# Bounded Variation in Multi-Stage Optimization

Dissertation
zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

Der Fakultät für Mathematik der
Technischen Universität Dortmund
vorgelegt von

Maja Valentina Hügging

im August 2025

**Dissertation**

# Bounded Variation in Multi-Stage Optimization

Fakultät für Mathematik
Technische Universität Dortmund

Erstgutachter: Prof. Dr. Christoph Buchheim

Zweitgutachter: Prof. Dr. Marc Pfetsch

Tag der mündlichen Prüfung: 31.10.2025

## Abstract

This thesis studies the computational complexity of a multi-stage optimization problem with either penalized or bounded variation. The input of the problem is a sequence of instances (one for each time stage), and the task is to find a sequence of solutions (one for each time stage) that achieves a tradeoff between the quality of the solutions in each time stage and their similarity. The amount of change in the sequence of solutions is quantified by the so-called *variation* which will be measured time-wise, component-wise, or in total. The variation of the solution sequence is either incorporated into the model by a penalty term or by imposing a hard upper bound in the set of constraints. This gives rise to two related but not equivalent multi-stage problems.

When only one binary decision can be made in each stage, the tractability of the problems with either of the three types of variation will be settled by presenting respective compact extended LP-reformulations. A thorough polyhedral study of the problem versions will yield complete, and usually exponentially large, descriptions of the convex hull of feasible solutions in the original variable space and also corresponding efficient separation algorithms.

For a higher-dimensional underlying combinatorial problem, the complexity of the problem versions will turn out to depend on the type of variation and also on whether this is penalized or bounded. Indeed, even for an underlying problem as trivial as a SELECTION PROBLEM, one version will be strongly $\mathcal{NP}$-hard, while others will be tractable.

An oracle-based approach will shift the perspective from a complexity-theoretical one to a rather information-related one in order to investigate the relative complexity of the bounded and penalized problem version with respect to the underlying combinatorial problem.

## Zusammenfassung

Diese Arbeit befasst sich mit der Komplexität eines multi-stage Optimierungsproblems unter entweder penalisierter oder beschränkter Variation. Das hier betrachtete multi-stage Problem besteht aus einer Menge von Instanzen (jeweils eine pro Zeitschritt), für die es eine Folge von Lösungen (ebenfalls eine pro Zeitschritt) zu finden gilt. Dabei muss ein Kompromiss zwischen der Qualität der Lösungen in den jeweiligen Zeitschritten und ihrer Ähnlichkeit erreicht werden. Die Variation der Lösungsfolge bezeichnet hier ein Maß für die zwischen den Lösungen auftretende Änderung. Sie wird in dieser Arbeit zeitlich, komponentenweise oder insgesamt gemessen und entweder gewichtet mit einem Strafterm in die Zielfunktion des Problems integriert oder durch eine feste obere Schranke in den Nebenbedingungen begrenzt. Dies führt zu zwei verwandten, aber nicht äquivalenten Modellen, wobei der Fokus meist auf letzterer Variante liegt.

Darf in jedem Zeitschritt nur eine einzige binäre Entscheidung getroffen werden, stellen sich alle Problemvarianten als effizient lösbar heraus, etwa durch eine kompakte erweiterte LP-Formulierung. Es erfolgt eine umfassende polyedrische Untersuchung jeder der Problemvarianten. Diese liefert jeweils für jede der Varianten eine vollständige Beschreibung der konvexen Hülle im ursprünglichen Variablenraum mittels linearer Ungleichungen. Da diese Beschreibungen hier meist exponentiell groß sind, werden ebenfalls entsprechende effiziente Separationsalgorithmen vorgestellt.

Bei einem höherdimensionalen zugrundeliegenden kombinatorischen Optimierungsproblem hängt die Komplexität der Problemvarianten sowohl von der Art der Variation ab, als auch davon, ob diese penalisiert oder beschränkt ist. Es stellt sich heraus, dass selbst bei einer trivialen kombinatorischen Struktur, wie sie etwa durch eine Selection-Bedinung gegeben ist, eine der Varianten stark $\mathcal{NP}$-schwer ist, wohingegen andere Varianten polynomiell lösbar sind.

Durch einen orakelbasierten Ansatz wandelt sich die komplexitätstheoretische Sichtweise zu einer eher informationsbezogenen, die darauf abzielt, die Komplexität der beschränkten und penalisierten Problemvarianten relativ zum zugrundeliegenden kombinatorischen Problem zu untersuchen.

## Acknowledgement

There are numerous people to whom I am sincerely grateful for their support throughout my PhD. First and foremost, I would like to thank my supervisor, Christoph Buchheim, for his advice and guidance. He was always approachable and supportive when I faced problems or had questions. The joint work on the results presented in this thesis was truly a pleasure and is very much appreciated.

I would also like to thank Prof. Dr. Marc Pfetsch for kindly taking on the role of reviewer for this thesis.

My time at TU Dortmund was a wonderful experience, and I am thankful to all my colleagues, whom I will dearly miss, for creating such a supportive and enjoyable working atmosphere.

Finally, I am deeply grateful to my friends, my boyfriend, and my family – especially my sisters, Tessa and Nora – for their unwavering support and belief in me.

# Contents

# Chapter 1

# Introduction

Adapt or pay the price for your negligence!

The world is ever-changing, therefore an optimization problem that models a real life problem should take into account the dynamic behavior of the environment it tries to capture. In classical combinatorial optimization, given an instance of the problem, the task is to determine a feasible solution optimizing the objective function. However, as the instance may change over time, a solution that is optimal today is unlikely to still be optimal tomorrow. Consequently, the solution has to be adapted as the input changes. The assumption that one may freely adapt the solution without any limitation is unrealistic, since the blunt approach to simply find a new optimum over and over again does not take into account the cost of changing the solution. This leads to the field of multi-stage optimization problems, where the input is a sequence of instances (one for each time stage), and the task is to find a sequence of solutions (one for each time stage) that achieves a tradeoff between the quality of the solutions in each time stage and their similarity. The model quantifies the amount of change in the sequence of solutions, called the *variation*, and assumes that this incurs some cost.

The *cost* of adapting a solution may be represented as penalty fee which is integrated into the objective function. For example, consider an instance of an assignment problem where the task is to compute a best assignment between actors and roles in a theater play. It is assumed that the value of assigning a role to an actor is known and depends on the actor's experience, acting ability, or salary. In the classical setting, this problem can be solved efficiently. Now, however, there is the additional requirement that across all performances of the play, the actor-role assignments should remain largely consistent, since changes in the assignments incur a penalty fee that compensates the actors for the extra work required to learn new lines. Furthermore, the value of actor-role assignments will change over time due to an actor getting a pay raise or gaining more experience. In the corresponding multi-stage model, the variation of all determined assignments is then quantified and

incorporated into the model as a penalty term which worsens the objective value of the determined schedule.

The latter multi-stage model is well suited only for settings where it is possible to compensate the variation in the solution by paying a penalty. However, there are also applications in which the variation has a hard upper bound that may not be exceeded under any circumstances. Here, the *cost* of the variation is instead a reduction of the capacity for any further changes. For example, consider a set of valves in a gas network where the task for each time stage is to determine the optimal set of valves which are open, while the others remain closed. Now, due to legal regulations, safety reasons, or just to avoid the wear and tear of the involved machinery, it is prohibited to have too many changes in the selection of the open and closed valves over time. The corresponding multi-stage model differs from the previous penalty model in the sense that the variation is now limited by a hard upper bound in the set of constraints, but it does not affect the objective value directly. In this second multi-stage model with bounded variation, it is particularly important how the variation is quantified. There exist several possibilities for the measurement of the variation, for example: *time-wise variation*, where the amount of change is measured separately between each pair of consecutive time stages, *switch-wise variation*, where the amount of change in each component of a solution is measured individually over all time stages, and, finally, *total variation*, which measures the total change over all components and all time stages.

This thesis is set out to study the computational complexity of the penalty multi-stage problem and the multi-stage problem with bounded variation for the latter three types of variation. To that end, it will begin with the study of the special case where only one binary decision is taken at each stage. In the context of the previous example, this will correspond to the setting in which there is only one valve, or *switch*, that needs to be controlled. This special case will be studied in detail, even beyond the question of complexity, by focusing on the investigation of the problem's polyhedral structure. Subsequently, this thesis continues to study a particular underlying problem with higher dimension, more precisely, a Selection Problem. In the valve-example from before, this corresponds to a constraint that restricts number of open valves at each stage. Here, the focus will lie on determining the computational complexity of the respective problem versions. As it will turn out, the complexity will strongly depend on the type of variation and also whether this is penalized or bounded. Finally, the multi-stage problem with bounded variation will be generalized one step further, in fact, the perspective will shift from a complexity-theoretical one rather an information-oriented one, which pursues an oracle-based approach.

A formal introduction will be given in the following chapter. It will also provide an overview of the state of the art, the contribution of this thesis, as well as a summary of the fundamental concepts needed for its understanding.

# Chapter 2

# Preliminaries and Related Work

This chapter is dedicated to the presentation of the problems which will be studied in the course of this thesis. It will provide an overview of the structure of this thesis, summarize its most important findings, and present the concepts needed in the following chapters.

Given a classical linear combinatorial optimization problem

$$\text{(LO)} \qquad \begin{cases} \max & c^\top x \\ \text{s.t.} & x \in X \subseteq \{0,1\}^n \,, \end{cases}$$

the task is to determine a feasible solution which maximizes the objective value. The problem parameters, here the objective $c \in \mathbb{R}^n$ and the feasible set $X$, can be considered a snapshot of the environment made at some (discrete) time point. Multiple such snapshots over a discrete time horizon $\{0, 1, ..., T\}$ produce $T + 1$ *underlying problems* (LO) with the corresponding data $X_t \subseteq \{0,1\}^n$ and $c_t \in \mathbb{R}^n$ for $t = 0, ..., T$. Clearly, all of these problems can be solved independently of one another. The disadvantage of this setting, however, is that a sequence of feasible solutions $x_t \in X_t$, $t = 0, ..., T$, may fluctuate significantly. A high fluctuation in the sequence may be especially undesirable as it can be very expensive (e.g., incurring personnel cost) or even forbidden (e.g., wear-down of material, safety reasons, government regulations).

Thus, this thesis introduces an additional stability requirement for the solutions by taking into account their *variation*, given by a function $\text{Var} : \times_{t=0}^{T} X_t \to \mathbb{R}^r_{\geq 0}$. Imagine the discrete time horizon $\{0, ..., T\}$ to be arranged on a horizontal line, and the vectors $x_t \in X_t$ as $n$-dimensional column vectors. Then, the structure of the problem can be sketched as depicted in Figure 2.1 below.

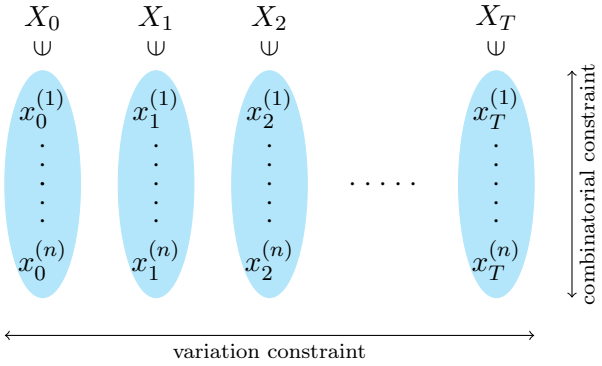



Figure 2.1: Sketch of the structure of Problem (BND)

As the variation establishes a *horizontal* relationship among the solutions, it is no longer possible to consider the underlying problem separately for each stage. The stage-wise feasibility with respect to the combinatorial structure is considered a *vertical* constraint. The aim of this thesis is to study this dynamic problem regarding its complexity, more precisely, regarding the question whether the consideration of the variation causes an increase in complexity.

Of course, this depends on the variation function itself, but also on how this is incorporated into the temporal setting. This thesis will study two possibilities in which the variation is taken into account. In the first variant, a positive variation incurs a *penalty* $\lambda \in \mathbb{R}^r_{>0}$ in the objective function.

$$\text{(P)} \qquad \begin{cases} \max & \displaystyle\sum_{i=1}^{n}\sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} - \lambda^\top \mathrm{Var}(x) \\ \text{s.t.} & x_t \in X_t \subseteq \{0,1\}^n \qquad t = 0, ..., T\,. \end{cases}$$

In the second variant, the variation may not exceed some given bound $S \in \mathbb{N}^r_{\geq 0}$. Thus, the variation is located in the set of constraints:

$$\text{(BND)} \qquad \begin{cases} \max & \displaystyle\sum_{i=1}^{n}\sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} \\ \text{s.t.} & \mathrm{Var}\,(x) \leq S \\ & x_t \in X_t \subseteq \{0,1\}^n \qquad t = 0, ..., T\,. \end{cases}$$

**Notation 2.0.1.** A single variable $x_t^{(i)}$ in either of the above models refers to the respective stage via the subscript $t \in \{0, ..., T\}$. The superscript $(i)$ refers to the $i$-th component of $x_t$. Just as $x_t$ represents the vector $(x_t^{(1)}, ..., x_t^{(n)})^\top \in \mathbb{R}^n$, the vector $x^{(i)}$ without subscript refers to $(x_0^{(i)}, ..., x_T^{(i)}) \in \mathbb{R}^{T+1}$. The same notation is

used for the objective coefficients $c$ and for future variables in the space $\mathbb{R}^{n\times(T+1)}$. Throughout this thesis, the $n$-dimensional vector consisting of ones is denoted by $\mathbb{1}_n$. The $n$-dimensional zero vector is denoted by $\mathbb{0}_n$.

For each component $i = 1, ..., n$, the (horizontal) vector $x^{(i)} = (x_0^{(i)}, ..., x_T^{(i)})$ can be considered a binary incidence vector of a *control switch* that is either *active* ($x_t^{(i)} = 1$) or *incative* ($x_t^{(i)} = 0$) at a stage $t$. Then, the dynamic problem with variation consists of $n$ parallel, horizontal switches that are connected vertically via the membership in $X_t$ for each stage $t$. The set $X_t$ can, for example, ensure that certain subsets of switches are not active at the same time, or restrict the total number of switches which are active at each stage. For a better intuition, the term *switch* will often be used interchangeably with the term *component* in the course of this thesis. In particular, this allows for an intuitive definition for an *activation* and *deactivation* of a switch:

**Notation 2.0.2.** If $x_{t-1}^{(i)} = 0$ and $x_t^{(i)} = 1$, then the switch $i \in \{1, ..., n\}$ is *activated* at stage $t \in \{1, ..., T\}$. Analogously, if $x_{t-1}^{(i)} = 1$ and $x_t^{(i)} = 0$, the switch $i$ is *deactivated* at stage $t$.

It remains to present the three types of variation studied in this thesis. The term *time-wise variation* is associated with the following variation function:

$$\mathrm{Var}_{tw} : \bigtimes_{t=0}^{T} X_t \longrightarrow \mathbb{R}^T_{\geq 0}, \qquad \mathrm{Var}_{tw}(x)_t := \sum_{i=1}^{n} |x_{t-1}^{(i)} - x_t^{(i)}| \quad t = 1, ..., T\ .$$

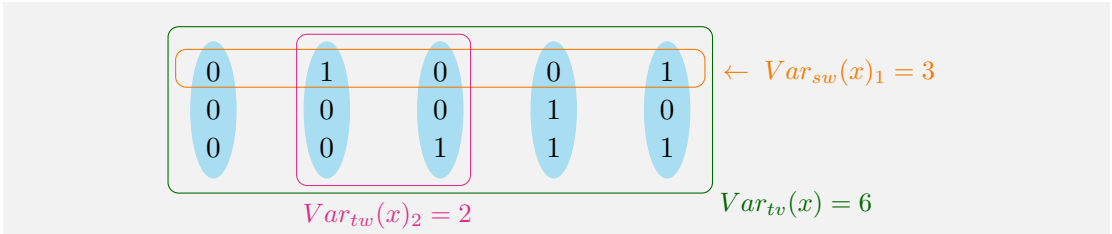
The term *switch-wise variation* is associated with the variation function

$$\mathrm{Var}_{sw} : \bigtimes_{t=0}^{T} X_t \longrightarrow \mathbb{R}^n_{\geq 0}, \qquad \mathrm{Var}_{sw}(x)_i := \sum_{t=1}^{T} |x_{t-1}^{(i)} - x_t^{(i)}| \quad i = 1, ..., n\ ,$$

and, finally, the term *total variation* is associated with function

$$\mathrm{Var}_{tv} : \bigtimes_{t=0}^{T} X_t \longrightarrow \mathbb{R}_{\geq 0}, \qquad \mathrm{Var}_{tv}(x) := \sum_{i=1}^{n}\sum_{t=1}^{T} |x_{t-1}^{(i)} - x_t^{(i)}|\ .$$

**Example 2.0.1.** For the binary elements depicted below, some exemplary values of the three different variation types are given. Each variation type is indicated by an individual color.




This thesis is set out to settle the *computational (time-)complexity* of the two problems (BND) and (P) considering the three presented types of variation.

**The theory of computational time-complexity** can be considered the foundation of discrete and, in particular, combinatorial optimization. Thus, the reader is assumed to be familiar with the overall concept, and the following surface-level description of the theory is restricted to the most important ideas with the aim to strike a compromise between formality and intuition. For an extensive and detailed introduction into the field, the reader is referred to [GJ90] or [AH74].

It is quite obvious that a combinatorial optimization problem can be solved in finite time by enumeration of all feasible solutions. It is just as obvious that this is generally not a good approach as the number of feasible solutions may be very large. This is what gave rise to the field of computational time-complexity which can be described as the study of problems regarding their *difficulty*, i.e., the time (and space) that is required to solve them. Since the work of Cook and Karp in the 1970s, the *efficiently solvable problems* are distinguished from the so-called $\mathcal{NP}$*-complete* (or $\mathcal{NP}$*-hard*) problems. This distinction is based on the time that is required to solve a given problem regarding its size. The latter depends on the encoding of the problem which is realized by a *languange*: Let $\Sigma$ be a finite set (e.g., $\Sigma = \{0, 1\}$), then $\Sigma$ is called an *alphabet* and its elements are called *letters*. An ordered finite sequence of letters from $\Sigma$ is called a *word* and $\Sigma^*$ is defined as the collection of all words from the alphabet $\Sigma$. A subset of $\Sigma^*$ is called *language* and the *size* or *encoding length* of a word $x$ is the number of its letters, denoted by $size(x)$.

A *decision problem* $P = (X, Y)$ consists of a language $X$ containing the encoded *instances* (or *input*) of the problem, and a subset $Y \subseteq X$ which contains the so-called *yes-instances*. The set $X \setminus Y$ is called the set of *no-instances*. For a given instance $x \in X$, the task of the decision problem $P = (X, Y)$ is to determine whether this is a yes-instance, i.e., $x \in Y$.

A decision or (discrete) optimization problem can be decided/solved by handing it to an *algorithm*. The concept of an algorithm as a list of instructions is formalized by Turing-machines. For a detailed description of the latter, the reader is referred to [GJ90]. For this thesis, it suffices to think of a Turing-machine as a tool which, given a word (over some alphabet), performs a finite number of certain steps and

finally stops with some output word (over the same alphabet). An algorithm is said to decide a decision problem $P = (X, Y)$ if for any instance $x \in X$, it stops with the correct *yes*/*no*-answer to the question whether $x$ belongs to $Y$. It is said to compute the function $f : X \to X'$ for two languages $X$, $X'$ over the same alphabet if for any word $x \in X$, the output is $x' = f(x) \in X'$. Accordingly, an (exact) algorithm solves an optimization problem if it computes the function mapping $x$ to the optimum solution of a given instance $x \in X$.

Mathematically, the running time of an algorithm $\mathcal{A}$ deciding/solving a decision/optimization problem with the input $x$ is defined as the number of steps that are performed by a Turing-machine realizing this algorithm before stopping. An algorithm is called *polynomial-time*, or *efficient*, if there exists a polynomial $p$ such that the running time of $\mathcal{A}$ is bounded by $p(size(x))$ for any input $x$.

The complexity class $\mathcal{P}$ contains all decision/optimization problems that can be solved efficiently. Another important complexity class is $\mathcal{NP}$, where the crucial part is that it is not required to solve a given decision problem $P = (X, Y)$ in polynomial time. Instead, it is demanded that given an instance $x \in X$ and a *certificate*, i.e., a proof that $x \in Y$, the latter can be verified in polynomial time. As every problem in $\mathcal{P}$ can be solved in polynomial time, clearly no certificate is needed. Thus, $\mathcal{P} \subseteq \mathcal{NP}$ follows. Currently, it is still unknown whether this inclusion is strict, or whether actually $\mathcal{P} = \mathcal{NP}$. The general belief, however, is $\mathcal{P} \neq \mathcal{NP}$.

The core notion of complexity theory is the *polynomial reducibility* of problems. A decision/optimization problem $P$ is called *polynomially reducible* to a decision/optimization problem $P'$ if there exists an algorithm $\mathcal{A}$ that decides/solves $P$ in polynomial time, given an oracle for the problem $P'$. An *oracle* for a decision/optimization problem is considered a black box subroutine which, for any instance of the problem, returns the correct answer/optimal solution in constant time. The reducibility of problems then implies:

1. If a decision/optimization problem $P'$ is polynomially solvable, and another decision/optimization problem $P$ polynomially reduces to $\mathcal{P}'$, then also $P$ is polynomially solvable.

