommendation of an estimator based on a combination of four data characteristics.

```
   large_strata(S) & small_oddsratio(S) & large_cases(S)
& small_prob(S) → good_recommendation(S,jk)

   moderate_strata(S) & small_oddsratio(S) & small_prob(S)
& large_cases(S) → moderate_recommendation(S,jk)

   moderate_strata(S) & moderate_oddsratio(S) & small_prob(S)
& large_cases(S) → moderate_recommendation(S,jk)
```

```
   moderate_strata(S) & small_oddsratio(S) & small_prob(S)
& very_large_cases(S) → good_recommendation(S,jk)

   large_strata(S) & large_oddsratio(S) & large_cases(S)
& centered_prob(S) → bad_recommendation(S,mh)

   large_strata(S) & small_oddsratio(S) & large_cases(S)
large_difference_prob(S) → good_recommendation(S,w_jk)

   large_strata(S) & large_oddsratio(S) & large_cases(S)
small_prob(S) → recommendation(S,jk_ii)
```

Using the traces for the learning process for metapredicate MP4 and with the help of the experts, we proposed some possible rules, entered them into the system MOBAL and investigated how many new facts were predicated by this rule. Thus, the following rules were discovered and accepted:

```
   small_strata(s) & large_oddsratio(S) & very_large_cases(S)
& centered_prob(S) & small_difference_prob(S) → good_recommendation(S,jk)

   large_strata(S) & large_oddsratio(s) & very_large_cases(S)
& centered_prob(S) & small_difference_prob(S) → moderate_recommendation(S,jk)

   small_strata(s) & large_oddsratio(S) & large_cases(S)
& unbalanced_ratio_CoCa(S) & large_difference_prob(S)
→ bad_recommendation(S,w_jk)

   moderate_strata(s) & centered_prob(S) & small_difference_prob(S)
& large_oddsratio(S) & large_cases(S) → bad_recommendation(S,jk_ii)

   small_strata(S) & large_oddsratio(S) & large_cases(S)
medium_balanced_ratio_CoCa(S) & unknown(small_prob(S))
→ bad_recommendation(S,woolf)
```

The four rules listed above do not predict any new facts in the represented knowledge base. Additionally, we discovered a rule with six premises. This rule also does not predict a new fact:

```
   small_strata(S) & large_oddsratio(S) & large_cases(S)
& unbalanced_ratio_CoCa(S) & small_difference_prob(S) & centered_prob(S)
→ bad_recommendation(S,bl)
```

3.4.4 Evaluating the rule set

With the help of MOBALs Rule Restructuring Tool (RRT) (see [14]) we have evaluated the selected rule set. In Table 3, the results of the evaluation of the selected rule set

Completeness	40%
Correctness	93%
Redundancy	no
Number of premises	3.6
Number of variables	1
Number of constants	1
Covered instances	4

Table 3: Results of the evaluation

according to the criteria completeness, correctness, redundancy, the rule lenght, and the number of covered instances are depicted.

3.4.5 Integration of the selected rules into the KBS

To develop the advice tool, we integrate the selected rules into the KBS. The advice tool then analyses the actual case–control data with respect to their characteristics and selects a rule whose premises are covered by these ascertained characteristics. This rule recommends an appropriate estimator for the common odds ratio. For efficiently encoding the rules, we have to determine an order for querying the data characteristics. Additionally, we have to fix an order for the rules, because there are data constellations, where more than one rule could fire. If there are many of such data situations, it would be recommendable to revise the rule set. The reason why those rules have been learned is that the simulation study does not cover all combinations of characteristics that will occur in practice. The former order is achieved by considering the frequencies of the data characteristics. The most frequent characteristic, that means the value of the odds ratio, is questioned first. For every category of this data characteristic the frequencies of the remaining properties are determined and so on. The latter order (of the rules) is determined by considering the number of positive examples for the rules. Figure 4 depicts a cutout of the resulting decision tree(s). We search for an applicable rule (a recommendation) top down in the left tree. If we cannot find any rule in this tree, we start searching in the tree on the right hand side.

4 Design

CORA is a Windows application. The user interface is composed of forms that are created with the tool Delphi (see [15]). Delphi simplified the implementation of the graphical interface and additionally allowed to encode the expertise and all statistical procedures.

In the following sections we give a short description of the design of the advice and the analysis tool as well as of the help system.

4.1 Design of the advice tool

The advice tool gives a recommendation of an estimator for the common odds ratio. This recommendation is based on two data sets, the data to be analysed and the data of the

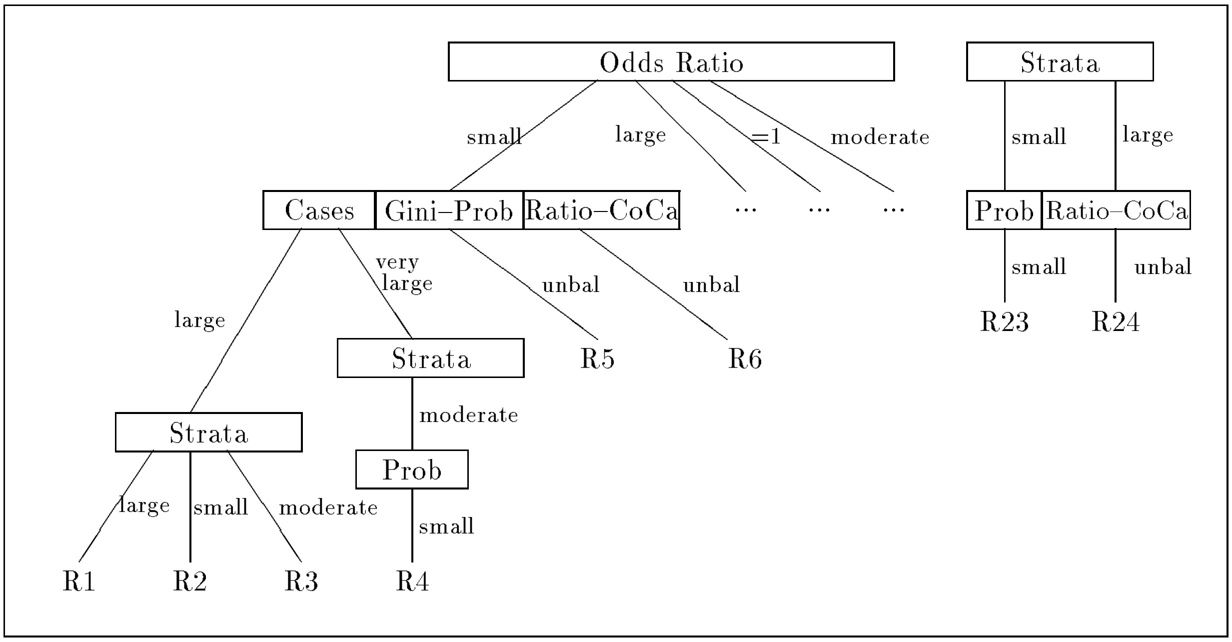

Table 4: Cutout of the resulting decision tree(s)

pilot study, where the pilot study could be an external study being carried out in advance or a random sample drawn from the original data set of size 10 %, for instance. The data of the pilot study is only used for deriving the recommendation and should not be used for further analyses. To give this recommendation, three steps are necessary:

- the investigation of the two data sets,
- the classification of the investigated characteristics of the data and
- the choice of an appropriate rule out of the set of rules integrated in CORA.

All these steps are within the scope of the system, that means, there are procedures to calculate the values of the data characteristics, procedures for their classification, and a procedure that implements the decision tree in form of production rules by using nested if–then–statements. Every if–statement corresponds to a premise of a rule that examines a classified data characteristics. The corresponding then statement represents the conclusion of the production rules.

While implementing this decision tree the order for the rules mentioned in Section 3.4.5 has to be taken into account. Then, the search for an appropriate rule stops as soon as the most preferable one has fired.

The interface for the advice tool consists of a three–sided form. On the first page (see Figure 2) the proposed point estimator and a possible bias correction as well as information about the degree of suitability of the estimator and the number of the rule by which this recommendation is derived are presented.