2. If a decision problem $P'$ belongs to $\mathcal{NP}$, and a decision problem $P$ polynomially reduces to $P'$, then also $P$ belongs to $\mathcal{NP}$.

A decision/optimization problem is called *$\mathcal{NP}$-hard* if every problem in $\mathcal{NP}$ polynomially reduces to it. A (decision) problem that is both $\mathcal{NP}$-hard and also in $\mathcal{NP}$ is called $\mathcal{NP}$-complete. Note that $\mathcal{NP}$-completeness only applies to decision problems, and *not* to optimization problems, as the membership in $\mathcal{NP}$ is only well-defined for the former. A decision problem $P = (X, Y)$ is called *strongly $\mathcal{NP}$-complete* if it remains $\mathcal{NP}$-complete, even when restricted to instances $x \in X$ where the largest numerical number is bounded by a polynomial in the encoding length of $x$. Observe that these definitions imply

1. If any $\mathcal{NP}$-complete or $\mathcal{NP}$-hard problem is proven to be polynomially solvable, its $\mathcal{NP}$-hardness implies that all problems in $\mathcal{NP}$ are, and hence, $\mathcal{P} = \mathcal{NP}$.

2. If $P$ is $\mathcal{NP}$-complete and reducible to problem $P' \in \mathcal{NP}$, then also $P'$ is $\mathcal{NP}$-complete.

3. If $P$ is $\mathcal{NP}$-hard and reducible to problem $P'$, then also $P'$ is $\mathcal{NP}$-hard.

The remainder of this chapter summarizes the contents of this thesis and presents the necessary concepts and theoretical foundations relevant to the following three main chapters.

## 2.1 Related Literature, Contribution, and Foundations for Chapter 3

Chapter 3 is concerned with the problems (BND) and (P) in the special case in which there is only a single switch, i.e., $n = 1$, so that the consideration of vertical constraints is not really sensible. The aim is to settle the complexity of both problems for the different types of variation, with a particular interest in the polyhedral structures of their convex hulls. As the penalized problem and the time-wise problem are dealt with quite fast, the vast majority of Chapter 3 is dedicated to the problem (BND) with switch-wise variation, with a focus on obtaining a description of the convex hull of the feasible set by linear inequalities.

**In the literature,** there is not much to be found on the latter topic. This is surprising, considering that many optimization problems arising in practical applications can be modeled as combinatorial optimization problems over such a binary sequence. A typical example is the *unit commitment problem*, where the optimal scheduling of on/off-decisions for a set of generating units, or switches, over a discrete time horizon has to be determined. This problem has been studied intensely, see, e.g., [MS14] for a survey. In fact, it has been studied both from an algorithmic and a polyhedral perspective, and for different types of constraints on the scheduling of the units (or switches), or rather, different constraints on the binary incidence vectors corresponding to a unit. For example, Lee, Leung, and Margot [LLM04] and Rajan and Takriti [RT05] consider a single-unit setting and so-called *min-up/min-down constraints*, which bound from below the minimal time span that a unit has to stay on (off) after being deactivated. They fully characterize the corresponding polytope by well-structured linear inequalities and present a linear-time separation algorithm [LLM04]. Bendotti, Fouilhoux, and Rottner [BFR18] analyze polyhedral aspects for the unit commitment problem with *min-up/min-down constraints* for multiple units, they provide valid inequalities and devise an efficient Branch-and-Cut algorithm. Another polyhedral study of the unit commitment problem is given

by Damci-Kurt, Küçükyavuz, Rajan, and Atamturk [DKKRA16]. The authors focus on *ramping constraints*, which model the maximum change in production level from one time period to the next, as well as production limits and the polyhedral properties of the resulting mixed-integer problem. The convex hull of feasible solutions for the two-period case is completely described and several classes of facet-defining inequalities for the general ramp-up/ramp-down polytope are derived. The polyhedral investigation of the unit commitment problem is extended by Pan and Guan [PG16]; they investigate the polytope defined simultaneously by *min-up/min-down constraints* and *ramping constraints*. The authors completely describe the convex hull of feasible solutions for the two-and three-period case. For the multiple period case, they derive strong facet-inducing inequalities, which can be separated in polynomial time. In [PG17], the same authors continue their polyhedral study of this problem, in particular, they prove convex hull descriptions for two special cases with an arbitrary number of time periods.

Despite this extensive study of different constraints in the unit commitment context, there does not seem to exist a polyhedral investigation for the class of constraints where the overall number of changes in the binary sequence is bounded, as is the case in (BND) with (switch-wise) variation. This is especially surprising, considering the simplicity of this variation constraint. Indeed, the lack of research can neither be explained by this constraint being too artificial nor too far-fetched. In fact, this type of constraint naturally arises in many applications in which fluctuations in binary sequences are expensive. An example is the discretization of binary optimal control problems subject to a bounded total variation; see, e.g., [BGM22a, SZ21]. Here, the binary string encodes a dynamic control variable that is permitted to change its state a limited number of times within the considered time horizon. Bounded variation is a particularly relevant constraint in the context of optimal control, as most standard approaches tend to produce solutions with high fluctuation. This is due to the fact that these approaches start from an optimal solution of the continuous relaxation and then try to approximate this solution by a binary sequence in a second phase. From a practical perspective, such a chattering behavior is highly undesirable. Using finite-dimensional projections, the approach presented in [BGM22a] instead obtains an outer description of a tailored convex relaxation of some parabolic optimal control problem by exploiting the outer descriptions described here in Chapter 3. More precisely, the approach in [BGM22a] uses valid inequalities that are obtained by lifting violated cutting planes for the finite-dimensional projections. In the case of bounded total variation, these finite-dimensional projections turn out to be exactly the polytopes considered in this thesis. The approach devised in [BGM22a] thus relies on the polyhedral description and the separation algorithm in Chapter 3, and uses the latter as a subroutine. However, the results of Chapter 3 can also be applied directly when the time horizon is discretized from the beginning, as it is often the case in unit commitment problems.

**The Contribution of Chapter 3** is an extensive study of the polyhedral properties of the convex hulls of (P) and (BND).

The first section will deal with the penalized problem (P). By an equivalent reformulation of (P) as an ILP with auxiliary variables indicating the activations and deactivations of the switch, the problem will reduce to a compact linear program due to total unimodularity reasons. The continuous relaxation will then be projected onto the original variable space by Fourier-Motzkin elimination of the activation and deactivation variables. The result of the projection will be a description by an exponential number of linear inequalities, for which a simple efficient separation technique will be devised. Part of these results were published in [BH22, BH23]. More precisely, [BH22] only claimed the polynomial solvability of (P) via linear programming and the result of its projection. The proof of the reduction to an LP was later published in [BH23]. As neither of these publications considered time-wise variation, the corresponding results for the penalized problem are new to this thesis, the same holds for the details of the proofs omitted in the publications.

The second section of Chapter 3 studies the problem (BND) with a bound on the time-wise variation. Here, the problem will be shown to reduce to an unconstrained binary optimization problem, which will prove its tractability. Furthermore, the continuous relaxation of its intuitive ILP formulation will yield a description of the feasible set by a polynomial number of linear inequalities. The results of this section are new and have not been published yet.

The third and main section of Chapter 3 will study the problem (BND) with a bound on switch-wise or total variation, which are equivalent if there is only one switch. This section will include a complete polyhedral study of the problem (BND), starting with the presentation of compact extended formulations that may be continuously relaxed. These models will again be obtained by the introduction of activation and deactivation variables. A shortened and less general study of the extended models was published in [BH23]. In particular, this paper was restricted to extended models with the additional constraint that the switch be inactive, i.e., fixed to zero in the first stage. In this thesis, an extended model for the general problem without fixation in the beginning will be given and its continuous relaxation will be proven to be integral. Furthermore, it will present a complete description of the convex hull of feasible solutions in the original variable space, using an exponential number of linear inequalities. The proof of completeness will be given by the description of a primal-dual merging algorithm, the primal part of which will yield a polynomial algorithm that can be implemented to run in $\mathcal{O}(n \log n)$ time. Moreover, the linear inequality description will be proven to be facet-inducing, and the corresponding separation problem will turn out to be efficiently solvable (in time $\mathcal{O}(n)$). The fast separation was an important ingredient in the approach presented in [BGM22b], and was confirmed by the computational experiments presented therein. In their core,

the latter results have been published in [BH22, BH23]. This thesis reproduces the proofs in more detail and with additional explanations and examples.

**The Foundations of Chapter 3** are described in the following. Unless stated otherwise, they are based on [KV02, NW88].

The first concept to be explained is the notion of an *extended model*. For more details and some examples of extended models for classical combinatorial problems, the reader is referred to the survey [CCZ10]. Following the terminology in this, a *perfect formulation* refers to a description of the convex hull of an integer set by linear inequalities. The interest in perfect formulations is obvious: they allow to reduce the optimization of a linear function over an integer set to linear programming. Often times, the number of linear constraints needed in a perfect formulation is exponential in the input of the problem. Adding a polynomial number of new variables to a perfect formulation sometimes results in an equivalent formulation that needs only polynomially many linear constraints in this extended variable space. Such a model is called a *compact extended formulation*.

The integer extended formulations in Chapter 3 will be obtained by introducing additional auxiliary variables indicating the activations and deactivations of the switch. The advantage of these extended models is the fact that their integrality constraints can actually be relaxed, so that each extended model can be reduced to linear programming. Most of the time, the integrality of such a continuous relaxation will follow due to the total unimodularity of its constraint matrix.

**Definition 2.1.1.** A matrix $A \in \{-1, 0, 1\}^{m \times n}$ is totally unimodular if each minor of $A$ is $-1$, 0, or 1.

Note that by this definition, it is obvious that appending (negated) unit vectors as rows (or columns) to a totally unimodular matrix yields a totally unimodular matrix again. The same holds for appending (negated) duplicates of rows (or columns) to the matrix again. Total unimodularity is a property that under certain assumptions implies the equivalency between an integer problem and its continuous relaxation, as was shown by Hoffman and Kruskal.

**Theorem 2.1.1.** ([HK56]) An integral matrix $A \in \mathbb{Z}^{m \times n}$ is totally unimodular if and only if the polyhedron $\{x \in \mathbb{Z}^n \colon Ax \leq b, x \geq 0\}$ is integral for each integer vector $b \in \mathbb{Z}^m$.

This thesis will repeatedly use the direction '$\Rightarrow$' to derive the integrality of polyhedra. Proving that a matrix is totally unimodular via its minors is usually impractical, hence the following characterization, attributed to Ghouila-Houri, is preferred here.

**Theorem 2.1.2.** ([GH62]) An integral matrix $A = (a_{ij}) \in \mathbb{Z}^{m \times n}$ is totally unimodular if and only if for every subset $R \subseteq \{1, ..., m\}$, there is a partition $R = R_1 \cup R_2$, $R_1 \cap R_2 = \emptyset$ such that

$$\sum_{i \in R_1} a_{ij} - \sum_{i \in R_2} a_{ij} \in \{-1, 0, 1\} \qquad \text{for all } j = 1, ..., n \, .$$

By this criterion, the total unimodularity of a matrix with the following characteristic is easily recognized.

**Lemma 2.1.1.** Let $A \in \{-1, 0, 1\}^{m \times n}$ be a matrix where each column contains at most one entry equal to 1 and at most one entry equal to $-1$. Then, $A$ is totally unimodular by Ghouila-Houri's criterion with $R_1 = \{1, ..., m\}$ and $R_2 = \emptyset$.

Many combinatorial structures can be modeled with a totally unimodular constraint matrix. For example, the incidence matrix of an undirected graph $G$ is totally unimodular if and only if $G$ is bipartite. Also, the incidence matrix of a digraph is totally unimodular.

For all but one extended model in Chapter 3, the integrality of the corresponding continuous relaxation can be derived from the total unimodularity of the constraint matrix. For the one exception, the integrality of the resulting relaxation is shown directly, i.e., by proving that each vertex of the relaxation is integral. Recall that a *vertex* of a polyhedron $P = \{x \in \mathbb{R}^n \colon Ax \leq b\}$ is a *face* of dimension zero, where a face is defined as either the polyhedron $P$ itself or its intersection with a supporting hyperplane of $P$. Further recall that the *dimension* of a set $X$ is $k$ if the maximum number of affinely independent points that it contains is given by $k+1$. An alternative characterizations of a vertex of a polyhedron is given below.

**Definition 2.1.2.** Let $P = \{x \in \mathbb{R}^n \colon Ax \leq b\}$ be a polyhedron and $x \in P$. The following statements are equivalent:

1. $x \in \mathbb{R}^n$ is a vertex of $P$.
2. There exists a vector $c$ such that $\delta := \max\{c^\top x \colon x \in P\}$ is finite and $x$ is the unique optimal solution.
3. There do not exist $x^+, x^- \in P$, $x^+ \neq x^-$ such that $x = \frac{1}{2}x^+ + \frac{1}{2}x^-$.

Due to the fact that all continuous relaxations of extended models in Chapter 3 will be both integral and compact, they will yield equivalent compact LP-reformulations and thus imply the tractability of the corresponding problems. The

aim of Chapter 3, however, will be to investigate the problems from a polyhedral perspective. In particular, the aim is to find a perfect formulation in the original variable space, i.e., a description of the convex hull of the integer feasible set by linear inequalities. A perfect formulation can, for example, be obtained by projecting the polyhedron of the extended model onto the original variable space, e.g., by *Fourier-Motzkin elimination* of the auxiliary variables. The method was first proposed by Fourier [Fou26] in 1826 and later re-discovered by Motzkin in 1936. The following description is based on Section 2.8 in [BT97], where further details can be found. The key to the method is the concept of a projection, defined as follows:

**Definition 2.1.3.** If $x = (x_1, ..., x_n)$ is a vector in $\mathbb{R}^n$ and $k \leq n$, then the projection mapping $\pi_k : \mathbb{R}^n \to \mathbb{R}^k$ *projects* $x$ onto its first $k$ coordinates:

$$\pi_k(x) = \pi_k(x_1, ...., x_n) = (x_1, ..., x_k) \ .$$

The *projection* $\Pi_k(S)$ of a set $S \subseteq \mathbb{R}^n$ is defined via

$$\begin{aligned} \Pi_k(S) &= \{(x_1, ..., x_k) \colon \text{there exist } x_{k+1}, ..., x_n \text{ s.t. } (x_1, ..., x_n) \in S\} \\ &= \pi_k(S) \ . \end{aligned}$$

In each iteration of the method, one variable is eliminated by performing certain row operations on a system of linear inequalities, similar to the Gauss method to solve systems of linear equations. The result is a polyhedron in a lower dimensional space. An iteration of the method is described in the following.

Let a polyhedron $P$ be given via linear inequality constraints of the form $\sum_{j=1}^{n} a_{ij} x_j \leq b_i$ for $i = 1, ..., m$. The aim is to eliminate the variable $x_n$ and to construct the projection $\Pi_{n-1}(P)$. The inequalities can be assumed to be given by

$$\bar{a}_i^\top \bar{x} + a_{in} x_n \leq b_i \quad i = 1, ..., m \ ,$$

where $a_{ij} \in \{-1, 0, 1\}$, $\bar{a}_i := (a_{i1}, ...., a_{in-1})\mathbb{R}^{n-1}$, and $\bar{x} := (x_1, ..., x_{n-1}) \in \mathbb{R}^{n-1}$. Otherwise, divide the $i$-th inequality by $|a_{in}|$ if $a_{in} \neq 0$. Then, the set of rows is partitioned as follows, depending on the coefficients of $a_{in}$:

$$\begin{aligned} I^+ &= \{i \in \{1, ..., n\} \colon a_{in} > 0\} \ , \\ I^- &= \{i \in \{1, ..., n\} \colon a_{in} < 0\} \ , \\ I^0 &= \{i \in \{1, ..., n\} \colon a_{in} = 0\} \ . \end{aligned}$$

Finally, define the polyhedron $Q$ in $\mathbb{R}^{n-1}$ by the following set of constraints:

$$\begin{aligned} (\bar{a}_i + \bar{a}_j)^\top \bar{x} &\leq b_i + b_j \qquad \text{for all } i \in I^+, j \in I^- \\ \bar{a}_j^\top \bar{x} &\leq b_j \qquad \text{for all } j \in I^0. \end{aligned}$$

This polyhedron $Q$ constructed by the Fourier-Motzkin elimination of $x_n$ is equal to the projection $\Pi_{n-1}(P)$ of $P$.

The disadvantage of Fourier-Motzkin elimination and the reason for which it is not commonly used, is the fact that it can output a polyhedron described by a very large number of constraints. In fact, this number can become exponentially large, as shown by an example in Section 12.2 of [Sch86]. Of course, a considerable number of the linear constraints obtained by the projection can be redundant, but they are not easily identified. This motivates the following question:

Given a polyhedron $P = \{x \in \mathbb{R}^n \colon Ax \leq b\}$, which of the inequalities are actually necessary in the description of $P$ and which can be removed?

This leads the study of a polyhedron regarding its facets. Recall that a facet $F$ of $P$ is a face with $dim(F) = dim(P) - 1$.

**Proposition 2.1.1.** Let a polyhedron $P$ be described by a system of linear inequalities. For each facet of $P$, there exists an inequality in the system that induces it, and this must be included in a description of $P$. Furthermore, if an inequality induces a face of $P$ of dimension less than $dim(P) - 1$, it is redundant to the description of $P$.

In fact, the linear inequalities necessary to describe a polyhedron are always the same, regardless of its initial inequality description.

**Theorem 2.1.3.** A full dimensional polyhedron $P$ has a unique (up to scalar multiplication) *minimal representation* by a finite set of linear inequalities. In particular, for each facet $F_i$ of $P$, where $i = 1, ..., k$, let $(a^i)^\top x \leq b_i$ be an inequality inducing $F_i$. Then, a minimal description of $P$ is given by

$$\left\{x \in \mathbb{R}^n \colon (a^i)^\top x \leq b_i \quad \text{for} \quad i = 1, ..., k\right\}.$$

In Chapter 3, the above results concerning facets will be used to show that the previously derived perfect formulations are in fact minimal, as each of their linear inequalities will turn out to be facet-inducing. However, these minimal perfect formulations will generally still require an exponential number of inequalities.

While an exponential number of inequalities in the description of a polyhedron is not desirable, it does not automatically have to cause any trouble. In fact, under a certain assumption, the linear optimization over such polyhedron is still guaranteed to work efficiently. Recall that the ellipsoid method is well-known to solve linear programs in polynomial time. However, given an LP with exponentially many constraints, the ellipsoid method runs into the problem that the feasibility of a point for the LP cannot be be decided efficiently by the usual method of substitution. This

can be overcome by using an alternative way to check the feasibility, e.g., by using an efficient subroutine that implicitly checks the feasibility and returns a violated inequality when the considered element is infeasible.

This motivates the study of the *separation problem*, here regarding a polyhedron.

**Problem 2.1.1.** Separation Problem ([GLS93], Definition 2.1.4)

**Input:** A vector $y \in \mathbb{Q}^n$, a polyhedron $P \subseteq \mathbb{R}^n$
**Task:** Decide whether $y \in P$, and if not, find a hyperplane that separates $y$ from $P$, i.e., find a vector $c \in \mathbb{Q}^n$ such that $c^\top y > \max\{c^\top x \colon x \in P\}$

The following theorem is a fundamental complexity result. It states that the (linear) optimization problem and the separation problem for a polyhedron are equivalent with respect to polynomial time solvability.

**Theorem 2.1.4.** ([GLS93], Theorem 6.4.9)
For a polyhedron $P$, the linear optimization problem over $P$ polynomially reduces to the separation problem for $P$, and vice versa.

In Chapter 3, an efficient separation algorithm for each of the exponentially large perfect formulations will be devised. Due to the above equivalency, this efficient separation algorithm will also imply the tractability of the corresponding linear optimization problem.

This concludes the presentation of fundamental concepts for Chapter 3.

## 2.2 Related Literature, Contribution, and Foundations for Chapter 4

In Chapter 4, the setting of multiple switches, i.e., $n > 1$, is investigated. The aim of the chapter is to study the computational complexity of the two problems (BND) and (P) for the different types of variation. It will partly consider the trivial combinatorial structure which consists of the entire binary space $X_t := \{0,1\}$ for each $t$, but the focus will lie on the structure $X_t := \left\{x_t \in \{0,1\}^n : \sum_{i=1}^n x_t^{(i)} = K_t\right\}$ with $K_t \in \mathbb{N}_{\geq 0}$ for all $t$. The chapter will be driven by the question about the complexity of each of the problem versions with either of these two combinatorial structures as a vertical constraint.

**In the literature,** the closest related field is multi-stage optimization. Here, the input is given by a combinatorial optimization problem and a sequence of objective functions, where each of the latter corresponds to one time step. If the solution

of the underlying combinatorial problem changes over time, additional costs arise in the objective function, or alternatively, a reward is collected if the solutions do not change. The inclusion of such rewards or penalties often renders a multi-stage problem $\mathcal{NP}$-hard, even if the underlying combinatorial problem is tractable. For example, the multi-stage version of Spanning Tree is $\mathcal{NP}$-hard [GTW14] and the multi-stage version of Bipartite Perfect Matching is $n^{1-\varepsilon}$-inapproximable even for only two stages, unless $\mathcal{P} = \mathcal{NP}$ [BELP18]. Bampis et al. [BET22] study the multi-stage version of Knapsack. They present a PTAS and prove that an FPTAS cannot exist, unless $\mathcal{P} = \mathcal{NP}$, even for only two stages and uniform transition costs or rewards. The authors claim to have found the first approximation scheme for a combinatorial multi-stage problem so far, contrasting the inapproximability results for other combinatorial problems from the literature. Furthermore, they prove the problem to be pseudo-polynomial for a fixed number of stages but strongly $\mathcal{NP}$-hard in the general case, even for uniform transition costs or rewards.