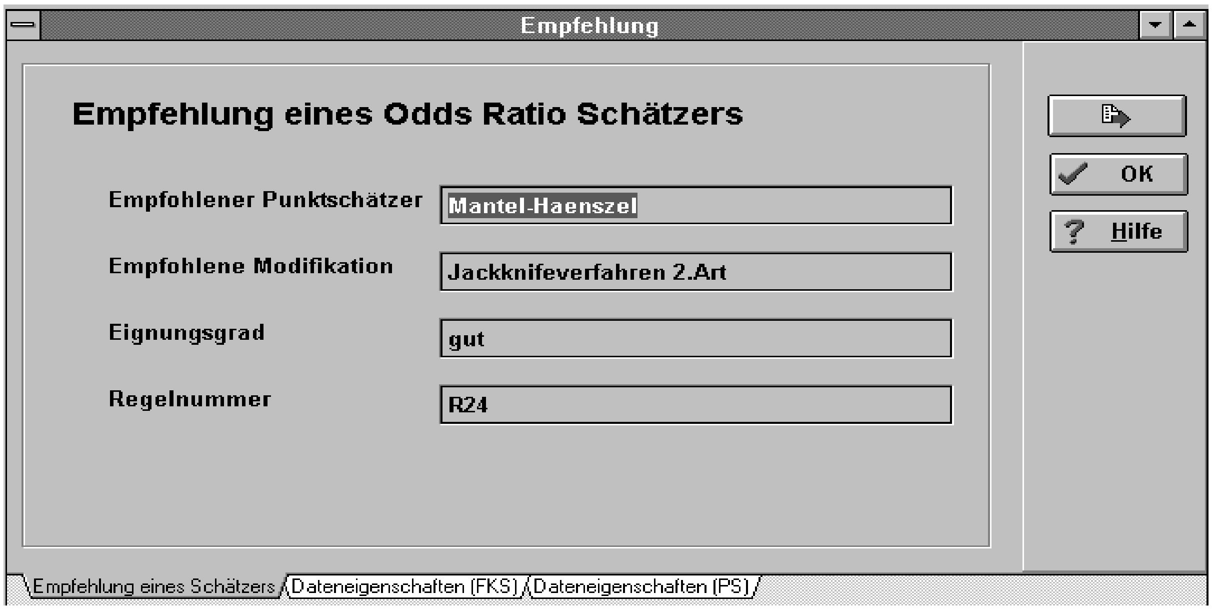

Figure 2: First page of the form for the recommendation

The second page (see Figure 3) shows the characteristics of the data that the user wants to analyse, namely the number of tables, the mean number of cases, the balance of the number of cases, the mean ratio of cases and controls, and the balance of this ratio. Both, the exact values and the classified characteristics are shown.

The characteristics of the pilot data are finally listed on the third page: the estimated common odds ratio, the mean probability for a control being exposed and its balance as well as the differences between the extreme probabilities.

For all these characteristics the exact values and their categorized versions are presented. The characteristics occurring as premises in the applied rule are checked. Thus, all information from the production rules is transparent for the user. Further information, e.g. the coverage of the applied rule, is so far not represented on the form.

4.2 Design of the analysis tool

As mentioned before, CORA is not only an intelligent interface, but includes also the used statistical procedures: there are two methods to stratify the data, five procedures for analysing homogeneity and independence of risk factor and disease as well as the above mentioned methods to estimate the common odds ratio. Additionally, the user has the possibility to get a general view of the data. The form that is depicted in Figure 4 shows the data as stratified 2×2 contingency tables. Repeated calls of this form allow to view several tables simultaneously. The second page of this form shows the corresponding estimates for the individual odds ratios, their estimated variances, and the confidence intervals. Two other forms show listings of all zero cells that are in the set of data and all individual odds ratios. There are two additional forms where defaults regarding the statistical methods can be set.

Figure 3: Second page of the form for the recommendation

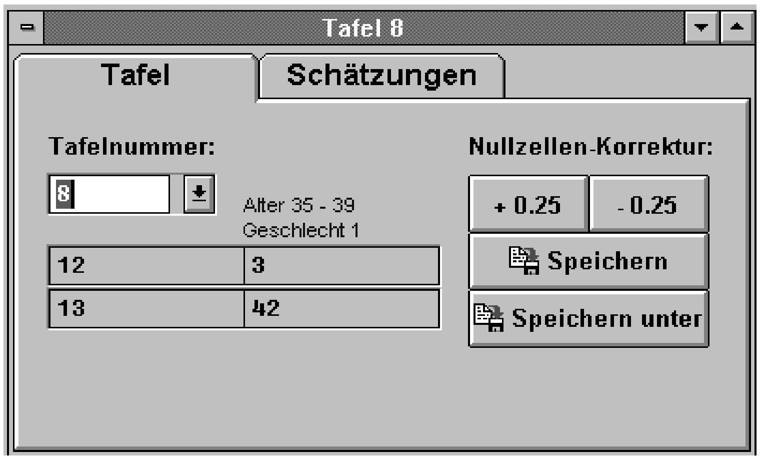

Figure 4: Form for the contingency tables

Figure 5: Form for the estimation of the common odds ratio

4.3 Design of the help system

The help system was created using the Windows help compiler. There are two different kinds of topics: on the one hand there are the statistical and epidemiological topics and on the other hand the user can get information about CORA and instructions how to use it. Especially, the use of the advice tool and its foundations are explained in this help system.

Every form of CORA contains a help button that allows to jump to a help topic that describes all its components. From there, it is possible to reach the relevant statistical topics.

The system also addresses users with limited statistical experience. Hence, the steps and basic concepts concerning a stratified contingency table analysis are explained. The statistical procedures are described and their finite and asymptotic properties are stated. Thus, these help topics complete the recommendations given by the advice tool.

5 Discussion

Two different aspects were important when developing CORA. On the one hand we analysed Monte–Carlo studies by using AI techniques for obtaining rules to be implemented in CORA. And on the other hand these rules for recommending a certain estimator in a given data situation and the complete system of support formed the essential part of our statistical analysis system.

In spite of efficiency problems due to the MOBAL system, past experience has shown that with the use of AI techniques we are able to improve the evaluation of Monte–Carlo studies by more specific results. This improvement is not only achieved by machine leraning but also by modelling the knowledge and by examining existing rules with the help of MOBAL.

While working with MOBAL, the domain expert is able to recognize his/her own way of evaluating Monte–Carlo studies and thus can follow the steps of modelling the expertise.

An important benefit from using machine learning is that also more extensive studies can be analysed without much additional effort. For an evaluation with conventional techniques the present Monte–Carlo study with its 240 investigated parameter constellations is already very complex. Thus more extensive studies cannot be carefully analysed with reasonable effort. But especially enlarged studies would be important to improve the quality of the rule set and hence to lead to rules that cover a wider range of potential data situations. Moreover, the modelling of these Monte–Carlo studies has revealed the importance to further examine especially situations with a poor behaviour of the estimators. This would enable the system to give "negative recommendations" that means warnings which estimator the user should avoid analysing the given data.

In addition, the expertise should be completed by rules based on asymptotical properties of the estimators and on characteristics resulting from the calculation of the estimators and thus being independent of the data structure. Within the scope of this approach it is quite simple to enlarge the rule set according to these aspects.

Other possible expansions of the underlying statistical expertise concern the fact that CORA only supports the decision process of the user with respect to the choice of an appropriate point estimator of the common odds ratio. The presented approach can, however, also be applied to support other choices of statistical methods that have to be made by the user during the analysis, e.g. concerning an appropriate test of homogeneity of the individual odds ratios or a test of independence of risk factor and disease.

In addition to these points, future work should provide such systems with better methods for graphical representation of data, which would increase the insight into the data.

Of course, the system could further be extended with respect to a support of the user not only when analysing the data, but also when planning the study and collecting the data.

Acknowledgements The author wishes to thank Prof. Dr. K. Morik for her support concerning the Machine Learning and Modeling aspects and Prof. Dr. I. Pigeot, the head of the CORA–project, for giving the impulse to develop CORA and for providing the statistical background.

References

[1] J.B.S. Haldane, The estimation and significance of the logarithm of a ratio of frequencies, Annals of Human Genetics 20 (1955) 309–311.

[2] B. Woolf, On estimating the relation between blood group and disease, Annals of Human Genetics 19 (1955) 251–253.

[3] N. Mantel and W. Haenszel, Statistical aspects of the analysis of data from retrospective studies of disease, Journal of the National Cancer Institute 22 (1959) 719–748.

[4] N.E. Breslow and K.Y. Liang, The variance of the Mantel–Haenszel estimator, Biometrics 38 (1982) 943–952.

[5] I. Pigeot, A jackknife estimator of the combined odds ratio, Biometrics 47 (1991) 373–381.

[6] K. Morik, Sloppy modeling, in: Knowledge representation and organization in machine learning, ed. K. Morik, pp. 107–134 (Springer–Verlag, Berlin, New York, 1989).

[7] K. Morik, S. Wrobel, J.U. Kietz, and W. Emde, Knowledge acquisition and machine learning: Theory, methods and applications, in: Knowledge–based systems (Academic Press, London, 1993).

[8] I. Pigeot, A simulation study of estimators of a common odds ratio in several 2×2 tables, Journal of Statistical Computation and Simulation 38 (1991) 65–82.

[9] I. Pigeot, Jackknifing estimators of a common odds ratio from several 2×2 tables, in: Bootstrapping and related techniques, eds. K.H. Jöckel, G. Rothe and W. Sendler, pp. 203–212 (Springer–Verlag, Berlin, Heidelberg, 1992).

[10] U. Robers, Entwicklung eines wissensbasierten Assistentensystems zur Analyse von Fall–Kontroll–Studien, Masters thesis, University of Dortmund, Dept. of Computer Science VIII, 1995.

[11] J.U. Kietz, S. Wrobel, Controlling the complexity of learning in logic through syntactic and task–oriented models, in: Inductive logic programming, ed. S. Muggleton, A.P.I.C. Series 18, pp. 335–360 (Academic Press, London, 1992).

[12] G. D. Plotkin, A note on inductive generalization, in: Machine Intelligence, eds. B. Meltzer and D. Michie, pp. 153–163 (American Elsevier, 1970).

[13] W. Buntine, Generalized subsumption and its applications to induction and redundancy, Artificial Intelligence 36 (1988) 149–176.

[14] E. Sommer, MOBALs theory restructuring tool RRT, ESPRIT Project ILP (6020), ILP Deliverable GMD 2.2, 1995.

[15] Borland GmbH (Ed.), Delphi for Windows 1.0, User Guide, 1995.