There also exists literature on multi-stage problems which does not consider penalties or rewards, but instead hard constraints on the variation. Fluschnik et al. [FNRZ22] study a multi-stage variant of Vertex Cover with a restriction of time-wise variation. The input is a temporal graph in which the set of nodes remains the same, but the set of edges may change over time. The task is to decide whether there exists a vertex cover with at most $k$ vertices for each stage, while ensuring that the symmetric difference of two vertex covers of consecutive stages does not exceed a given bound. The authors show that their problem variant remains $\mathcal{NP}$-hard even if the graph of each stage has only one edge. Further, they present some fixed-parameter tractability results for the problem parameterized by the bound on the symmetric difference.

Another related research area involves reconfiguration problems, which are based on combinatorial or graph-theoretical problems. Given two solutions $R$, $Q$ to the underlying problem, the respective reconfiguration problem then asks whether there exists a sequence of feasible solutions $R = s_0, s_1, ..., s_T = Q$ that sequentially transforms $R$ into $Q$ under the constraint that solution $s_{i+1}$ can be obtained from $s_i$ by certain reconfiguration operations. As an example, Lokshtanov and Mouawad [LM17] study Stable Set Reconfiguration. Among others, they consider the variant where each $s_i$ must be a stable set of size at least $k$ in a bipartite graph and only a single vertex per step may be added or removed; the resulting decision problem is $\mathcal{NP}$-complete. This can be interpreted as a very restricted case of the problem (BND) with time-wise variation studied in this thesis, with no more than one change per time-step and an objective function that is only used to fix the first and last solution. Note that the corresponding reconfiguration problem for Selection is trivial.

Lendl et al. [LPT22] consider a generalization of a multi-stage problem where feasible solutions are bases of an arbitrary matroid; however, their study is restricted to very few stages. More precisely, they consider two stages and a time-wise variation

that is bounded from below or above, or must be matched exactly, and prove that all three variants are tractable. They briefly discuss an extension to more than two stages and show that their extension to a constraint that bounds the cardinality of the intersection of all solutions is polynomially solvable for an upper bound only, while the other two versions become $\mathcal{NP}$-hard.

**The Contribution of Chapter 4** is an extensive study of the complexity of (P) and (BND) for the three types of variation where the combinatorial structure corresponds to the feasible set of a SELECTION PROBLEM.

The first section of the chapter will deal with the penalty problem. It will begin with the presentation of an ILP model for the problem (P) without a combinatorial constraint and prove that its continuous relaxation is integral. It will turn out that the incorporation of the selection constraint into the model will not destroy its integrality, proving the problem's tractability by means of linear programming.

Subsequently, the complexity of the bounded problem (BND) is studied for a vertical selection constraint. It will turn out that the problem's complexity strongly depends on which type of variation is considered. Indeed, while the problem with time-wise variation will turn out to be tractable by a reduction to a min cost flow problem, the problem (BND) with switch-wise variation will prove to be strongly $\mathcal{NP}$-hard, even in a very restricted setting. However, the special cases with an upper bound on the variation that is uniformly equal to one, or a constant number of stages, will be tractable.

The problem (BND) with total variation and an additional selection constraint will be studied thereafter. Although tractable special cases will be presented, the general complexity of this problem will remain an open research question. The final section will briefly discuss a few related problems and their complexity.

The results concerning the problem (P), as well as the results for (BND) with time-wise and switch-wise variation are publicly available as a preprint [BH25] and currently submitted for publication.

**The Foundations of Chapter 4** that go beyond the foundations of Chapter 3 are described in the following. The combinatorial structure on which Chapter 4 will focus corresponds to the feasible set of the following problem.

**Problem 2.2.1.** SELECTION PROBLEM

| | |
|---|---|
| **Input:** | Rational profits $c_1, ..., c_n$ and some natural number $k \in \mathbb{N}_{\geq 0}$ |
| **Task:** | Find a subset $S \subseteq \{1, ..., n\}$ with $\lvert S \rvert = k$ such that $\sum_{j \in S} c_j$ is maximized |

This is a special case of the well-known Binary Knapsack Problem with uniform weights. In the latter problem, each item $i \in \{1, \dots, n\}$ has non-negative weight $a_i \in \mathbb{N}_{\geq 0}$ and the task is to find a subset of items that maximizes the profit under the constraint $\sum_{i \in S} a_i = k$.

The tractability of the selection problem follows quickly via the following trivial efficient algorithm: First, sort all profits in decreasing order. Then, select the first $k$ items with the highest profit. The correctness of this algorithm (and its polynomial running time) follows from the fact that the problem is a linear optimization problem over a *matroid*.

**Definition 2.2.1.** Let $E$ be a ground set and let $\mathcal{F}$ be a family of subsets over $E$. Then, the set system $(E, \mathcal{F})$ is an *independence system* if $\emptyset \in \mathcal{F}$, and the set $\mathcal{F}$ is closed under taking subsets, i.e., $X \subseteq Y \in \mathcal{F}$ implies $X \in \mathcal{F}$. The elements of $\mathcal{F}$ are called *independent*, and maximal independent sets are called *bases* of $X$.
An independence system is a *matroid* if $X, Y \in \mathcal{F}$ and $|X| > |Y|$ implies that there is an $x \in X \setminus Y$ with $Y \cup \{x\} \in \mathcal{F}$.

In fact, the feasible set of the selection problem corresponds to the set of bases of the well-known *uniform matroid*. A nice property of matroids is that if the membership in $\mathcal{F}$ can be verified efficiently, any linear objective can be maximized over the matroid by calling the following simple greedy algorithm. Note that given an arbitrary objective, elements with negative weight never appear in an optimum solution so that negative weights may be ignored.

**Algorithm 1:** Best-In-Greedy Algorithm (see [KV02])

**Input:** Independence system $(E, \mathcal{F})$, independence oracle, weights $c : E \to \mathbb{R}_{\geq 0}$
**Output:** A set $F \in \mathcal{F}$

```
1 Sort E = {e_1, ..., e_n} such that c(e_1) ≥ ··· ≥ c(e_n);
2 Set F := ∅;
3 for i = 1, ..., n do
4     If F ∪ {e_i} ∈ F then set F := F ∪ {e_i}
```

Here, an *independence oracle* is a placeholder for an efficient subroutine that verifies the membership in $\mathcal{F}$ for any given set $F \subseteq E$. For general independence systems, the greedy algorithm does not return an independent set with maximum objective value. In fact, matroids are exactly the structures where this always works.

**Theorem 2.2.1.** ([Rad57],[Edm71])
An independence system $(E, \mathcal{F})$ is a matroid if and only if the Best-In-Greedy

algorithm finds an optimum solution for the maximization problem for $(E, \mathcal{F})$ for arbitrary cost functions $c : E \to \mathbb{R}_{\geq 0}$.

Building on the work of Edmonds, Lawler showed that it is also possible to maximize a function over the intersection of two matroids $(E, \mathcal{F}_1)$ and $(E, \mathcal{F}_2)$, provided that the membership in $\mathcal{F}_1$ and $\mathcal{F}_2$ can be verified efficiently [Law75]. However, as soon as three matroids are considered, this problem is $\mathcal{NP}$-hard. Indeed, as shown in Chapter 8.5 of [Wel76], the $\mathcal{NP}$-hard problem Hamiltonian Path in Directed Graphs reduces to the maximization of a function over the intersection of three matroids. The polynomial reduction given therein is sketched in the following:

Given a directed graph $G = (V, E)$ with two specified vertices $s, t$, let $\delta^{in}(v) \subseteq E$ denote the set of arcs with endpoint $v$, and let $\delta^{out}(v) \subseteq E$ be the set of arcs with $v$ as start point. It is known that the independence sets defined in the following are in fact matroids:

- $\mathcal{M}_0 = (E, \mathcal{F}_0)$ where each independent set $F \in \mathcal{F}_0$ is a forest in the undirected graph obtained from $G$ by removing the arc orientations (called *graphic* matroid)

- $\mathcal{M}_1 = (E, \mathcal{F}_1)$ with $\mathcal{F}_1 = \{F \subseteq E \colon |\delta^{in}(v) \cap F| \leq 1, v \neq s\}$, i.e., each independent set $F \in \mathcal{F}_1$ corresponds to a set of arcs so that no two arcs have the same endpoint $v \neq s$ (called *partition* matroid)

- $\mathcal{M}_2 = (E, \mathcal{F}_2)$ with $\mathcal{F}_2 = \{F \subseteq E \colon |\delta^{out}(v) \cap F| \leq 1, v \neq t\}$, i.e., each independent set $F \in \mathcal{F}_1$ corresponds to a set of arcs so that no two arcs have the same start point $v \neq t$ (another *partition* matroid)

The maximization of the linear objective $\mathbb{1}_{|E|}$ over the intersection $\mathcal{F}_0 \cap \mathcal{F}_1 \cap \mathcal{F}_2$ yields an optimal value of $|V| - 1$ if and only if all three matroids have a common base or, in other words, if and only if there exists a Hamiltonian $s$-$t$-path in $G$. This shows the polynomial reduction.

Another important combinatorial structure used in Chapter 4 are matchings. Recall that a *matching* in an undirected graph $G = (V, E)$ is a subset of pairwise disjoint edges $M \subseteq E$, and a *perfect matching* is a matching with $|M| = \frac{|V|}{2}$, i.e., such that every vertex is covered by an edge in the matching. Note that while the set of all matchings in a graph $G = (V, E)$ is an independence system on the edge set $E$, and in bipartite graphs, this set can actually be obtained as the intersection of two matroids, the set of all matchings is generally *not* a matroid itself.

Given a graph with edge weights, one can consider the following problem:

**Problem 2.2.2.** Minimum Weight Perfect Matching Problem (see [KV02])

**Input:** An undirected graph $G = (V, E)$, edge weights $c : E \to \mathbb{R}$
**Task:** Find a perfect matching with minimal weight $c(M) := \sum_{e \in M} c(e)$

This problem can be solved in time $\mathcal{O}(|V|^3)$ by Edmond's Blossom algorithm, which he first proposed in [Edm65b] for uniform weights and extended it to arbitrary weights in [Edm65a]. Furthermore, it is equivalent to the so-called Maximum Weight Matching Problem, which consists in finding a matching in an undirected graph that maximizes the total weight, without demanding the matching to be perfect. For a bipartite graph, the tractability of optimizing a linear function over the set of matchings (with, or without them being perfect) follows more easily, as the continuous relaxation of the intuitive ILP formulation via the incidence matrix is integer ([Edm65a]). Indeed, the integrality of the LP relaxation can be derived from the fact that the adjacency matrix of any bipartite graph is totally unimodular. Recall that a maximum weight matching in a bipartite graph can be phrased as a maximization problem over the intersection of two partition matroids: Given a bipartite graph $G = (V \cup U, E)$, define two partition matroids $\mathcal{M}_1 = (E, \mathcal{F}_1)$, $\mathcal{M}_2 = (E, \mathcal{F}_2)$ where $\mathcal{F}_1$ contains all subsets $F \subseteq E$ so that no two edges in $F$ have a common endpoint in $V$ and $\mathcal{F}_2$ is defined analogously, only with the vertex set $U$ instead of $V$. Then, clearly the elements in the intersection of $\mathcal{F}_1$ and $\mathcal{F}_2$ are matchings in $G$. Hence, the tractability of weighted maximum matching is also implied by the tractability of weighted matroid intersection.

The last concept to be explained here are *flow networks*, the reader is referred to [AMO93] for more information.

**Definition 2.2.2.** A *flow network* is defined by a directed graph $G = (V, A)$, where each arc $(i, j) \in A$ has an associated cost $c_{ij} \in \mathbb{Q}$, a capacity $u_{ij} \in \mathbb{Q}$, and a lower bound $l_{ij} \in \mathbb{Q}$. Each node $i \in V$ has a *balance*, i.e., a number $b(i) \in \mathbb{Q}$ that describes its *supply* if $b(i) > 0$, or *demand* if $b(i) < 0$. A node with supply is called *source*, a node with demand is called *sink*. The balances satisfy $\sum_{i \in N} b(i) = 0$.

The search for a flow with minimal cost leads to the following problem.

**Problem 2.2.3.** minimum cost flow problem (see [AMO93])

**Input:** Directed network $G = (N, A)$, balances $b(i)$ for all $i \in N$, cost $c_{ij}$, capacity $u_{ij}$ and lower bound $l_{ij}$ for all $(i, j) \in A$

**Task:** Find a *flow* with minimal cost, i.e., an optimal solution of

$$\begin{aligned} \min \quad & \sum_{(i,j)\in A} c_{ij} x_{ij} \\ \text{s.t.} \quad & \sum_{\{j\colon (i,j)\in A\}} x_{ij} - \sum_{\{j\colon (j,i)\in A\}} x_{ji} = b(i) && i \in N \\ & l_{ij} \leq x_{ij} \leq u_{ij} && (i,j) \in A\,. \end{aligned}$$

Since this problem can be modeled as an LP (see above) via the totally unimodular incidence matrix of the corresponding directed graph, it always has an integer optimum solution for an arbitrary objective function if all balances, lower bounds and capacities are integer. However, there also exists a variety of combinatorial (pseudo-) polynomial optimization algorithms, such as the cycle-canceling algorithm, the successive shortest path algorithm, and also a primal-dual algorithm, to name only some. Often times, these optimization algorithms assume that the costs on the arcs are non-negative. However, many min cost flow algorithms can actually handle negative arc costs, as long as there exists no negative cycle in the directed network. For more details concerning minimum cost flows, the reader is referred to [AMO93].

## 2.3 Related Literature, Contribution, and Foundations for Chapter 5

Chapter 5 is dedicated to the study of the bounded problem (BND) for arbitrary feasible sets. For generality, it will not rely on particular properties of the feasible sets $X_t$, but instead will assume that the feasible set $X_t$ of each stage $t$ is given via a linear optimization oracle for the corresponding underlying problem. The chapter aims to investigate the complexity of (BND) in this oracle setting. It studies the relative complexity of (BND) compared to the complexity of the respective underlying combinatorial problems, i.e., the additional complexity that is introduced by the variation constraint. The main question is whether outsourcing part of the solution to the oracle allows to solve (BND) polynomially.

**In the literature,** the study of optimization problems and their algorithms with respect to oracles is considered to have begun in the late 1970s with the joint work of Nemirovski and Yudin, the English translation of which was published in 1983 [NY83]. They studied the limitations of efficient optimization methods in convex optimization based on the following black-box model for a convex optimization

problem: Given a convex feasible set $X$, the objective function $f : X \to \mathbb{R}$ is considered to be unknown. It can be be accessed through queries to either a *zero-th order oracle*, which returns the value $f(x)$ when given some $x \in X$, or a *first order oracle*, which given some $x \in X$, returns a subgradient of $f$ at $x$. In their work, the authors studied the so-called *oracle complexity* of convex optimization methods for these oracles, i.e., how many oracle calls are necessary and sufficient to find an $\varepsilon$-approximate minimum of a convex function. They performed a general analysis of optimization methods regarding their oracle complexity and compared them to each other in this regard. A detailed overview of their results can be found in [Nem95]. While the central question will be the same in Chapter 5, the oracle model that will be assumed will differ from the previously described models in the sense that the objective function will be known and instead, the feasible set will be given via a *linear optimization oracle* for the underlying problem.

Among other, the authors of [GLS93] studied such linear optimization oracles. Moreover, they also considered a separation oracle, a membership oracle, a violation oracle, as well as a validity oracle. A large part of their work in [GLS93] was dedicated to the investigation of the relationship between these oracles for convex sets under different assumptions. Their most fundamental result is the polynomial equivalency of linear optimization and separation for polyhedra (see Theorem 2.1.4).

Other recent literature continued the study of convex optimization problems in such an abstract form involving linear optimization or separation oracles. For example, Lee, Sidfort, and Wong improved the running time for finding a point in a convex set given a separation oracle [LSW15].

Regarding the oracle-polynomial solution of combinatorial problems, Boyd [Boy96] presented a cutting plane algorithm for the solution of combinatorial linear programs where the program is given by a separation oracle. He demonstrated that for problems which contain a ball whose size is polynomially bounded from below, there exists an algorithm whose running time is polynomial in the running time of the separation oracle and pseudo-polynomial in the size of the objective function. Using this generic algorithm, the cardinality versions of many combinatorial optimization problems were shown to be solvable in polynomial time.

Indeed, there are many areas in which the oracle perspective appears quite natural, such as in the context of robust optimization. Here, the oracle-approach is well-suited, since the certain underlying combinatorial problem of the robust counterparts is usually already well-studied, thus motivating an oracle-approach.

More recently, the authors of [BBDSR24] developed an oracle-based optimization algorithm for the robust counterpart of combinatorial optimization problems under discrete uncertainty in the (linear) objective. The authors consider a linear optimization oracle for the underlying certain problem based on which they devised an algorithm which can significantly outperform other existing approaches.

Due to the $\mathcal{NP}$-hardness of many robust counterparts of tractable combinatorial optimization problems, it is implied that any algorithm for such a robust counterpart that accesses the underlying certain problem via a linear optimization oracle has to call this oracle more than a polynomial number of times, unless $\mathcal{P} = \mathcal{NP}$. Buchheim [Buc20] strengthened this statement and showed that this is actually true independent of whether $\mathcal{P} = \mathcal{NP}$ or not. Beyond that, he showed that neither oracle-based pseudo-polynomial nor approximation algorithms can exist.

**The Contribution of Chapter 5** is an information-related study of the bounded problem (BND) in its most abstract form, i.e., for arbitrary feasible sets. It lives in an oracle-setting and tackles the question about the degree to which the problem (BND) owes its complexity to the bound on the variation. More precisely, it studies the relative complexity of the problem (BND) with respect to the complexity of the underlying problem (LO), where the latter is given via a linear optimization oracle.

The first section will make the assumption that the number of switches $n$ is fixed and will develop an oracle-algorithm based on a dynamic programming for each type of variation. For a fixed number of switches, each algorithm will be oracle-polynomial. In fact, the algorithms for total variation and time-wise variation will be oracle-$FPT$ algorithms in the parameter $n$.

The second section will pursue a complexity-theoretical approach and derive as its main result that, unless $\mathcal{P} = \mathcal{NP}$, a polynomial number of oracle calls for the underlying problem does not suffice to solve (BND) with switch-wise variation, even if the feasible sets of (BND) do not change over time and the bound on the variation is uniformly equal to two. Indeed, this will be an immediate consequence of the $\mathcal{NP}$-hardness of (BND) for switch-wise variation and a selection structure, obtained in the preceding chapter. Furthermore, the problem (BND) with time-wise variation, and parameterized in the number of stages, will be proven as $W[1]$-hard, even if the bound on the variation is uniformly one and the feasible set does not change over time.

In the third section, the main result of the chapter will be presented. By taking more of an information-related approach, it will be shown that the problem (BND) cannot be solved using only a polynomial number of calls to the oracle for the underlying problem. This will hold regardless of the type of variation and even in the quite restricted setting in which there are only two stages with the same feasible set and a bound on the variation that is uniformly one. Moreover, a polynomial number of oracle calls will not even suffice to decide the feasibility of (BND) for two stages and with a variation bound being uniformly zero, i.e., when the solution needs to be constant over all stages. Due to this result, the remaining part of the chapter will propose an oracle-algorithm for time-wise and total variation that is based on a different, more powerful oracle for the so-called Optimal Constant Extension Problem which will be closely related to (BND). Indeed, using this alternative or-

acle, the proposed algorithm runs in polynomial time if both the variation bound $S$ and the number of stages $T$ is fixed. For a large number of combinatorial structures, the Optimal Constant Extension Problem will turn out to be tractable which, in turn, will yield the tractability of (BND) for those structures for a fixed number of stages and a fixed variation bound.

**The Foundations of Chapter 5** that go beyond the foundations of Chapter 3 and 4 are explained in the following.

The notion of an *oracle* as a black box subroutine that returns the correct answer/solution for a certain task in an instant has already come up before. Recall that for any polynomial reduction of a problem $P$ to a problem $P'$, an oracle for the decision/optimization problem $P'$ is assumed. Of course, there also exist other oracles. A *membership oracle* for a set $X$ is an oracle which is given an element $x$ and verifies its membership in $X$. Observe that the independence oracle from the previous section is actually a membership oracle for the set $\mathcal{F}$ of an independence system $(E, \mathcal{F})$. Another commonly used oracle is a *separation oracle* for a set $X$, which solves the corresponding separation problem over $X$ when given an element $x$.

In Chapter 5, a *linear optimization oracle* for the underlying problem (LO) is assumed. As mentioned before, this can be considered a black box method for a respective set $X_t$ that returns an optimal solution of the linear problem (LO) in an instant when given any objective $c \in \mathbb{R}^n$, regardless of the general complexity of (LO). Recall that the linear optimization over a finite set $X$ is equivalent to the optimization over the convex hull $\text{conv}(X)$, which is a polyhedron. Thus, in the setting of Chapter 5, the polynomial equivalency between linear optimization and separation will hold (Theorem 2.1.4). In particular, this means that every statement in the chapter that assumes a linear optimization oracle will also hold with a separation oracle for the respective convex hull.

Following the definition in [GLS93], an *oracle algorithm* is an algorithm that contains some black box subroutine (the call of the oracle), where the latter is performed in an instant. For an oracle, a usual assumption is that there always exists an answer given by the oracle whose size is polynomially bounded in the size of the question. In Chapter 5, this is trivially satisfied by the binarity of the feasible sets $X_t$. If an oracle algorithm runs in time polynomial in the input size, it is called an *oracle-polynomial algorithm*, where the oracle is clear from the context. The study of oracle algorithms is motivated by the question of whether outsourcing part of the problem's solution to the oracle allows for a polynomial solution, i.e., it is interested in the complexity of the problem relative to the complexity of the subproblem handled by the oracle.

In Chapter 5, the development of oracle-polynomial algorithms will be shown to be possible under the assumption that the number of switches $n$ is fixed. This will build a bridge to the field of *parameterized tractability*, the study how different parameters influence the complexity of a problem. For a comprehensive introduction

into this theory with appropriate notions of reduction and completeness, the reader is referred to the work of Downey and Fellow [DF13]. The following description is restricted to the concepts needed in this thesis and mostly based on [CFK+15]. Its aim is to give an intuition of the theory rather than describing each formal detail.

**Definition 2.3.1.** A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$, where $\Sigma$ is a fixed, finite alphabet. For an instance $(x, k) \in \Sigma^* \times \mathbb{N}$, $k$ is called the *parameter*.

A parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is called *fixed-parameter tractable FPT*) if there exists an algorithm $\mathcal{A}$ (called fixed-parameter algorithm), a computable function $f : \mathbb{N} \to \mathbb{N}$, and a constant $c$ such that, given $(x, k) \in \Sigma^* \times \mathbb{N}$, the algorithm $\mathcal{A}$ correctly decides whether $(x, k) \in L$ in time bounded by $f(k) \cdot size(x)^c$. The complexity class containing all fixed-parameter tractable problems is called $FPT$.

In parameterized complexity theory, the notation $\mathcal{O}^*$ suppresses factors polynomial in the input size, i.e., a running time $\mathcal{O}^*(f(k))$ means that it is upper bounded by $f(k) \cdot n^{\mathcal{O}(1)}$, where $n$ is the input size. The complexity class $FPT$ can be considered the parameterized analogue to the complexity class $\mathcal{P}$.

**Definition 2.3.2.** A parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is called *slice-wise polynomial* ($XP$) if there exists an algorithm $\mathcal{A}$ and two computable functions $f, g : \mathbb{N} \to \mathbb{N}$ such that, given $(x, k) \in \Sigma^* \times \mathbb{N}$, the algorithm $\mathcal{A}$ correctly decides whether $(x, k) \in L$ in time bounded by $f(k) \cdot size(x)^{g(k)}$. The complexity class containing all slice-wise polynomial problems is called $XP$.

Note that these definitions can be generalized to multiple parameters. In parameterized complexity theory, the analogue to a polynomial time reduction is the concept of an $FPT$-time reduction.

**Definition 2.3.3.** A *parameterized reduction* from an instance $(x, k)$ of a parameterized problem to an instance $(x', k')$ of second parameterized problem is a reduction that can be performed in $FPT$-time, that is, in time $\mathcal{O}(f(k) \cdot size(x)^{\mathcal{O}(1)})$ such that the parameter $k'$ only depends on $k$; in other words, $k' \leq g(k)$ for a computable function $g$.

Just like the class $FPT$ can be considered the parameterized analogue to the class $\mathcal{P}$, the complexity class $W[1]$ can be considered the analogue to the complexity class $\mathcal{NP}$. Roughly speaking, a decision problem is in $W[1]$ if a *certificate* for it can be verified efficiently as long as the parameter is small. It contains all *hard* parameterized problems that are analyzed in their complexity with respect to the

parameter $k$. Here, the hardness of a problem is measured with respect to the difficulty to satisfy a *boolean circuit* in which only $k$ variables may be set to one. For more information on the idea of boolean circuits, the reader is referred to literature on parameterized complexity, e.g., [DF13]. It trivially holds $FPT \subseteq W[1]$ and it is generally assumed, although not yet proven, that the inclusion is strict. Note that a $W[1]$-hard parameterized problem is fixed-parameter intractable, unless $FPT$=W[1]. Thus, the $W[1]$-hardness of a problem can be used to show that it is not in $FPT$ (unless $FPT = W[1]$).

In Chapter 5, an oracle-polynomial algorithm for (BND) with time-wise and total variation will be devised. Under certain conditions, this algorithm will then yield an efficient algorithm for combinatorial problems for which a particular optimization problem related to (BND) is tractable. The tractability of the latter problem can be proven for some classical combinatorial problems which are described in the following.

**Problem 2.3.1.** Maximum Weight Stable Set

**Input:** An undirected graph $G = (V, E)$ with vertex weights $w : V \to \mathbb{R}$
**Task:** Find a *stable set*, i.e., a subset $S \subseteq V$ where $(v, w) \notin E$ for all $v, w \in S$, such that the weight $w(S) := \sum_{v \in S} w(v)$ is maximized

**Problem 2.3.2.** Minimum Weight Vertex Cover

**Input:** An undirected graph $G = (V, E)$ with vertex weights $w : V \to \mathbb{R}$
**Task:** Find a *vertex cover*, i.e., a subset $S \subseteq V$ where each $e \in E$ is incident with some $v \in S$ such that the weight $w(S) := \sum_{v \in S} w(v)$ is minimized

While both these problems are (strongly) $\mathcal{NP}$-hard in general graphs, they are tractable when restricted to certain classes of graphs such as trees, cycles, perfect, and bipartite graphs. Note that the complement of a maximum (cardinality) stable set is a minimum (cardinality) vertex cover.

In fact, these two problems can be recognized as members of more abstract classes of combinatorial problems formalized below.

**Problem 2.3.3.** Packing Problem

**Input:** A discrete ground set $V$ with weights $w(v) \in \mathbb{R}$ for all $v \in V$, and a finite collection of subsets $S_i \subseteq V$, $i = 1, ..., m$
**Task:** Find a subset $V' \subseteq V$ with $|S_i \cap V'| \leq 1$ for all $i = 1, ..., m$ that maximizes the weight $w(V') = \sum_{v \in V'} w(v)$

Obviously, Stable Set is a packing problem with the ground set $V$ of all vertices of the graph where each sets $S_i \in V^2$ corresponds to an edge of the graph, given by its vertices. Another example is the maximum weight matching problem in a in a graph $G = (V, E)$. This is a packing problem with the ground set $E$ of all edges of the graph where each set $S_i$ contains all edges that are adjacent to an edge $e_i \in E$.

The problem Minimum Vertex Cover, however, belongs to another class.

**Problem 2.3.4.** Covering Problem

| | |
|---|---|
| **Input:** | A discrete ground set $V$ with weights $w(v) \in \mathbb{R}$ for all $v \in V$, and a finite collection of subsets $S_i \subseteq V$, $i = 1, ..., m$ |
| **Task:** | Find a subset $V' \subseteq V$ with $\lvert S_i \cap V' \rvert \geq 1$ for all $i = 1, ..., m$ that minimizes the weight $w(V') = \sum_{v \in V'} w(v)$ |

It is obvious that Minimum Vertex Cover is a covering problem on the ground set $V$ of all vertices of the graph where $S_i \in V^2$ corresponds again to an edge of the graph, given by its vertices.

# Chapter 3

# Bounded Variation in a Binary Switch

This chapter is dedicated to the special cases of (BND) and (P) in which $n = 1$, i.e., in which there exists only a single switch. Consequently, the vertical dimension vanishes in this chapter, which makes this special case much simpler. Indeed, it will turn out that the penalty and the bounded problem are efficiently solvable, no matter which of the given three types of variation is considered.

This chapter is divided into three main sections. The first section will deal with the penalty problem and will show that it almost immediately reduces to a linear program by means of a compact extended formulation. It will also be possible to obtain a linear inequality description of the convex hull of feasible solutions by a projection onto the original space. However, the result of the projection will not be compact, as it will consist of exponentially many inequalities. The second section will study the bound on time-wise variation, which will again almost immediately reduce to a linear program. Finally, the third section will investigate the bound on switch-wise (or total) variation. The complexity of this case will quickly be settled by the existence of an efficient dynamic programming scheme and, as it turns out, there also exists a compact LP-formulation of the problem in an extended variable space. The focus of this section will be on a polyhedral study of the problem in the original space. As a result, it will yield an (exponentially large) LP-reformulation in the original space, a corresponding separation algorithm, and a second optimization algorithm that will be much faster than the dynamic programming scheme.

**Remark 3.0.1.** According to Notation 2.0.1, the superscript (1) in the variable $x_t^{(1)}$ denotes the index of the single switch that is considered in this chapter. For better readability, the superscript (1) in $x$ is omitted for the remainder of this chapter.

## 3.1 Penalized Variation

In the single-switch setting, the penalty problem (P) reads

$$(\mathrm{P}^{(1)}) \qquad \begin{cases} \max & c^\top x - \lambda^\top \mathrm{Var}(x) \\ \text{s.t.} & x \in \{0,1\}^{T+1}\,, \end{cases}$$

where depending on the type of variation, $\lambda \in \mathbb{R}^r_{>0}$ denotes a penalty vector of the appropriate dimension. The latter problem can be linearized by introducing auxiliary variables $u \in \mathbb{R}^T_{\geq 0}$ and $d \in \mathbb{R}^T_{\geq 0}$, which indicate whether the switch is activated or deactivated at stage $t$, and which receive individual penalty vectors $\mu, \delta \in \mathbb{R}^T_{>0}$. This way, one actually obtains a more powerful, i.e., more general problem than ($\mathrm{P}^{(1)}$), due to the possibility to assign more (or less) priority to whether the switch is activated or deactivated at a certain stage. In the original problem, the activation and deactivation both incur the same penalty. The problem ($\mathrm{P}^{(1)}$) is therefore equivalent to an instance of the following problem with $\mu := \delta := \lambda \in \mathbb{R}^T_{>0}$ for time-wise variation or, respectively, $\mu_t := \delta_t := \lambda \in \mathbb{R}_{>0}$ for all $t = 1, ..., T$ for switch-wise and total variation:

$$(\mathrm{P}^{(1)}\text{-LIN}) \qquad \begin{cases} \max & c^\top x - \mu^\top u - \delta^\top d & \\ \text{s.t.} & x_t - x_{t-1} \leq u_t & t = 1, ..., T \\ & x_{t-1} - x_t \leq d_t & t = 1, ..., T \\ & x \in \{0,1\}^{T+1},\ u, d \in \mathbb{R}^T_{\geq 0} & \end{cases}$$

The advantage of the above formulation it is easy to see:

**Lemma 3.1.1.** The continuous relaxation of ($\mathrm{P}^{(1)}$-LIN) is integral. Furthermore, the inclusion of additional constraints $x_t = b$ with $b \in \{0,1\}$ and $t \in \{0, ..., T\}$ does not destroy the integrality of the resulting continuous relaxation.

*Proof.* The matrix that models the constraints of ($\mathrm{P}^{(1)}$-LIN) has the form

$$\begin{pmatrix} A & -I_T & 0 \\ -A & 0 & -I_T \end{pmatrix},$$

where the columns of $A \in \{0,1\}^{T \times (T+1)}$ correspond to the components of the $x$ variables. Since the matrix $A$ is totally unimodular according to Ghouila-Houri's criterion, where one of the partition sets contains all rows while the other is empty, it follows that the entire constraint matrix is totally unimodular, too. Furthermore, note that since the inclusion of a constraint $x_t = b$ for some $b \in \{0,1\}$ and $t \in \{0, ..., T\}$ only adds (negated) unit vectors as rows to the former constraint matrix, one may

include as many such constraints as desired without losing the total unimodularity. Now, as the right hand side is integer, the integrality of the continuous relaxation follows. □

This settles the tractability of ($P^{(1)}$), as it reduces to a linear program. However, it is worthwhile to consider a second formulation of ($P^{(1)}$), which introduces further auxiliary variables $S \in \mathbb{R}^r$ and will allow the elimination of the variables $u$ and $d$. Note that Fourier-Motzkin elimination of the activation and deactivation variables $u$ and $d$ in the previous formulation ($P^{(1)}$-LIN) would encounter the problem that the auxiliary variables also appear in the objective. This is no longer the case for the following second formulation of ($P^{(1)}$), which also seems to be more closely related to the respective bounded problem versions that will be addressed in Sections 3.2 and 3.3. Therefore, it simplifies the recognition of similarities and differences between the penalized and the bounded problem versions. It is given by

$$(P^{(1)}\text{-S}) \qquad \begin{cases} \max & c^\top x - \lambda^\top S & \\ \text{s.t.} & B(u+d) \leq S & \\ & x_t - x_{t-1} \leq u_t & t = 1, ..., T \\ & x_{t-1} - x_t \leq d_t & t = 1, ..., T \\ & x \in \{0,1\}^{T+1}, \;\; u, d \in \mathbb{R}^T_{\geq 0}, \; S \in \mathbb{R}^\ell \,, & \end{cases}$$

where for time-wise variation, the matrix $B$ is defined by $B := I_T$ so that the inequality $B(u+d) \leq S$ holds component-wise, whereas $B := (1, .., 1) \in \mathbb{R}^{1\times T}$ for total and switch-wise variation. Recall that the latter two types of variation coincide for $n = 1$. Here, the corresponding constraint matrix is *not* totally unimodular, as the following example shows.

**Example 3.1.1.** Consider the formulation ($P^{(1)}$-S) for switch-wise/total variation and $T = 2$. Then, the constraint matrix reads

$$\left(\begin{array}{cccccccc} x_0 & x_1 & x_2 & d_1 & d_2 & u_1 & u_2 & S \\ \hline 1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & -1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & -1 \end{array}\right)$$

Then, the submatrix

$$\begin{pmatrix} -1 & -1 & 0 \\ -1 & 0 & -1 \\ 0 & 1 & 1 \end{pmatrix}$$

that consists of rows $1, 4, 5$ and columns $2, 4, 7$ has determinant $-2$, hence the matrix cannot be totally unimodular.

Of course, the fact that the constraint matrix of ($P^{(1)}$-S) is not totally unimodular does not imply that the respective continuous relaxation is not integral. Indeed, the continuous relaxation turns out to be integral nonetheless.

**Lemma 3.1.2.** The problem ($P^{(1)}$-S) is equivalent to ($P^{(1)}$). Furthermore, its continuous relaxation is integral, even if additional fixation constraints $x_t = b$ with $b \in \{0,1\}$ and $t \in \{0,...,T\}$ are included.

*Proof.* Consider an optimal solution $(x^*, u^*, d^*, S^*)$ of ($P^{(1)}$-S). Since $\lambda$ is positive, the variables $u^*, d^*, S^*$ must achieve their lower bounds, i.e., $S^* = B(u^* + d^*)$, $u_t^* = \max\{x_t - x_{t-1}, 0\}$ and $d_t^* = \max\{x_{t-1} - x_t, 0\}$. In particular, this implies the binarity of $u^*$, $d^*$, and the integrality of $S^*$. The partial solution $(x^*, u^*, d^*)$ is obviously feasible for ($P^{(1)}$-LIN) with $\mu := \delta := \lambda^\top B$, and by definition of $B$, these two solutions achieve the same objective value.

Conversely, given any optimal solution $(x^*, u^*, d^*)$ of ($P^{(1)}$-LIN), this must be binary due to $\lambda$ being positive. Defining $\mu := \delta := \lambda^\top B$ and $S^* := B(u^* + d^*)$ yields a feasible solution $(x^*, u^*, d^*, S^*)$ for ($P^{(1)}$-S) with the same objective value. In other words, ($P^{(1)}$-S) is equivalent to ($P^{(1)}$-LIN) with $\mu := \delta := \lambda^\top B$. Since ($P^{(1)}$-S) is equivalent to ($P^{(1)}$), the first claim follows.

Regarding the second claim, let $F$ denote the feasible set of ($P^{(1)}$-S) and let $F^{relax}$ denote the feasible set of its corresponding continuous relaxation. The proof will now show $\text{conv}(F) = F^{relax}$, where the inclusion $\text{conv}(F) \subseteq F^{relax}$ is trivial due to $F \subseteq F^{relax}$. Thus, it remains to show $\text{conv}(F) \supseteq F^{relax}$:

Given some $(x, u, d, S) \in F^{relax}$, first determine a permutation $\sigma$ of $\{0, 1, ..., T\}$ such that $x_{\sigma(0)} \leq x_{\sigma(1)} \leq ... \leq x_{\sigma(T)}$ and define binary vectors $x^j \in \{0,1\}^{T+1}$ for $j = 0, ..., T$ by

$$x^0 := \mathbb{1}_{T+1} \quad \text{and} \quad x_t^j := \begin{cases} 0 & x_{\sigma(t)} < x_{\sigma(j)} \\ 1 & x_{\sigma(t)} \geq x_{\sigma(j)} \end{cases} \quad \text{for } j = 1, ..., T\ .$$

Further, define binary vectors $u^j, d^j \in \{0,1\}^T$ and integral vectors $S_j \in \mathbb{N}^r$ for all $j = 0, ..., T$ by

$$u_t^j := x_t^j - x_{t-1}^j,\ d_t^j := x_{t-1}^j - x_t^j \quad \text{for } t \in \{0, ..., T\} \quad \text{and} \quad S^j := B(u^j + d^j)\ .$$

It is easy to verify that $(x^j, u^j, d^j, S^j) \in F$ for $j = 0, ..., T$.
For $\lambda_0 := 1 - x_{\sigma(T)}$,$\lambda_1 := x_{\sigma(1)}$ and $\lambda_j := x_{\sigma(j)} - x_{\sigma(j-1)}$ for $j \geq 2$, one obtains

$$\sum_{j=0}^{T} \lambda_j (x^j, u^j, d^j, S^j) = (x, \tilde{u}, \tilde{d}, \tilde{S})\ ,$$

with $\tilde{u}_t = x_t - x_{t-1}$, $\tilde{d}_t = x_{t-1} - x_t$ for $t \in \{1, ..., T\}$ and $\tilde{S} = B(\tilde{u} + \tilde{d})$.
Note that it holds $u_t \geq x_t - x_{t-1} = \tilde{u}_t$ and $d_t \geq x_{t-1} - x_t = \tilde{d}_t$ which follows from

$(x, u, d, S) \in F^{relax}$. Thus, for all $j = 0, ..., T$, define $\bar{u}^j := u^j + u - \tilde{u}$, $\bar{d}^j := d^j + d - \tilde{d}$ and $\bar{S}^j := S^j + S - \tilde{S}$. Then, it still holds $(x^j, \bar{u}^j, \bar{d}^j, \bar{S}^j) \in F^{relax}$, since

$$\bar{u}_t^j \geq u_t^j = x_t^j - x_{t-1}^j,\ \bar{d}_t^j \geq d_t^j = x_{t-1}^j - x_t^j \qquad \text{due to } u \geq \tilde{u}, d \geq \tilde{d}\ ,$$

and $\bar{S}^j \geq B(\bar{u}^j + \bar{d}^j)$, since $\bar{S}^j = S^j + S - \tilde{S}$ and

$$B(\bar{u}^j + \bar{d}^j) = \underbrace{B(u^j + d^j)}_{=S^j} + \underbrace{B(u + d)}_{\geq S} - \underbrace{B(\tilde{u} + \tilde{d})}_{\tilde{S}}\ .$$

Then, $\sum_{j=0}^{T} \lambda_j = 1$ finally yields

$$\sum_{j=0}^{T} \lambda_j (x^j, \bar{u}^j, \bar{d}^j, \bar{S}^j) = (x, \tilde{u}, \tilde{d}, \tilde{S}) + (\mathbb{0}_{T+1}, u - \tilde{u}, d - \tilde{d}, S - \tilde{S}) = (x, u, d, S)\ .$$

This concludes the proof. □

Consequently, the continuous relaxation of ($\mathrm{P}^{(1)}$-S) yields a tractable reformulation of ($\mathrm{P}^{(1)}$) in an extended solution space that includes the additional variables $u$, $d$, and $S$. As mentioned before, this formulation allows to equivalently state problem ($\mathrm{P}^{(1)}$) as a linear program in the space of only the variables $x$ and $S$ by Fourier-Motzkin elimination of the variables $u$ and $d$. Depending on the considered type of variation, this yields two different projections onto this reduced space.

**Lemma 3.1.3.** For time-wise variation, the convex hull of feasible solutions of ($\mathrm{P}^{(1)}$), i.e., the projection of ($\mathrm{P}^{(1)}$-S) with $B := I_T$ onto the $x, S$-space is given by the linear constraints $0 \leq x_t \leq 1$ for $t = 0, ..., T$, as well as

$$x_t - x_{t-1} \leq S_t \quad \text{and} \quad x_{t-1} - x_t \leq S_t \quad \text{for all } t = 1, ..., T.$$

If the state of the switch is fixed to $b \in \{0, 1\}$ at some stage $t$, then a complete description in the reduced space is given by the above set of constraints and the corresponding constraint for the fixation $x_t = b$.

*Proof.* The proof is straightforward: By Fourier-Motzkin elimination of the auxiliary variables $u$ and $d$ in the continuous relaxation of ($\mathrm{P}^{(1)}$-S), the resulting projection is obtained. The elimination process is briefly sketched in the following.

First, eliminate a variable $u_j$ for some $j \in \{1, ..., T\}$. This variable occurs negatively in the constraint $-u_j \leq 0$ and in the constraint $-x_{j-1} + x_j - u_j \leq 0$, and it occurs positively in the constraint $u_j + d_j \leq S_j$. Fourier-Motzkin of $u_j$ then gets rid of the former constraints and replaces them with the following new linear constraints

$$d_j \leq S_j \quad \text{and} \quad -x_{j-1} + x_j + d_j \leq S_j\ ,$$

while all constraints that do not contain $u_j$ stay unchanged. Therefore, after eliminating all variables $u$, the following linear constraints remain:

$$\begin{aligned} d_t &\le S_t, & -x_{t-1} + x_t + d_t &\le S_t & t &= 1, ..., T \\ -d_t &\le 0, & x_{t-1} - x_t - d_t &\le 0 & t &= 1, ..., T \\ & & 0 \le x_t &\le 1 & t &= 0, ..., T \end{aligned}$$

Then, elimination of the variables $d$ shows the claim. It is obvious that the linear inequalities corresponding to a fixation constraint are not affected by the elimination process, consequently they remain untouched throughout the projection. □

While the previous projection of ($P^{(1)}$-S) onto the original space remains compact if time-wise variation is considered, its projection becomes exponentially large if switch-wise or total variation is considered.

**Lemma 3.1.4.** For switch-wise and total variation, the convex hull of feasible solutions of ($P^{(1)}$), i.e., the projection of ($P^{(1)}$-S) with $B := (1, .., 1) \in \mathbb{R}^{1\times T}$ onto the $x, S$-space is completely described by the linear constraints

$$0 \le x_t \le 1 \quad \text{for all} \quad t = 0, ..., T \qquad \text{and}$$

$$\sum_{t=1}^{T} (-1)^{\sigma(t)} (x_{t-1} - x_t) \le S \quad \text{for all} \quad \sigma \colon \{1, ..., T\} \to \{0, 1\} \, .$$

If the state of the switch is fixed to $b \in \{0, 1\}$ at some stage $t$, then a complete description in the reduced space is given by the above set of constraints together with the corresponding constraint $x_t = b$.

*Proof.* The proof is again straightforward and hence only outlined.

Consider the continuous relaxation ($P^{(1)}$-S) for switch-wise and total variation. Let the Fourier-Motzkin elimination begin to eliminate to an arbitrary first variable $u_j$ with $j \in \{1, ..., T\}$. This variable occurs negatively in the non-negativity constraint $-u_j \le 0$ and the constraint $-x_{j-1} + x_j - u_j \le 0$, and it occurs positively in the constraint $\sum_{t=1}^{T} u_t + \sum_{t=1}^{T} d_t \le S$. Thus, elimination of $u_j$ yields the following new linear constraints

$$\sum_{t=1, t\neq j}^{T} u_t + \sum_{t=1}^{T} d_t \le S \quad \text{and} \quad -x_{j-1} + x_j + \sum_{t=1, t\neq j}^{T} u_t + \sum_{t=1}^{T} d_t \le S \, ,$$

while all constraints that do not contain $u_j$ stay unchanged. When eliminating a second variable $u_\ell$ with $\ell \neq j \in \{1, ..., T\}$, this variable occurs negatively in $-u_\ell \le 0$ and $-x_{\ell-1} + x_\ell - u_\ell \le 0$, and it occurs positively in the two new constraints above.

Elimination of $u_\ell$ then yields the following new linear constraints that replace the former constraints involving $u_\ell$:

$$-x_{j-1} + x_j + \sum_{t \neq j,\ell} u_t + \sum_{t=1}^{T} d_t \leq S, \quad -x_{\ell-1} + x_\ell + \sum_{t \neq j,\ell} u_t + \sum_{t=1}^{T} d_t \leq S,$$

$$\sum_{t \neq j,\ell} u_t + \sum_{t=1}^{T} d_t \leq S, \quad \text{and} \quad -x_{j-1} + x_j - x_{\ell-1} + x_\ell + \sum_{t \neq j,\ell} u_t + \sum_{t=1}^{T} d_t \leq S\ .$$

All constraints that do not contain $u_\ell$ stay unchanged. It is not hard to see that elimination of all auxiliary variables $u$ and $d$ results in the set of linear constraints given in the lemma. Again, it is obvious that the linear inequalities corresponding to a fixation constraint are not affected by the elimination process, so they remain untouched throughout the projection. □

Note that the number of inequalities given in Lemma 3.1.4 for the convex hull of ($P^{(1)}$) is exponential in $T$, whereas it is linear in $T$ in the extended models ($P^{(1)}$-LIN) and ($P^{(1)}$-S). Nevertheless, these inequalities can be separated efficiently:

**Corollary 3.1.1.** For switch-wise and total variation, consider the linear inequalities described in Lemma 3.1.4. Given some $(\bar{x}, \bar{S})$, the corresponding separation problem can be solved efficiently. Indeed, it is easily checked by substitution whether the trivial constraints $0 \leq x_t \leq 1$ are satisfied for all $t$. For the second type of constraints, a most violated inequality is obtained by setting $\sigma(t) := 1$ if and only if $\bar{x}_{t-1} \leq \bar{x}_t$.

This concludes the study of the penalty problem ($P^{(1)}$). As a result of this section, the problem is very easy to deal with, either by compact extended formulation or by a fast separation algorithm in the original space. The following short section will deal with the bounded problem (BND) with time-wise variation which will also not be too difficult.

## 3.2 Bounded Time-Wise Variation

For a single switch, the bounded problem (BND) with time-wise variation reads

$$(\mathrm{BND}^{(1)}_{tw}) \qquad \begin{cases} \max & c^\top x \\ \text{s.t.} & |x_t - x_{t-1}| \leq S_t \quad t = 1, ..., T\ , \\ & x \in \{0,1\}^{T+1} \end{cases}$$

where $S \in \mathbb{N}^T_{\geq 0}$ is no longer a variable, but part of the problem input. Without loss of generality, let $S \in \{0,1\}^T$, as any integer time-wise variation bound $S_t$ larger than

zero is redundant. Then, multiple consecutive zero-bounds $S_j = S_{j+1} = ... = S_\ell = 0$ imply that it must hold $x_{j-1} = x_j = ... = x_\ell \in \{0,1\}$ so that the problem $(\mathrm{BND}^{(1)}_{tw})$ can be rephrased as follows:

Given some subsets $\Delta_j \subseteq \{0,...,T\}$, $j = 1,...,\kappa$, containing sequences of consecutive stages such that all subsets are pairwise disjoint, determine a binary vector $x \in \{0,1\}^{T+1}$ that maximizes the objective function $c^\top x$ so that $x$ is constant over each given sequence $\Delta_j$,$j = 1,...,\kappa$, or, more formally, so that for each $j = 1,...,\kappa$, there exists a $y_j \in \{0,1\}$ so that $x_t = y_j$ for all $t \in \Delta_j$. This gives rise to the following equivalent linear reformulation of $(\mathrm{BND}^{(1)}_{tw})$

$$\begin{aligned} \max \quad & \sum_{t \notin \Delta} c_t x_t + \sum_{j=1}^{\kappa} \sum_{t \in \Delta_j} c_t x_t \\ \text{s.t.} \quad & x_t = y_j && j = 1,...,\kappa,\ t \in \Delta_j \\ & x \in \{0,1\}^{T+1} \\ & y \in \{0,1\}^{\kappa}\ , \end{aligned}$$

where $\Delta := \bigcup_{j=1}^{\kappa} \Delta_j$. This problem is an unconstrained binary optimization problem, as it can be equivalently stated as follows

$$\begin{aligned} \max \quad & \sum_{t \notin \Delta} c_t x_t + \sum_{j=1}^{\kappa} C_j y_j \\ \text{s.t.} \quad & x_t \in \{0,1\} && t \notin \Delta, \\ & y_j \in \{0,1\} && j = 1,...,\kappa\ , \end{aligned}$$

where $C_j := \sum_{t \in \Delta_j} c_t$. It can be solved in linear time by setting a variable $x_t$ (or $y_j$, resp.) to one if and only if the corresponding objective coefficient $c_t$ (or $C_j$, resp.) is positive. If the problem $(\mathrm{BND}^{(1)}_{tw})$ includes a fixation constraint $x_t = b$ for some stage $t \in \{0,...,T\}$, then this can easily be included into this simple optimization algorithm, since it just predetermines the value of the corresponding variable $x_t$ and takes away the task to assign a value to it based on the objective coefficient. Of course, if $t \in \Delta_j$ for some $j \in \{1,...,\kappa\}$, then this fixation extends to all stages in the respective sequence $\Delta_j$. Note that the identification of the sequences $\Delta_1,...,\Delta_\kappa$ needs linear time $\mathcal{O}(T)$. Thus follows

**Lemma 3.2.1.** The problem $(\mathrm{BND}^{(1)}_{tw})$ can be solved in linear time. The same holds true if additional constraints $x_t = b$ with $b \in \{0,1\}$ for $t \in \{0,...,T\}$ are included.

Observe that the latter formulation of $(\mathrm{BND}^{(1)}_{tw})$ also yields a linear programming formulation, since the integrality constraint may be relaxed. However, there exists an interesting connection between the convex hull of feasible solutions for the

bounded problem ($\text{BND}^{(1)}_{tw}$) and the convex hull of feasible solutions for the penalized problem ($\text{P}^{(1)}$) that is more obvious in the following second formulation of ($\text{BND}^{(1)}_{tw}$), obtained by using classical linearization of the absolute expressions in ($\text{BND}^{(1)}_{tw}$). The classical linearization introduces auxiliary variables $z_t$ that are bounded from below by $x_{t-1} - x_t$ and $-x_{t-1} + x_t$ as well as the constraints $z_t \leq S_t$ for all $t = 1, ..., T$.

Actually, these auxiliary variables $z_t$ turn out to be superfluous here, as they can simply be replaced with the upper bound $S_t$. This yields

$$(\text{BND}^{(1)}_{tw}\text{-LIN}) \qquad \begin{cases} \max & c^\top x \\ \text{s.t.} & x_{t-1} - x_t \leq S_t \quad t = 1, ..., T \\ & x_t - x_{t-1} \leq S_t \quad t = 1, ..., T \\ & x \in \{0,1\}^{T+1} . \end{cases}$$

Observe that the constraint matrix of ($\text{BND}^{(1)}_{tw}$-LIN) is a submatrix of the constraint matrix of ($\text{P}^{(1)}$-LIN) and thus, the former matrix inherits the total unimodularity from the latter. As a result, the binarity constraint may be relaxed and the problem can be solved by linear programming. This once again shows the tractability of ($\text{BND}^{(1)}_{tw}$). Clearly, one may include a constraint $x_t = b$ with $b \in \{0, 1\}$ for any stage $t$, since this does not affect the total unimodularity of the constraint matrix. Thus, the (continuous relaxation of) problem ($\text{BND}^{(1)}_{tw}$-LIN) is a compact model that lives in the original space of the $x$ variables without any auxiliary variables. This formulation of ($\text{P}^{(1)}$) should seem familiar. Indeed,

**Remark 3.2.1.** The feasible set of the continuous relaxation of ($\text{BND}^{(1)}_{tw}$-LIN) can be obtained as the intersection of the feasible set of problem ($\text{P}^{(1)}$-S) from Lemma 3.1.3 with the hyperplane that fixes the variable $S$ to the given bound on the time-wise variation. Moreover, the previous LP model is a compact perfect formulation of the convex hull of feasible solutions for ($\text{BND}^{(1)}_{tw}$).

This concludes the study of the bound on time-wise variation. For the remainder of this chapter, the bound on total variation and switch-wise variation is investigated.

## 3.3 Bounded Total and Switch-Wise Variation

In this section, the problem (BND) with switch-wise and total variation is studied. These two types of variation are indistinguishable for $n = 1$, thus this section uses

the term *variation* to refer to both of them at once. The problem (BND) then reads

$$(\mathrm{BND}^{(1)}_{\mathrm{var}}) \qquad \begin{cases} \max & c^\top x \\ \text{s.t.} & \sum_{t=1}^{T} |x_{t-1} - x_t| \leq S \\ & x \in \{0,1\}^{T+1} \,. \end{cases}$$

This section is motivated by the same questions that motivated the preceding ones for the penalty problem ($\mathrm{P}^{(1)}$) and the problem ($\mathrm{BND}^{(1)}_{tw}$). The aim is, of course, to settle the complexity of ($\mathrm{BND}^{(1)}_{\mathrm{var}}$), but beyond that, the aim will be to find a perfect formulation, i.e., a description of the convex hull of feasible solutions by linear inequalities. The latter will make up the majority of this section because the polyhedral study of ($\mathrm{BND}^{(1)}_{\mathrm{var}}$), compared to the previous problems, will turn out to be much more involved. In this entire section, it is assumed w.l.o.g. that $S < T$ as otherwise, the variation constraint is redundant and the problem ($\mathrm{BND}^{(1)}_{\mathrm{var}}$) becomes trivial.

Although there is a relation between the penalty problem ($\mathrm{P}^{(1)}$), or equivalently ($\mathrm{P}^{(1)}$-S), and ($\mathrm{BND}^{(1)}_{\mathrm{var}}$), the knowledge obtained about ($\mathrm{P}^{(1)}$) in Section 3.1 is not helpful here. Indeed, while ($\mathrm{P}^{(1)}$-S) can clearly be reduced to ($\mathrm{BND}^{(1)}_{\mathrm{var}}$) by enumerating all $S \in \{0, ..., T\}$, a reduction in the other direction is not possible. In fact, it might happen that the optimal solution of ($\mathrm{BND}^{(1)}_{\mathrm{var}}$) cannot be obtained with ($\mathrm{P}^{(1)}$-S), no matter how the penalty parameter $\lambda$ is chosen:

**Example 3.3.1.** Consider ($\mathrm{BND}^{(1)}_{\mathrm{var}}$) with $T = 2$, $S = 1$, and $c = (-100, 3, -2)^\top$. Then, the obvious candidates for optimal solutions of ($\mathrm{BND}^{(1)}_{\mathrm{var}}$) are given by

$$x^{(1)} = (0,0,0)^\top,\ x^{(2)} = (0,0,1)^\top,\ \text{ and } x^{(3)} = (0,1,1)^\top \,,$$

where $x^{(3)}$ is the unique optimizer with optimal value 1. This solution, however, cannot be obtained from ($\mathrm{P}^{(1)}$-S) for any choice of the penalty parameter $\lambda$. Indeed, the candidates for optimum solutions of ($\mathrm{P}^{(1)}$-S) are the following:

| solution | objective value |
|---|---|
| $x^{(1)} = (0,0,0)^\top, S^{(1)} = 0$ | $0$ |
| $x^{(2)} = (0,0,1)^\top, S^{(2)} = 1$ | $-2 - \lambda$ |
| $x^{(3)} = (0,1,1)^\top, S^{(3)} = 1$ | $1 - \lambda$ |
| $x^{(4)} = (0,1,0)^\top, S^{(4)} = 2$ | $3 - 2\lambda$ |

For $\lambda \leq 3/2$, the solution $x^{(4)}$, $S^{(4)}$ is optimal for ($\mathrm{P}^{(1)}$-S), otherwise $x^{(1)}$, $S^{(1)}$ is optimal; but the optimal solution $x^{(3)}$ of ($\mathrm{BND}^{(1)}_{\mathrm{var}}$) is never optimal for ($\mathrm{P}^{(1)}$-S).

Moreover, the next example highlights a contrast to Remark 3.2.1: while the intersection of the polyhedron of the continuous penalty problem ($\mathrm{P}^{(1)}$-S) with the

hyperplane fixing $S$ to the given bound is integral for time-wise variation, the same does not hold here.

**Example 3.3.2.** Consider ($\mathrm{P}^{(1)}$-S) for $T = 3$, $c = (-100, 1, -3, 1)^\top$, and any $\lambda > 0$. According to Corollary 3.1.2, this problem is equivalent to

$$\begin{aligned} \max \quad & -100 \cdot x_0 + x_1 - 3 \cdot x_2 + x_3 - \lambda \cdot S \\ \text{s.t.} \quad & u_1 + u_2 + u_3 + d_1 + d_2 + d_3 \leq S \\ & x_t - x_{t-1} \leq u_t && t = 1, ..., 3 \\ & x_{t-1} - x_t \leq d_t && t = 1, ..., 3 \\ & x \in [0,1]^4, u, d \in \mathbb{R}^3_{\geq 0} \, . \end{aligned}$$

Fixing $S := 2$ in the above problem yields $(0, 1/2, 0, 1)^\top$ as unique optimizer of the resulting linear program. In particular, the intersection of the continuous relaxation of ($\mathrm{P}^{(1)}$-S) with the hyperplane given by $S = 2$ is not an integral polytope and therefore, does not agree with the convex hull of feasible solutions for ($\mathrm{BND}^{(1)}_{\mathrm{var}}$).

Nonetheless, the problem ($\mathrm{BND}^{(1)}_{\mathrm{var}}$) is tractable.

**Lemma 3.3.1.** The problem ($\mathrm{BND}^{(1)}_{\mathrm{var}}$) can be solved efficiently with a dynamic programming scheme. A straightforward implementation of the scheme requires a running time of $\mathcal{O}(S \cdot T)$. In particular, if $S$ is considered part of the input, this leads to a running time of $\mathcal{O}(T^2)$, since $S < T$. If instead $S$ is assumed to be a fixed constant, the running time of the dynamic programming scheme reduces to $\mathcal{O}(T)$.

*Proof.* The dynamic programming scheme is devised in the following: Let $c^*(t, s, b)$ denote the optimal value of ($\mathrm{BND}^{(1)}_{\mathrm{var}}$) restricted to time stages $\{0, ..., t\}$, with a bound $s \in \{0, ..., S\}$ on the variation, and such that $x_t = b$. Now, define $c^*(0, s, b) := b \cdot c_0$ for $b \in \{0, 1\}$ and all $s \in \{0, ..., S\}$, then the subsequent recursion formula holds:

$$c^*(t, s, b) = b \cdot c_t + \max \begin{cases} c^*(t-1, s, b) \\ c^*(t-1, s-1, 1-b) \quad \text{if } s \geq 1 \end{cases}$$

Indeed, an optimal solution for $\{0, ..., t\}$ with $x_t = b$ and a variation of $s$ is obtained by either extending an optimal solution for $\{0, ..., t-1\}$ with $x_{t-1} = b$ and the same variation, or by extending an optimal solution for $\{0, ..., t-1\}$ with $x_{t-1} = 1-b$ with a variation one less, since in the latter case an additional change arises between $x_{t-1}$ and $x_t$. The term $b \cdot c_t$ represents the additional weight incurred by setting $x_t$ to $b$. Thus, the optimal value of ($\mathrm{BND}^{(1)}_{\mathrm{var}}$) is given by $\max\{c^*(T, S, 0), c^*(T, S, 1)\}$. It can

be computed in $\mathcal{O}(S \cdot T)$ time, and an optimal solution can be constructed from the recursion. In order to adapt the scheme for the respective fixated problem versions, simply define the initial values either by $c^*(0, s, 0) := 0$ and $c^*(0, s, 1) := -\infty$, or by $c^*(0, s, 0) := -\infty$ and $c^*(0, s, 1) := c_0$ for all $s$. □

Note that because $(\mathrm{P}^{(1)})$ can be polynomially reduced to the solution of at most $T$ instances of $(\mathrm{BND}^{(1)}_{\mathrm{var}})$, the tractability of $(\mathrm{BND}^{(1)}_{\mathrm{var}})$ established in Lemma 3.3.1 once again shows the tractability of $(\mathrm{P}^{(1)})$.

The remainder of this section is now dedicated to the study of $(\mathrm{BND}^{(1)}_{\mathrm{var}})$ from a polyhedral perspective. Here, the focus will lie on the following special case in which the problem's symmetry is broken by fixing the state of the switch at the first stage to zero.

$$(\mathrm{BND}^{(1)}_{\mathrm{var},0}) \qquad \begin{cases} \max & c^\top x \\ \text{s.t.} & \sum_{t=1}^{T} |x_{t-1} - x_t| \leq S \\ & x_0 = 0,\ x \in \{0,1\}^{T+1} \end{cases}$$

Analogously, let $(\mathrm{BND}^{(1)}_{\mathrm{var},1})$ refer to the respective variant in which the state of the switch is instead fixed to one at the first stage.

**Definition 3.3.1.** Let $\Omega$ denote the feasible set of $(\mathrm{BND}^{(1)}_{\mathrm{var}})$. Accordingly, denote the feasible set of $(\mathrm{BND}^{(1)}_{\mathrm{var},0})$ by $\Omega_0$ and conversely, let $\Omega_1$ refer to the feasible set of $(\mathrm{BND}^{(1)}_{\mathrm{var},1})$.

**Remark 3.3.1.** The problems $(\mathrm{BND}^{(1)}_{\mathrm{var},0})$ and $(\mathrm{BND}^{(1)}_{\mathrm{var},1})$ are point-symmetrical with respect to the point $(\frac{1}{2}, ..., \frac{1}{2})$. Indeed,

$$x \in \Omega_0 \quad \Leftrightarrow \quad y := \mathbb{1}_{T+1} - x \in \Omega_1 \ .$$

Therefore, a valid inequality $a^\top x \leq b$ for the polytope $\mathrm{conv}(\Omega_0)$ implies the validity of the inequality $\tilde{a}^\top y \leq \tilde{b}$ for the polytope $\mathrm{conv}(\Omega_1)$, where $\tilde{a} := -a$ and $\tilde{b} := b - \mathbb{1}^\top_{T+1} a$. In particular, facets of $\mathrm{conv}(\Omega_0)$ map bijectively to facets of $\mathrm{conv}(\Omega_1)$, and vice versa.

The remaining section is divided into three subsections, each of which is dedicated to a particular polyhedral aspect of the convex hulls $\mathrm{conv}(\Omega_0)$, $\mathrm{conv}(\Omega_1)$, and $\mathrm{conv}(\Omega)$, where the focus will continue to lie on the first polytope. The next subsection will present three compact extended models, one for each problem $(\mathrm{BND}^{(1)}_{\mathrm{var},0})$, $(\mathrm{BND}^{(1)}_{\mathrm{var},1})$ and $(\mathrm{BND}^{(1)}_{\mathrm{var}})$. The main result will be that the continuous relaxation of

each model is integral, so that it yields a polyhedral description in the respective extended solution space. The subsequent Section 3.3.2 will then derive a polyhedral description in the original space of only the $x$-variables. The mode in which the polyhedral description will be proven is particularly interesting in two regards: It will use primal-dual argumentation, unlike any preceding proof, but, more importantly, it will also yield an alternative efficient optimization algorithm, which will significantly outperform the dynamic programming scheme. The third and final subsection will conclude the polyhedral study by showing that the linear inequality description obtained in the preceding subsection is minimal, i.e., it consists of only facet-inducing inequalities. In addition, it will present an exact linear-time separation algorithm.

### 3.3.1 Extended Formulations and Complete Description

This section deals with extended linear programming models for the three problem variants ($\mathrm{BND}^{(1)}_{\mathrm{var},0}$), ($\mathrm{BND}^{(1)}_{\mathrm{var},1}$) and ($\mathrm{BND}^{(1)}_{\mathrm{var}}$). While the correctness of the proposed models for the first two problems follows easily from total unimodularity arguments, proving the correctness of the extended formulation for the more general problem ($\mathrm{BND}^{(1)}_{\mathrm{var}}$) turns out to be more sophisticated.

Inspired by the results for the penalty problem and the bound on the time-wise variation, it is tempting to try and directly adapt the model ($\mathrm{P}^{(1)}$-S) so as to obtain

$$\begin{aligned}
\max \quad & c^\top x \\
\text{s.t.} \quad & \sum_{t=1}^{T}(u_t + d_t) \leq S \\
& x_t - x_{t-1} \leq u_t && t = 1, ..., T \\
& x_{t-1} - x_t \leq d_t && t = 1, ..., T \\
& x \in \{0,1\}^{T+1},\ u, d \in \mathbb{R}^T_{\geq 0}\ ,
\end{aligned}$$

where $S$ is a hard bound now. However, it follows from Example 3.3.2 that the feasible set of the LP-relaxation of this model is not integral, even for $S = 2$. As it turns out, the key is paying attention to the parity of $S$. More precisely, observe that if the switch is inactive at the first stage, i.e., $x_0 = 0$, and $S$ is odd, then a bound of $S$ on the total variation is actually equivalent to a bound of $\lfloor S/2 \rfloor$ on the total number of times the switch is *deactivated*. Indeed, suppose that $x_0 = 0$, but $x$ contains (at least) $\lfloor S/2 \rfloor + 1 = (S-1)/2 + 1$ deactivations. Each deactivation implies the existence of a respective activation that precedes it, therefore the total variation of $x$ is (at least) $2 \cdot ((S-1)/2 + 1) = S + 1$ and thus, $x \notin \Omega_0$. The reasoning for the other direction is similar: if $x$ satisfies $x_0 = 0$, but it has a total variation of (at least) $S+1$, then the $S+1$-st change in $x$ must be a deactivation. Again, each deactivation is preceded by a corresponding activation, therefore the total number of deactivations is (at least) $S+1/2 = \lceil S/2 \rceil > \lfloor S/2 \rfloor$.

Conversely, an even bound $S$ on the total variation is equivalent to bounding the total number of times the switch is *activated* by $S/2$. Indeed, assume that $x_0 = 0$ and that $x$ contains (at least) $S/2 + 1$ activations. Each activation (except the very first) requires a respective deactivation that precedes it, therefore the total variation of $x$ is (at least) $1 + 2 \cdot S/2 = S + 1$, and thus $x \notin \Omega_0$. Similarly, if $x_0 = 0$ and $x$ has a total variation of (at least) $S + 1$, then the $S + 1$-st change in $x$ must correspond to an activation. Since each activation (other that the very first) implies a preceding deactivation, the total number of activations is (at least) $1 + S/2 > S/2$.
This observation leads to the model (EXT-BND$_0$) below which is an extended formulation for (BND$^{(1)}_{\text{var},0}$). Note that this model works not only independent of the parity of $S$, but it also only needs the deactivation variables $d$ and no activation variables.

$$
\text{(EXT-BND}_0\text{)} \qquad \begin{cases} \max & c^\top x & & \\ \text{s.t.} & x_T + \sum_{t=1}^{T} d_t \leq \lfloor \frac{S-1}{2} \rfloor + 1 & & (1) \\ & \sum_{t=1}^{T} d_t \leq \lfloor \frac{S}{2} \rfloor & & (2) \\ & x_{t-1} - x_t \leq d_t & t = 1, ..., T & \\ & x_0 = 0,\ x \in \{0,1\}^{T+1},\ d \in \mathbb{R}^T_{\geq 0} & & \end{cases}
$$

**Remark 3.3.2.** Depending on the parity of $S$, either constraint (1) or constraint (2) becomes redundant. Indeed, if $S$ is even, both right hand sides of (1) and (2) are equal to $S/2$ so that constraint (2) is redundant, as it is implied by (1) and $x_T \geq 0$. Conversely, if $S$ is odd, it holds $\lfloor S/2 \rfloor = (S-1)/2 < \lfloor (S-1)/2 \rfloor + 1 = (S-1)/2 + 1$ so that constraint (1) is redundant, as it is implied by (2) and $x_T \leq 1$.
Moreover, due to the fixation $x_0 = 0$, each deactivation implies the existence of a respective activation that precedes it, therefore the total number of activations is given by the total number of all deactivations plus $x_T$, i.e., the state of the switch at the last stage, as the latter indicates the existence of final activation.

**Lemma 3.3.2.** The model (EXT-BND$_0$) is an extended formulation for (BND$^{(1)}_{\text{var},0}$).

*Proof.* Let $x \in \Omega_0$ be given and define $u'$ and $d'$ as the corresponding auxiliary activation and deactivation vectors with respect to $x$, i.e., $u'_t := \max\{x_t - x_{t-1}, 0\}$ and $d'_t := \max\{x_{t-1} - x_t, 0\}$ for $t = 1, ..., T$. The fact that the total variation in $x$ is at most $S$, and that $x_0 = 0$, is equivalent to

$$
\sum_{t=1}^{T} u'_t \leq \lfloor \tfrac{S-1}{2} \rfloor + 1 \quad \text{if } S \text{ is even} \qquad \text{or} \qquad \sum_{t=1}^{T} d'_t \leq \lfloor \tfrac{S}{2} \rfloor \quad \text{if } S \text{ is odd}\,.
$$

Furthermore, $\sum_{t=1}^{T} u'_t = x_T + \sum_{t=1}^{T} d'_t$ due to $x_0 = 0$. Thus, the existence of some $d \in \mathbb{R}^T$ so that $(x, d)$ is feasible for (EXT-BND$_0$) follows.
For the other direction, let $(x, d)$ be an arbitrary feasible element of (EXT-BND$_0$) and w.l.o.g. , assume $d$ to be minimal regarding the constraints $x_{t-1} - x_t \leq d_t$ for all $t$. Depending on the parity of $S$, one may ignore the respective redundant constraint (1) or (2). The remaining constraint then implies that the total variation of $x$ does not exceed $S$, thus $x \in \Omega_0$. □

**Lemma 3.3.3.** The continuous relaxation of (EXT-BND$_0$) is integer.

*Proof.* Recall that inequality (2) is redundant if $S$ is even, while inequality (1) is redundant if $S$ is odd. One may remove the respective redundant inequality and thus, derive a constraint matrix, the total unimodularity of which is easy to recognize by Ghouila-Houri's criterion. Since all right hand sides are integer, the integrality of the continuous relaxation follows. The integrality of the latter is clearly not impacted by the re-inclusion of the redundant inequality. This shows the claim. □

Analogous reasoning for the symmetric problem (BND$^{(1)}_{\text{var},1}$) leads to the following extended formulation:

$$
\text{(EXT-BND}_1\text{)} \quad
\begin{cases}
\max & c^\top x & \\
\text{s.t.} & (1 - x_T) + \displaystyle\sum_{t=1}^{T} u_t \leq \lfloor \tfrac{S-1}{2} \rfloor + 1 & (3) \\
& \displaystyle\sum_{t=1}^{T} u_t \leq \lfloor \tfrac{S}{2} \rfloor & (4) \\
& -x_{t-1} + x_t \leq u_t \qquad t = 1, ..., T & \\
& x_0 = 1,\ x \in \{0,1\}^{T+1},\ u \in \mathbb{R}^T_{\geq 0} &
\end{cases}
$$

**Lemma 3.3.4.** The model (EXT-BND$_1$) is an extended formulation for (BND$^{(1)}_{\text{var},1}$) . Moreover, its continuous relaxation is integral.

*Proof.* The reasoning is analogous to the one in the proofs of Lemmas 3.3.2 and 3.3.3: If $S$ odd and $x_0 = 1$, then a bound of $S$ on the total variation is equivalent to a bound of $\lfloor S/2 \rfloor$ on the the number activations. Moreover, the constraint (3) is redundant, as it is implied by (4) and $x_T \geq 0$.
Conversely, an even bound $S$ on the total variation is equivalent to the number deactivations being bounded by $S/2$. Here, the total number of deactivations is given by the sum of $(1 - x_T)$ and the total number of activations due to the fact that every activation requires the existence of a preceding deactivation, and there is a

final deactivation if and only if $x_T = 0$. For $S$ even, the constraint (4) is redundant, as it is implied by (3) and $x_T \leq 1$. The integrality of the corresponding continuous relaxation follows analogously. □

It remains to look for an extended formulation for the more general problem ($\text{BND}^{(1)}_{\text{var}}$). One might hope that simply dropping the fixation constraints from the extended formulations (EXT-$\text{BND}_0$), (EXT-$\text{BND}_1$) and combining the two resulting formulations yields an extended model for the general case. However, this is only true to some extent, as the crucial part is that the variable $x_0$ needs to be included in a particular way. More precisely, an extended formulation for ($\text{BND}^{(1)}_{\text{var}}$) is given by

$$
\text{(EXT-BND)} \quad \left\{ \begin{array}{lll}
\max & c^\top x & \\
\text{s.t.} & -x_0 + x_T + \sum_{t=1}^{T} d_t \leq \lfloor \frac{S-1}{2} \rfloor + 1 & (1a) \\
& -x_0 + \sum_{t=1}^{T} d_t \leq \lfloor \frac{S}{2} \rfloor & (2a) \\
& -(1-x_0) + (1-x_T) + \sum_{t=1}^{T} u_t \leq \lfloor \frac{S-1}{2} \rfloor + 1 & (3a) \\
& -(1-x_0) + \sum_{t=1}^{T} u_t \leq \lfloor \frac{S}{2} \rfloor & (4a) \\
& x_t - x_{t-1} \leq u_t \qquad t = 1, ..., T & (5) \\
& x_{t-1} - x_t \leq d_t \qquad t = 1, ..., T & (6) \\
& x \in \{0,1\}^{T+1},\ u, d \in \mathbb{R}^T_{\geq 0}\ . &
\end{array} \right.
$$

Note that for $x_0 = 0$, the constraints (1a) and (2a) actually correspond exactly to the constraints (1) and (2) from (EXT-$\text{BND}_0$) while for $x_0 = 1$, the constraints (3a) and (4a) correspond to the constraints (3) and (4) from (EXT-$\text{BND}_1$). The correctness of (EXT-$\text{BND}_0$) and (EXT-$\text{BND}_1$) thus immediately imply the correctness of (EXT-BND). Similarly, the redundancy of constraints (1a) and (3a) follows if $S$ is odd, whereas constraints (2a) and (4a) are redundant if $S$ is even. In particular, this redundancy does not depend on the integrality of $x$ so that it also holds for the continuous relaxation of (EXT-BND). But, unfortunately, the total unimodularity of the models (EXT-$\text{BND}_0$) and (EXT-$\text{BND}_1$) does not transfer to (EXT-BND). Nevertheless, the continuous relaxation of (EXT-BND) is still integral. The corresponding Theorem 3.3.1 and its proof are delayed until after the next two auxiliary lemmas which help to characterize vertices of the continuous relaxation of (EXT-BND). Both these lemmas and the subsequent Theorem 3.3.1 are proven using the same strategy, which is to give a proof by contradiction relying on the fact that a vertex of a polyhedron *cannot* be expressed as a (non-trivial) convex combination of two other feasible

elements. More precisely, from now on let $(x^*, d^*, u^*)$ denote an arbitrary vertex of the polyhedron which corresponds to the continuous relaxation of (EXT-BND). Then, $(x^*, d^*, u^*)$ must possess the characteristics claimed in the following Lemmas 3.3.5, 3.3.6 and Theorem 3.3.1 as otherwise, it could be obtained as the (non-trivial) convex combination of two particular feasible elements $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ and thus would not be a vertex. The particular elements $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ are obtained from $(x^*, d^*, u^*)$ by slightly *altering* certain components. For a precise description of this alteration, define $\varepsilon$ for the vertex $(x^*, d^*, u^*)$ by

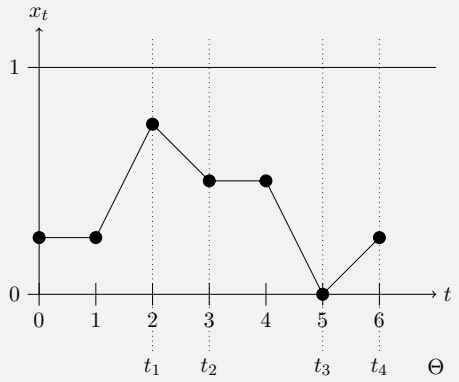
$$0 < \varepsilon < \min \left\{ \min_{t=1,...,T} |x_t^* - x_{t-1}^*|, \min_{x_t^*<1} (1 - x_t^*), \min_{x_t^*>0} x_t^* \right\} .$$

In order to refer to the necessary indices in the subsequent proofs, let $k$ describe the total number of deactivations and activations in $x^*$ and let the set $\Theta \subseteq \{1, ..., T\}$ collect all of the respective indices, i.e., $\Theta := \{t \in \{1, ..., T\} : |x_{t-1}^* - x_t^*| > 0\}$. The elements of $\Theta$ are denoted by $t_1, ..., t_k$, where $t_j < t_{j+1}$ for $j = 1, ..., k-1$. Roughly speaking, the set $\Theta$ captures all indices that belong to either an activation or a deactivation regarding only the values in the vector $x^*$. For a better grasp of this set, consider the following brief example.

**Example 3.3.3.** Consider the continuous relaxation of (EXT-BND) for $T = 6$ and $S = 2$. Then, a feasible solution is given by:

$$x = (1/4, 1/4, 3/4, 1/2, 1/2, 0, 1/4),\ u = (0, 5/8, 0, 0, 0, 1/4),\ d = (0, 1/8, 1/4, 0, 1/2, 0) .$$

The structure of the vector $x$ is visualized in the graph below in the form of a discrete function over the time horizon. However, to ease the understanding in the following proofs, each two adjacent discrete points of the graph are connected linearly. These linear segments between two consecutive discrete points symbolize the *exact* variation: An incline indicates an activation in $x$, i.e., $-x_{t_j-1} + x_{t_j} > 0$, and gives a lower bound on the variable $u_{t_j}$, while a decline indicates a decactivation in $x$, i.e., $x_{t_j-1} - x_{t_j} > 0$, and gives a lower bound on the variable $d_{t_j}$. The corresponding set $\Theta$ is marked on the $t$-axis.



Note that while the inclines and declines illustrated in the graph give the exact variation between two consecutive values in $x$, they do not necessarily show the value of $u$ or $d$, but only a lower bound. For example, the incline between time point 1 and 2 is given by $x_2 - x_1 = 1/2$, which is exceeded by the value $u_2 = 5/8$. Similarly, the value $d_2 = 1/8$ is positive, although there occurs no deactivation at this stage.

Using that notation, the first auxiliary lemma can be phrased. Roughly speaking, it states when considering only the $x^*$-vector of $(x^*, d^*, u^*)$, this cannot have two consecutive changes in the same direction.

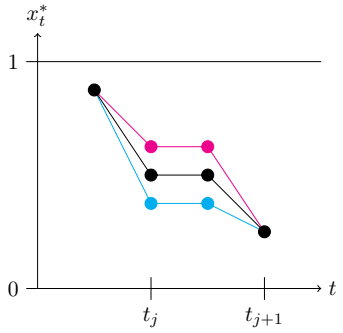
**Lemma 3.3.5.** Let $(x^*, d^*, u^*)$ be a vertex of the polyhedron corresponding to the continuous relaxation of (EXT-BND). For $t_j, t_{j+1} \in \Theta$, it either holds

$$x^*_{t_j-1} > x^*_{t_j} < x^*_{t_{j+1}} \qquad \text{or} \qquad x^*_{t_j-1} < x^*_{t_j} > x^*_{t_{j+1}} \,.$$

*Proof.* Let $(x^*, d^*, u^*)$ be a vertex of the continuous relaxation of (EXT-BND) and assume that there exist $t_j, t_{j+1} \in \Theta$ with $x^*_{t_j-1} > x^*_{t_j} > x^*_{t_{j+1}}$. This way, both $d^*_{t_j}$ and $d^*_{t_{j+1}}$ are forced to take a positive value by constraints (6). Now, the solution $(x^*, d^*, u^*)$ is slightly altered, so as to obtain two new feasible solutions $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ via *shifting* the values $x^*_t$ by the constant value $\varepsilon$ defined above for all $t$ with $t_j \leq t < t_{j+1}$. This shift is also carried over to $d^*$ and $u^*$ to make sure that (5) and (6) are still satisfied. More formally, define

$$x^+_t := \begin{cases} x^*_t + \varepsilon & t_j \leq t < t_{j+1}, \\ x^*_t & \text{otherwise}, \end{cases} \qquad d^+_t := \begin{cases} d^*_t - \varepsilon & t = t_j, \\ d^*_t + \varepsilon & t = t_{j+1}, \\ d^*_t & \text{otherwise}, \end{cases} \qquad u^+ := u^* \,.$$

Now and in the following, let $(x^-, d^-, u^-)$ be defined analogously, only with inverted signs for $\varepsilon$. The figure below illustrates the idea behind the above definition of $x^+$ (magenta) and $x^-$ (cyan).



It is easy to see that $x^+$ and $x^-$ satisfy the trivial lower and upper bounds by definition of $\varepsilon$. The constraints (5) and (6) are satisfied by definition of $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$. It remains to show that constraints (1a)-(4a) are satisfied. Observe that by definition, it holds $x_0^+ = x_0^- = x_0^*$ as well as $x_T^+ = x_T^- = x_T^*$. It thus suffices to show that the total sum of the respective variables $d^+$, $d^-$ and $u^+$, $u^-$ does not change., i.e., that that $\sum_{t=1}^T d_t^+ = \sum_{t=1}^T d_t^- = \sum_{t=1}^T d_t^*$ and $\sum_{t=1}^T u_t^+ = \sum_{t=1}^T u_t^- = \sum_{t=1}^T u_t^*$. This, however, is obvious by their definition. Consequently, the vertex $(x^*, d^*, u^*)$ can be expressed as the convex combination of $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$:

$$(x^*, d^*, u^*) = \tfrac{1}{2}(x^+, d^+, u^+) + \tfrac{1}{2}(x^-, d^-, u^-) \; .$$

This yields a contradiction to $(x^*, d^*, u^*)$ being a vertex. The argument for the other case $x^*_{t_j-1} < x^*_{t_j} < x^*_{t_{j+1}}$ is analogous. □

Now, since the vector $x^*$ cannot have two consecutive changes in the same direction, a consequence of the previous lemma is that each index in $\Theta$ must belong to a local maximum or a local minimum in $x^*$. The next lemma studies these local extrema more closely. Its essence is that the vector $(x^*_{t_1}, x^*_{t_2}, ..., x^*_{t_{k-1}})$ can contain at most one fractional local extremum $x^*_{t_\ell}$.

**Lemma 3.3.6.** Let $(x^*, d^*, u^*)$ be a vertex of the continuous relaxation of (EXT-BND). Then, there exists at most one index $t_\ell \in \Theta \setminus \{t_k\}$ with $x^*_{t_\ell} \in (0, 1)$.

*Proof.* Let $(x^*, d^*, u^*)$ be a vertex of the continuous relaxation of (EXT-BND) and assume that the statement does not hold, i.e., that there exist two indices $t_\ell \neq t_j$, $t_\ell, t_j \in \Theta \setminus \{t_k\}$ with $x^*_{t_\ell}, x^*_{t_j} \in (0, 1)$, where w.l.o.g. $t_\ell < t_j$.
First, consider the case $t_{\ell+1} < t_j$ where both $x^*_{t_\ell}$ and $x^*_{t_j}$ are extrema of the same type. W.l.o.g., assume that both are local maxima (otherwise switch the roles of $t_{j+1}$ and $t_\ell$ as well as $t_{\ell+1}$ and $t_j$ in the following definition of $d^+$,$u^+$) and define

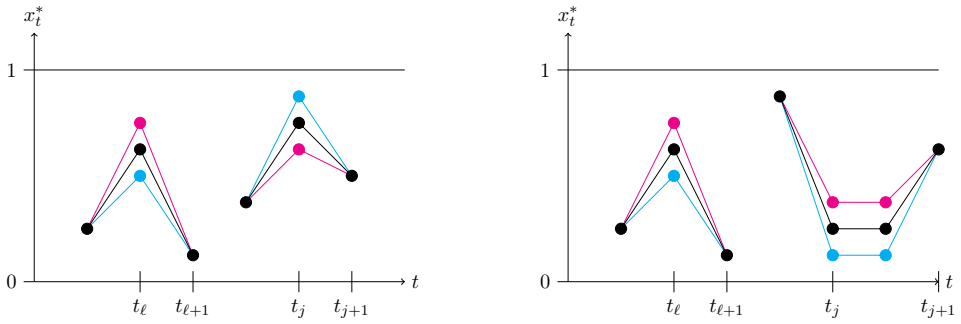
$$x_t^+ := \begin{cases} x_t^* + \varepsilon & t_\ell \le t < t_{\ell+1}, \\ x_t^* - \varepsilon & t_j \le t < t_{j+1}, \\ x_t^* & \text{otherwise}, \end{cases} \; d_t^+ := \begin{cases} d_t^* + \varepsilon & t = t_{\ell+1}, \\ d_t^* - \varepsilon & t = t_{j+1}, \\ d_t^* & \text{otherwise}, \end{cases} \; u_t^+ := \begin{cases} u_t^* + \varepsilon & t = t_\ell, \\ u_t^* - \varepsilon & t = t_j, \\ u_t^* & \text{otherwise}, \end{cases}$$

and define $(x^-, d^-, u^-)$ analogously, only with inverted signs for $\varepsilon$. The idea behind the definition of $x^+$ and $x^-$ is sketched below in Figure 3.1a.

Otherwise, if $t_{\ell+1} < t_j$ and $x^*_{t_\ell}$ and $x^*_{t_j}$ are opposite extrema, assume w.l.o.g. that $x^*_\ell$ is a local maximum (otherwise switch the roles of $\ell$ and $j$ in the definition of $d^+$ and $u^+$) and define

$$x_t^+ := \begin{cases} x_t^* + \varepsilon & t_\ell \le t < t_{\ell+1}, \\ x_t^* + \varepsilon & t_j \le t < t_{j+1}, \\ x_t^* & \text{otherwise}, \end{cases} \; d_t^+ := \begin{cases} d_t^* + \varepsilon & t = t_{\ell+1}, \\ d_t^* - \varepsilon & t = t_j, \\ d_t^* & \text{otherwise}, \end{cases} \; u_t^+ := \begin{cases} u_t^* + \varepsilon & t = t_\ell, \\ u_t^* - \varepsilon & t = t_{j+1}, \\ u_t^* & \text{otherwise}, \end{cases}$$

and again, let $(x^-, d^-, u^-)$ be defined analogously, only with inverted signs for $\varepsilon$. The subsequent figure illustrates the idea behind the definition of $x^+$ and $x^-$ depending on the previous case distinction.




(a) Two same-type local extrema

(b) Two different-type local extrema

Figure 3.1: Depiction of $x^+$ (magenta) and $x^-$ (cyan) for the case $t_{\ell+1} < t_j$

Finally, consider the case $t_{\ell+1} = t_j$. Here, assume that $x^*_{t_\ell}$ is a local maximum and $x^*_{t_{\ell+1}} = x^*_{t_j}$ is a local minimum (the reverse case is analogous) and define

$$x_t^+ := \begin{cases} x_t^* + \varepsilon & t_\ell \le t < t_{j+1}, \\ x_t^* & \text{otherwise}, \end{cases} \quad d^+ := d^*, \; u_t^+ := \begin{cases} u_t^* + \varepsilon & t = t_\ell, \\ u_t^* - \varepsilon & t = t_{t+1}, \\ u_t^* & \text{otherwise}, \end{cases}$$

and $(x^-, d^-, u^-)$ with inverted signs for $\varepsilon$. The figure below sketches the new vectors $x^+$ and $x^-$ in this case.

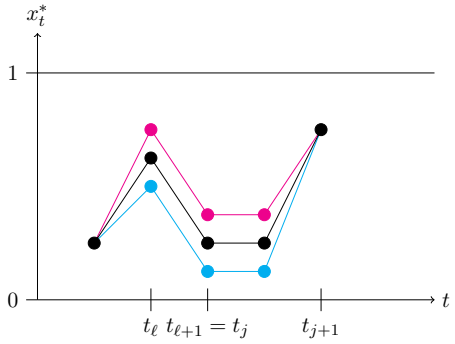


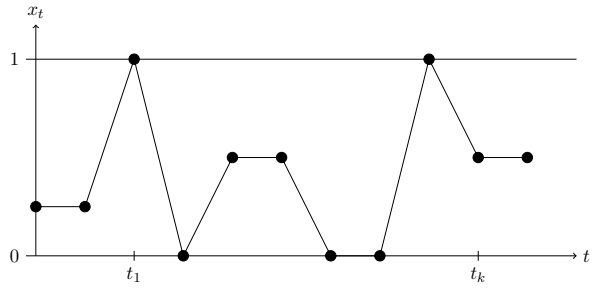

Figure 3.2: Depiction of $x^+$ and $x^-$ for the case $t_{\ell+1} = t_j$

With the same reasoning as in the proof of Lemma 3.3.5, the feasibility of the solutions $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ follows for each of the previous cases. By construction,

$$(x^*, d^*, u^*) = \tfrac{1}{2}(x^+, d^+, u^+) + \tfrac{1}{2}(x^-, d^-, u^-)$$

which leads to a contradiction, since $(x^*, d^*, u^*)$ is a vertex. □

With the two previous auxiliary Lemmas 3.3.5 and 3.3.6, it is established that given some vertex $(x^*, d^*, u^*)$, the vector $(x^*_{t_1}, x^*_{t_1+1}, ..., x^*_{t_k-1})$ must be composed of all binary local extrema, except for at most one constant sequence $x^*_{t_\ell}, x^*_{t_\ell+1}, ..., x^*_{t_{\ell+1}-1}$ corresponding to fractional local extrema. So far, no claim has yet been made about the value of the (constant) sequence $x^*_0, ..., x^*_{t_1-1}$ or the (constant) sequence $x^*_{t_k}, ..., x^*_T$. As of now, the $x^*$-vector of a vertex $(x^*, d^*, u^*)$ could therefore still have a structure as depicted below in an exemplary way:




It turns out, however, that no vertex $(x^*, d^*, u^*)$ can have an $x^*$-vector with such a structure. More precisely, any vertex $(x^*, d^*, u^*)$ of the continuous relaxation of (EXT-BND) must be integer.

**Theorem 3.3.1.** The continuous relaxation of (EXT-BND) is integral.

*Proof.* The following proof is restricted to the case in which $S$ is odd, since the reasoning for an even $S$ is analogous.

So, let $S$ be odd and let $(x^*, d^*, u^*)$ be a vertex of the continuous relaxation of (EXT-BND). Due to the parity of $S$, the constraints (1a) and (3a) are redundant. Hence, the only non-trivial constraints that need to be considered are

$$-x_0 + \sum_{t=1}^{T} d_t \leq \lfloor \tfrac{S}{2} \rfloor \quad (2a) \qquad \text{and} \qquad x_0 + \sum_{t=1}^{T} u_t \leq \lfloor \tfrac{S}{2} \rfloor + 1 \quad (4a)\,.$$

The subsequent proof is split into two parts. The first part is dedicated to proving the integrality of the vector $x^*$. This is achieved by a large case distinction based on the structure of $x^*$ which exploits that at least one of the two constraints (2a),(4a) must be tight for $(x^*, d^*, u^*)$. Second, it is shown that the integrality of $x^*$ implies the integrality of $d^*$ and $u^*$.

To commence the first part, suppose that $x^*$ is not integer. According to Lemma 3.3.6, there can exist at most one index $t_\ell \in \Theta \setminus \{t_k\}$ corresponding to a *fractional extremum* $x^*_{t_\ell} \in (0, 1)$. Thus, three possible structures for $x^*$ remain, depending on the location of the fractional extremum: either it is located in the beginning/end ($\ell \in \{1, k-1\}$), or somewhere in the middle ($1 < \ell < k-1$), or there

exists no fractional extremum at all. But no matter which of these structures the vector $x^*$ possesses, it follows that at least one of the constraints (2a) or (4a) must be tight for $(x^*, d^*, u^*)$. Indeed, let there be a positive slack $\delta$ in both constraints. If there exists a fractional extremum in $x^*$, from now on it is always assumed that this is a local maximum (otherwise, switch $t_\ell$ and $t_{\ell+1}$ in $d^+$, $u^+$, and negate the sign of $\varepsilon$ in $x^+$) and define

$$(*)\quad x_t^+ := \begin{cases} x_t^* + \varepsilon & t_\ell \le t < t_{\ell+1}, \\ x_t^* & \text{otherwise}, \end{cases} \quad d_t^+ := \begin{cases} d_t^* + \varepsilon & t = t_{\ell+1}, \\ d_t^* & \text{otherwise}, \end{cases} \quad u_t^+ := \begin{cases} u_t^* + \varepsilon & t = t_\ell, \\ u_t^* & \text{otherwise}. \end{cases}$$

From now on, let $(x^-, d^-, u^-)$ always be defined analogous to $(x^+, d^+, u^+)$, only with negated signs for $\varepsilon$. This definition ensures that

$$\tfrac{1}{2}(x^+, d^+, u^+) + \tfrac{1}{2}(x^-, d^-, u^-) = (x^*, d^*, u^*)\ ,$$

which contradicts the fact that $(x^*, d^*, u^*)$ is a vertex if both $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ are feasible. In order to give a proof by contradiction it thus suffices to show the feasibility of $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$.
Here, the feasibility of $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ follows for $\varepsilon$ chosen small enough, i.e., $\varepsilon < \delta$. To see this, observe that compared to $(x^*, d^*, u^*)$, the values on the left hand sides of (2a) and (4a) each decrease by $\varepsilon$ for $(x^-, d^-, u^-)$, whereas they increase by $\varepsilon$ for $(x^+, d^+, u^+)$.

If no fractional extremum exists, the argument is similar: Since $x^*$ is fractional by assumption, at least one of the values $x_0^*, x_T^*$ must be fractional. Assume w.l.o.g. that $x_0^* \in (0, 1)$ (the case $x_T^* \in (0, 1)$ works analogously) and define

$$x_t^+ := \begin{cases} x_t^* + \varepsilon & t < t_1, \\ x_t^* & \text{otherwise}, \end{cases} \quad d_t^+ := \begin{cases} d_t^* + (1 - x_{t_1}^*) \cdot \varepsilon & t = t_1, \\ d_t^* & \text{otherwise}, \end{cases} \quad u_t^+ := \begin{cases} u_t^* - x_{t_1}^* \cdot \varepsilon & t = t_1, \\ u_t^* & \text{otherwise}. \end{cases}$$

If $x_{t_1}^* = 1$, then the left hand side value of (4a) neither changes for $(x^+, d^+, u^+)$ nor for $(x^-, d^-, u^-)$, as an increase/decrease in $x_0^*$ is always canceled out by the decrease/increase in $u_{t_1}^*$. Both elements also satisfy (2a) for $\varepsilon < \delta$, as the value on the left hand side decreases by $\varepsilon$ for $(x^-, d^-, u^-)$, whereas it increases by $\varepsilon$ for $(x^+, d^+, u^+)$. The argument is symmetric for $x_{t_1}^* = 0$. Consequently, the two solutions $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ are feasible.

Thus, at least one of the two constraints (2a) or (4a) must be tight for $(x^*, d^*, u^*)$. This is exploited in the following case distinction, which is based on the location of the fractional extremum.

Case 1: *The fractional extremum is at the beginning or at the end, i.e.,* $\ell \in \{1, k-1\}$

Suppose that (2a) is tight. If $d^*$ is minimal (with respect to $x^*$ and constraints (6)) and the fractional extremum is located at the beginning (i.e., $\ell = 1$), then the fact

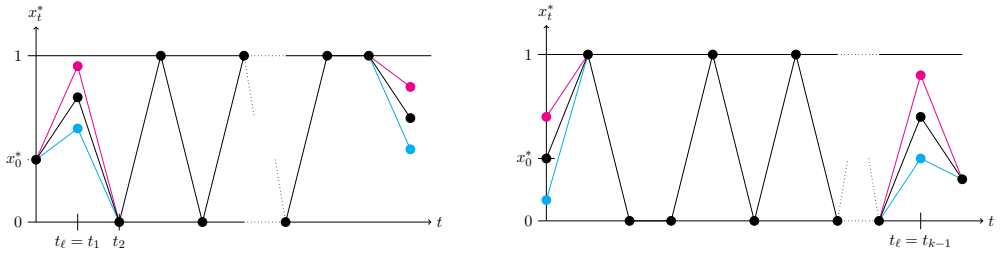
that (2a) is tight implies that the last change in the vector $x^*$ must be a fractional deactivation $d^*_{t_k} = x^*_{t_{k-1}} - x^*_{t_k} = 1 - x^*_T \in (0,1)$ and the last entry $x^*_T$ must also be fractional. In this case, begin with the definition $(*)$ and extend it as follows:

$$x^+_t := x^*_t + \varepsilon \quad \text{for} \quad t \geq t_k \,, \qquad d^+_t := d^*_t - \varepsilon \quad \text{for} \quad t = t_k \,.$$

If the fractional extremum is instead located at the end (i.e., $\ell = k-1$), then the fact that (2a) is tight implies that the entry $x^*_0$ must be fractional and that the first change in the vector $x^*$ must be a fractional activation $u^*_t \geq x^*_{t_1} - x^*_0 = 1 - x^*_0 \in (0,1)$. In this case, extend the definition $(*)$ as follows:

$$x^+_t := x^*_t + \varepsilon \quad \text{for} \quad t \leq t_1 \,, \qquad u^+_t := u^*_t - \varepsilon \quad \text{for} \quad t = t_1 \,.$$

In either case, $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ satisfy (2a), since the value on the left hand side does not change. If $u^*$ is minimal, too, then constraint (4a) cannot be tight for $(x^*, d^*, u^*)$, i.e., it has a positive slack $\delta > 0$. Compared to $(x^*, d^*, u^*)$, the left hand side of this constraint increases by $\varepsilon$ for $(x^+, d^+, u^+)$, whereas it decreases by $\varepsilon$ for $(x^-, d^-, u^-)$. Therefore, if $u^*$ is minimal, then $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ are feasible, as long as $\varepsilon$ is chosen smaller than the slack $\delta$. The figure below illustrates the idea behind the previous definitions of $(x^+, d^+, u^+)$ (magenta) and $(x^-, d^-, u^-)$ (cyan).




$\ell = 1$ $\qquad\qquad$ $\ell = k-1$

In case $u^*$ is not minimal, (4a) might be tight for $(x^*, d^*, u^*)$, so that the current definition of $(x^+, d^+, u^+)$ does not yield a feasible solution due to the increase by $\varepsilon$ on the left hand side of (4a). This, however, can be rectified by adding $u^+_i := u^*_i - \varepsilon$ to the current definition, where $i$ is the index of some *too large* value $u^*_i$, i.e., $u^*_i > x^*_i - x^*_{i-1} \geq 0$. Then, both $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ are feasible, if $\varepsilon$ is smaller than $u^*_i - (x^*_i - x^*_{i-1})$.

Now, if $d^*$ is not minimal, the fact that (2a) is tight does not imply anything about the structure of $x^*$ anymore. In this case, begin with the definition $(*)$ and extend it by $d^+_j := d^*_j - \varepsilon$, where $j$ is the index of some *too large* value $d^*_j$, i.e., $d^*_j > x^*_{j-1} - x^*_j \geq 0$. Again, the fact that $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ satisfy (2a) is easy to see. For $(x^+, d^+, u^+)$, the left hand side of (4a) increases by $\varepsilon$, whereas

is decreases by the same value for $(x^-, d^-, u^-)$. If (4a) is not tight for $(x^*, d^*, u^*)$, i.e., it has a positive slack $\delta > 0$, then $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ are feasible solutions, if $\varepsilon$ is smaller than the slack $\delta$. If (4a) is tight, then the current definition needs to be further modified in order to yield a feasible element. More precisely, if $u^*$ is not minimal, it suffices to add $u_i^+ := u_i^* - \varepsilon$ to the current definition, where $i$ is the index of some *too large* value $u_i^*$, i.e., $u_i^* > x_i^* - x_{i-1}^* \geq 0$ and to ensure that $\varepsilon$ does not exceed $u_i^* - (x_i^* - x_{i-1}^*)$. Otherwise, if $u^*$ is minimal, then it is possible to discover more about the structure of $x^*$ from the fact that (4a) is tight.

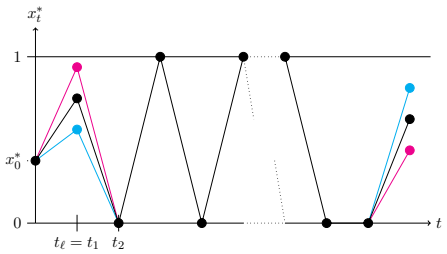
If the fractional extremum is located at the beginning (i.e., $\ell = 1$), then the fact that (4a) is tight implies that the last change in the vector $x^*$ must be a fractional activation $u_{t_k}^* \in (0, 1)$ and the last entry $x_T^*$ must also be fractional. In this case, extend the current definition of $(x^+, d^+, u^+)$ as follows:

$$x_t^+ := x_t^* - \varepsilon \quad \text{for} \quad t \geq t_k \,, \quad u_t^+ := u_t^* - \varepsilon \quad \text{for } t = t_k \,.$$

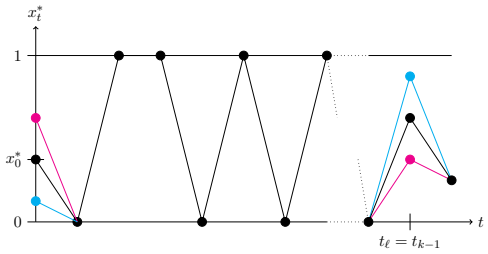
If the fractional extremum is instead located at the end (i.e., $\ell = k - 1$), then the fact that (4a) is tight implies that the entry $x_0^*$ must be fractional and that the first change in the vector $x^*$ must be a fractional deactivation $d_t^* \geq x_0^* - x_{t_1}^* = x_0^* - 0 \in (0, 1)$. In this case, extend the current definition of $(x^+, d^+, u^+)$ as follows:

$$x_t^+ := x_t^* - \varepsilon \quad \text{for} \quad t \leq t_1 \,, \quad d_t^+ := d_t^* - \varepsilon \quad \text{for } t = t_1 \,.$$

Now, the elements $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ satisfy both (2a) and (4a). The figure below illustrates this definition of $(x^+, d^+, u^+)$ (magenta) and $(x^-, d^-, u^-)$ (cyan) depending on the location of the fractional extremum.




$\ell = 1$




$\ell = k - 1$

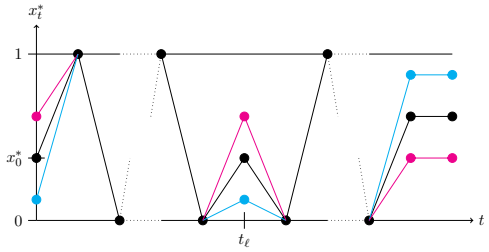
This concludes the case that (2a) is tight for $(x^*, d^*, u^*)$. The argument for the case that (4a) is tight is analogous.

Case 2: *The fractional extremum is somewhere in the middle, i.e.,* $1 < \ell < k - 1$
First of all, note that the case $x_0^* \in \{0, 1\}$ can be dealt with as described in Case 1. Indeed, if $x_0^* \in \{0, 1\}$ and (2a) is tight (and $d^*$ is minimal), then it follows $d_{t_k}^*, x_T^* \in (0, 1)$. Otherwise, (4a) being tight (and $u^*$ minimal) implies that it must hold $u_{t_k}^*, x_T^* \in (0, 1)$. Therefore, one can use the respective arguments from

the previous case and assume $x_0^* \in (0,1)$ from now on.
The subsequent case distinction is based on whether the first change in $x^*$ is an activation ($x_{t_1}^* = 1$) or a deactivation ($x_{t_1}^* = 0$). First, consider $x_{t_1}^* = 1$.
Let $x_0^* \in (0,1)$ and $x_{t_1}^* = 1$. If (4a) is tight and $u^*$ is minimal (regarding $x^*$ and (5)), then it must hold $u_{t_k}^* \in (0,1)$ and $x_T^* \in (0,1)$. In this case, extend the definition $(*)$ as follows:

$$x_t^+ := \begin{cases} x_t^* + \varepsilon & t < t_1, \\ x_t^* - \varepsilon & t \geq t_k, \end{cases} \qquad u_t^+ := \begin{cases} u_t^* - \varepsilon & t = t_1, \\ u_t^* - \varepsilon & t = t_k. \end{cases}$$

Then, $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ satisfy (2a) and (4a), since the values on the left hand sides do not change. The figure below illustrates the definition of $(x^+, d^+, u^+)$ (magenta) and $(x^-, d^-, u^-)$ (cyan).




If $u^*$ is not minimal, it is not ensured that $u_{t_k}^* \in (0,1)$ and $x_T^* \in (0,1)$. In this case, extend the definition $(*)$ as follows

$$x_t^+ := x_t^* + \varepsilon \quad \text{for } t < t_1, \qquad u_t^+ := \begin{cases} u_t^* - \varepsilon & t = t_1, \\ u_i^* - \varepsilon & t = i, \end{cases}$$

where $i$ is the index of some *too large* value $u_i^* > x_i^* - x_{i-1}^* \geq 0$. If $\varepsilon$ does not exceed $u_i^* - (x_i^* - x_{i-1}^*)$, then $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ are feasible solutions and satisfy both (2a) and (4a), as the values on the respective left hand sides do not change.
Next, assume that (2a) is tight for $(x^*, d^*, u^*)$. If (4a) is also tight, then one can argue as described before. If, however, (4a) is not tight, then it has a positive slack $\delta$ and it suffices to extend the definition $(*)$ as follows:

$$x_t^+ := x_t^* + \varepsilon \quad \text{for } t < t_1, \qquad u_t^+ := u_t^* + \varepsilon \quad \text{for } t = t_1 \,.$$

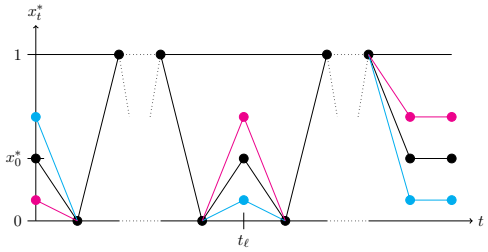
The value on the left hand side of (2a) does not change for $(x^+, d^+, u^+)$ nor for $(x^-, d^-, u^-)$. The value on the left hand side of (4a) decreases by $\varepsilon$ for $(x^-, d^-, u^-)$, whereas it increases by $\varepsilon$ for $(x^+, d^+, u^+)$. Therefore, the solutions are feasible if $\varepsilon < \delta$.

The reverse case in which $x_{t_1}^* = 0$ works with symmetric arguments and is

therefore only sketched briefly in the following. If (2a) is tight and $d^*$ is minimal, it must hold $d^*_{t_k}, x^*_T \in (0,1)$. Then, extend the definition $(*)$ as follows:

$$x_t^+ := \begin{cases} x_t^* - \varepsilon & t < t_1, \\ x_t^* + \varepsilon & t \geq t_k, \end{cases} \qquad d_t^+ := \begin{cases} d_t^* - \varepsilon & t = t_1, \\ d_t^* - \varepsilon & t = t_k. \end{cases}$$

This yields feasible solutions $(x^+, d^+, u^+)$, $(x^-, d^-, u^-)$. The sketch below depicts the idea behind the definition of $(x^+, d^+, u^+)$ (magenta) and $(x^-, d^-, u^-)$ (cyan).




If $d^*$ is not minimal, then there exists some *too large* value $d^*_j > x^*_{j-1} - x^*_j \geq 0$ that can be used to balance out the otherwise occurring increase on the left hand side of (2a). If (4a) is tight and (2a) is tight, too, then the previous argument still works. If (4a) is tight and (2a) is not tight, then extend the definition $(*)$ as follows

$$x_t^+ := x_t^* - \varepsilon \quad \text{for } t < t_1, \qquad d_t^+ := d_t^* - \varepsilon \quad \text{for } t = t_1\ ,$$

and ensure that $\varepsilon$ does not exceed the slack in constraint (2a).

Case 3: *there is no fractional extremum*
If $x_0^* \in \{0,1\}$, then it must hold that $x_T^* \in (0,1)$, since $x^*$ is not integer by assumption. In case $x^*_{t_{k-1}} = 0$, it must hold $u^*_{t_k} > 0$, even if $u^*$ is minimal. Then, define

$$x_t^+ = \begin{cases} x_t^* + \varepsilon & t \geq t_k, \\ x_t^* & \text{otherwise}, \end{cases} \qquad d^+ := d^*, \qquad u_t^+ = \begin{cases} u_t^* + \varepsilon & t = t_k, \\ u_t^* & \text{otherwise}. \end{cases}$$

If $u^*$ is minimal, there must be a positive slack $\delta$ in (4a) and the above definition yields feasible elements $(x^+, d^+, u^+)$,$(x^-, d^-, u^-)$ if $\varepsilon < \delta$. If $u^*$ is not minimal, additionally include $u_j^+ := u_j^* - \varepsilon$ for some $j$ with $u_j^* > x_j^* - x_{j-1}^*$ and make sure that $\varepsilon < u_j^* - (x_j^* - x_{j-1}^*)$. Otherwise, if $x^*_{t_{k-1}} = 1$, it must hold $d^*_{t_k} > 0$, even if $d^*$ is minimal. Then, define

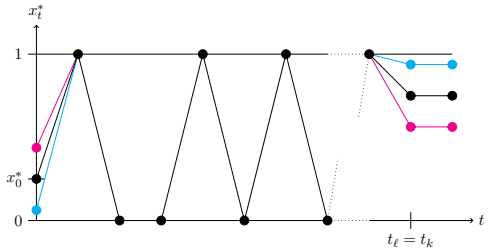
$$x_t^+ = \begin{cases} x_t^* - \varepsilon & t \geq t_k, \\ x_t^* & \text{otherwise}, \end{cases} \qquad d^+ := \begin{cases} d_t^* + \varepsilon & t = t_k, \\ d_t^* & \text{otherwise}, \end{cases} \qquad u^+ := u^*\ .$$

The argument for the feasibility is analogous.
Since the case $x_0^* \in \{0,1\}$ is now settled, let $x_0^* \in (0,1)$ and assume that the first

change in $x^*$ corresponds to an activation, i.e., $x^*_{t_1} = 1$. If (2a) is tight and $d^*$ is minimal, then it must hold $d^*_{t_k} \in (0,1)$ and $x^*_T \in (0,1)$. Now, define

$$x^+_t = \begin{cases} x^*_t + \varepsilon & t < t_1, \\ x^*_t - \varepsilon & t \geq t_k, \\ x^*_t & \text{otherwise}, \end{cases} \qquad d^+_t = \begin{cases} d_t + \varepsilon & t = t_k, \\ d^*_t & \text{otherwise}, \end{cases} \qquad u^+_t = \begin{cases} u^*_t - \varepsilon & t = t_1, \\ u^*_t & \text{otherwise}. \end{cases}$$

The feasibility of $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ is easy to see. The sketch below illustrates the idea behind this definition of $(x^+, d^+, u^+)$ (magenta) and $(x^-, d^-, u^-)$ (cyan).

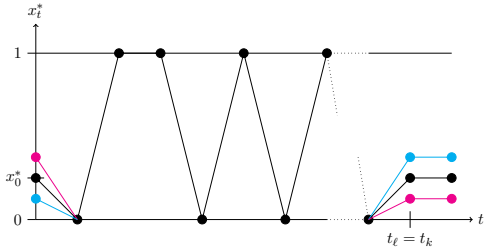



Now, finally, if $d^*$ is not minimal, then it is not ensured that the last change in $x^*$ is a (fractional) deactivation. In this case, however, remove all definitions for the index $t_k$ in the previous expression and instead include $d^+_j := d^*_j + \varepsilon$ for some *too large* value $d^*_i > x^*_{i-1} - x^*_i \geq 0$. This yields a feasible solution as long as $\varepsilon < d^*_i - |x^*_{i-1} - x^*_i|$. Now, assume that (4a) is tight. If (2a) is also tight, argue as before. If (2a) is not tight, remove all definitions for the index $t_k$ from the above expression and yield a feasible elements $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$, as long as $\varepsilon$ is smaller than the slack in (2a).

The reverse case in which $x^*_0 \in (0,1)$ and the first change in $x^*$ corresponds to a deactivation is symmetric and is therefore only briefly sketched. If (4a) is tight and $u^*$ is minimal, then it must hold $u^*_{t_k}, x^*_T \in (0,1)$. Then, define

$$x^+_t = \begin{cases} x^*_t + \varepsilon & t < t_1, \\ x^*_t - \varepsilon & t \geq t_k, \\ x^*_t & \text{otherwise}, \end{cases} \qquad d^+_t = \begin{cases} d_t + \varepsilon & t = t_1, \\ d^*_t & \text{otherwise}, \end{cases} \qquad u^+_t = \begin{cases} u^*_t - \varepsilon & t = t_k, \\ u^*_t & \text{otherwise}. \end{cases}$$

The sketch below illustrates the idea behind this definition of $(x^+, d^+, u^+)$ (magenta) and $(x^-, d^-, u^-)$ (cyan).




If $u^*$ is not minimal, then remove all definitions for the index $t_k$ in the above expression and instead include $u_i^+ := u_i^* - \varepsilon$ for some *too large* value $u_i^*$. If (2a) is tight and (4a) is also tight, then one may use the former argument. If (2a) is tight, but (4a) is not, then just remove all definitions for the index $t_k$ from the above expression and yield a feasible elements $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$, as long as $\varepsilon$ is smaller than the slack in (4a).

This concludes the case distinction and shows that the vertex $(x^*, d^*, u^*)$ must have an integer $x^*$. It remains to show that this integrality also implies the integrality of the vectors $d^*$ and $u^*$. This is done in the following second part of the proof.

Suppose that $x^*$ is integer, but either $d^*$ or $u^*$ is not entirely integer. Thus, there exists at least one index $i \in \{1, ..., T\}$ with $d_i^* \notin \mathbb{Z}$ (the argument is analogous in case $u^*$ is not integer). In particular, this implies that $d_i^*$ is not minimal with respect to $x^*$ and (6), i.e., $d_i^* > x_{i-1}^* - x_i^* \geq 0$. Now either, the constraint (2a) holds with equality, or it has a positive slack $\delta$. In the first case, there must exist a second fractional entry $d_j^* \notin \mathbb{Z}$ which therefore cannot be minimal regarding $x^*$, i.e., $d_j^* > x_{j-1}^* - x_j^* \geq 0$. Now, define

$$d_t^+ := \begin{cases} d_t^* + \varepsilon & t = i, \\ d_t^* - \varepsilon & t = j, \\ d_t^* & \text{otherwise}, \end{cases} \quad \text{and} \quad d_t^- := \begin{cases} d_t^* - \varepsilon & t = i, \\ d_t^* + \varepsilon & t = j, \\ d_t^* & \text{otherwise}. \end{cases}$$

It is easy to see that $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ are both feasible for $\varepsilon$ smaller than $\min\{d_i^* - (x_{i-1}^* - x_i^*), d_j - (x_{j-1}^* - x_j^*)\}$. If constraint (2a) is not tight, then it is not ensured that there exists a second fractional value in $d^*$ other than $d_i^*$. In this case, it suffices to delete the definitions for the index $j$ from the above expression to yield feasible elements $(x^+, d^+, u^+)$ and $(x^-, d^-, u^-)$ for $\varepsilon < \min\{d_i^* - (x_{i-1}^* - x_i^*), \delta\}$. This shows the integrality of $d^*$. The integrality of $u^*$ follows with analogous arguments using the constraint (4a). □

This concludes the study of extended models and, in a natural way, ensues the question how a perfect formulation of $(\text{BND}_{\text{var}}^{(1)})$ in the original space of only the $x$-variables looks like. Such a description in the original space can, for example, be obtained by projection of the polyhedra that correspond to the continuously

relaxed extended models. In fact, Fourier-Motzkin elimination of the deactivation and activation variables in (EXT-BND) yields

**Theorem 3.3.2.** For $1 \leq S \leq T$, consider any index set $\{t_0, ..., t_m\} \subseteq \{0, ..., T\}$ with $t_j \leq t_{j+1}$ for $j = 0, ..., m-1$ such that $m$ is odd if and only if $S$ is even, and let $m \geq S$. Define the following inequalities which will be called *alternating inequalities*:

$$\sum_{j=0}^{m}(-1)^{j+1}x_{t_j} \leq \lfloor S/2 \rfloor \quad (\mathrm{ALT}_0^\Omega) \qquad \text{and} \qquad \sum_{j=0}^{m}(-1)^{j}x_{t_j} \leq \lceil S/2 \rceil \quad (\mathrm{ALT}_1^\Omega)\ .$$

Let $P$ be the polytope defined by these alternating inequalities and the trivial inequalities $x \in [0,1]^{T+1}$. Then, $P = \mathrm{conv}(\Omega)$.

*Proof.* Consider the continuous relaxation of (EXT-BND) and recall that constraints (2a) and (4a) are redundant if $S$ is even, whereas (1a) and (3a) are redundant if $S$ is odd. The redundant constraints can be removed from the model.

Furthermore, observe that one may perform the elimination of the deactivation variables $d$ completely independent of the elimination of the activation variables $u$, as no inequality in (EXT-BND) contains both of them simultaneously. Thus, the constraints involving the activation variables $u$ are ignored in the elimination of the deactivation variables $d$ that is described in the following.

Recall that the set of constraints in the continuous relaxation of (EXT-BND) that involve only $d$ is given by

$$-x_0 + x_T + \sum_{t=1}^{T} d_t \leq \lfloor \tfrac{S-1}{2} \rfloor + 1 \quad (1a)\ , \qquad -x_0 + \sum_{t=1}^{T} d_t \leq \lfloor \tfrac{S}{2} \rfloor \quad (2a)\ ,$$
$$x_{t-1} - x_t - d_t \leq 0 \qquad t = 1, ..., T\ , \qquad -d_t \leq 0 \qquad t = 1, ..., T\ .$$

Consider an iteration of the Fourier-Motzkin method which eliminates a variable $d_\ell$, $\ell \in \{1, ..., T\}$. Let $K \subseteq \{1, ..., T\}$ be the set of indices of deactivation variables that have been eliminated in earlier iterations of the Fourier-Motzkin method. Furthermore, assume that $S$ is even, i.e., constraint (2a) is removed, due to its redundancy. Then, the set of linear inequalities that has been obtained by the method up until the current iteration $\ell$ satisfies the following:

1. The variable $d_\ell$ only occurs negatively on the left hand side in two constraints, namely the deactivation constraint $x_{\ell-1} - x_\ell - d_\ell \leq 0$ and the non-negativity constraint $-d_\ell \leq 0$.

2. The variable $d_\ell$ occurs positively on the left hand side in all of the following inequalities:

$$-x_0 + x_T + \sum_{j\in J}(x_{j-1} - x_j) + \sum_{j\in\{1,...,T\}\setminus K} d_j \leq \lfloor \tfrac{S-1}{2} \rfloor + 1 \qquad \text{for all } J \subseteq K \ .$$

3. All other inequalities do not contain $d_\ell$ and are given by $x_{t-1} - x_t - d_t \leq 0$, $-d_t \leq 0$ for all $t \in \{1, ..., T\} \setminus (K \cup \{\ell\})$.

For the elimination of $d_\ell$, both inequalities in 1. have to be combined with all inequalities in 2. The set of inequalities that is obtained by the elimination of $d_\ell$ is again of the form as described by 1.-3. with $K := K \cup \{\ell\}$. It is easy to see that the number of constraints roughly doubles with every elimination of a variable $d_\ell$. After all deactivation variables have been eliminated, the set of remaining inequalities is thus given by

$$\sum_{j=0}^{m}(-1)^{j+1}x_{t_j} \leq \lfloor \tfrac{S-1}{2} \rfloor + 1 = \tfrac{S}{2} = \lfloor \tfrac{S}{2} \rfloor$$

for any index set $\{t_0, ..., t_m\} \subseteq \{0, ..., T\}$ with $t_j \leq t_{j+1}$ for $j = 0, ..., m-1$ such that $m$ is odd.

If $S$ is instead assumed to be odd, the argument is analogous with the only difference that, in this case, the constraint (1a) is removed and thus, the set of inequalities in 2. is instead given by

$$-x_0 + \sum_{j\in J}(x_{j-1} - x_j) + \sum_{j\in\{1,...,T\}\setminus K} d_j \leq \lfloor \tfrac{S}{2} \rfloor \quad \text{for all } J \subseteq K \ .$$

Thus, the elimination of $d$ for an odd $S$ yields

$$\sum_{j=0}^{m}(-1)^{j+1}x_{t_j} \leq \lfloor \tfrac{S}{2} \rfloor$$

for any index set $\{t_0, ..., t_m\} \subseteq \{0, ..., T\}$ with $t_j \leq t_{j+1}$ for $j = 0, ..., m-1$ such that $m$ is even.

Next, consider the projection of the constraints in the continuous relaxation of (EXT-BND) that involve only $u$. Recall that those inequalities are given by

$$+ x_0 - x_T + \sum_{t=1}^{T} u_t \leq \lfloor \tfrac{S-1}{2} \rfloor + 1 \quad (3a) \ , \qquad +x_0 + \sum_{t=1}^{T} u_t \leq \lfloor \tfrac{S}{2} \rfloor + 1 \quad (4a) \ ,$$

$$x_t - x_{t-1} - u_t \leq 0 \qquad t = 1, ..., T \ , \qquad -u_t \leq 0 \qquad t = 1, ..., T \ .$$

Next, consider the elimination of the activation variables $u$, using the notation from before. If $S$ is even, the set of linear inequalities that has been obtained by Fourier-Motzkin elimination of $u$ up until the current iteration $\ell$, in which $u_\ell$ is eliminated, satisfies the following:

1. The variable $u_\ell$ only occurs negatively on the left hand side in two constraints, namely the activation constraint $x_\ell - x_\ell - u_{\ell-1} \leq 0$ and the non-negativity constraint $-u_\ell \leq 0$.

2. The variable $u_\ell$ occurs positively on the left hand side in all of the following inequalities

$$+x_0 - x_T + \sum_{j \in J}(x_j - x_{j-1}) + \sum_{j \in \{1,\dots,T\} \setminus K} u_j \leq \lfloor \tfrac{S-1}{2} \rfloor + 1 \quad \text{for all } J \subseteq K \,.$$

3. All other inequalities do not contain $u_\ell$ and are given by $x_t - x_{t-1} - u_t \leq 0$, $-u_t \leq 0$ for all $t \in \{1, ..., T\} \setminus (K \cup \{\ell\})$.

Thus, the elimination of all activation variables $u$ yields the inequalities

$$\sum_{j=0}^{m} (-1)^j x_{t_j} \leq \lfloor \tfrac{S-1}{2} \rfloor + 1 = \tfrac{S}{2} = \lceil \tfrac{S}{2} \rceil$$

for any index set $\{t_0, ..., t_m\} \subseteq \{0, ..., T\}$ with $t_j \leq t_{j+1}$ for $j = 0, ..., m-1$ such that $m$ is odd. If $S$ is odd, the argument is analogous with the only difference that, in this case, the constraint (3a) is removed and thus, the set of inequalities in 2. is instead given by

$$+x_0 + \sum_{j \in J}(x_j - x_{j-1}) + \sum_{j \in \{1,\dots,T\} \setminus K} u_j \leq \lfloor \tfrac{S}{2} \rfloor + 1 \quad \text{for all } J \subseteq K \,.$$

Thus, for an odd $S$, the elimination of $d$ yields

$$\sum_{j=0}^{m} (-1)^{j+1} x_{t_j} \leq \lfloor \tfrac{S}{2} \rfloor + 1 = \lceil \tfrac{S}{2} \rceil$$

for any index set $\{t_0, ..., t_m\} \subseteq \{0, ..., T\}$ with $t_j \leq t_{j+1}$ for $j = 0, ..., m-1$ such that $m$ is even. It is easy to see that the inequalities ($\text{ALT}_0^\Omega$) and ($\text{ALT}_1^\Omega$) are redundant if for $m < S$, hence they may be omitted. This shows the claim. □

Note that the extended model (EXT-BND) is defined in dimension $3T + 1$ and the number of its constraints is linear in $T$ as well; it is thus only slightly larger than the original model ($\text{BND}_{\text{var}}^{(1)}$). On the contrary, the respective complete description in the original space by the inequalities defined in Theorem 3.3.2 has an exponential number of constraints.

A perfect formulation of $\text{conv}(\Omega_0)$ and $\text{conv}(\Omega_1)$ in the original space can also be obtained by projection of the polyhedra of the corresponding extended models (EXT-$\text{BND}_0$) and (EXT-$\text{BND}_1$) onto the $x$-space. However, this thesis will refrain from doing so to avoid redundancies. In fact, the next subsection will devise a primal-dual optimization algorithm for the problem ($\text{BND}_{\text{var},0}^{(1)}$), and the proof of its

correctness will actually also serve as a proof of a complete description for $\text{conv}(\Omega_0)$. Therefore, this section will now present the relevant inequalities and only show their validity for $\text{conv}(\Omega_0)$ and $\text{conv}(\Omega_1)$. Note that the index sets defined below are now subsets of $\{1, ..., T\}$, i.e., the first stage $t = 0$ is excluded.

**Lemma 3.3.7.** For $1 \leq S \leq T$, consider any index set $\{t_1, ..., t_m\} \subseteq \{1, ..., T\}$ with $t_j < t_{j+1}$ for $j = 1, ..., m-1$ where $m$ is odd if and only if $S$ is even, and $m \geq S$. Then, the following alternating inequalites are valid for $\text{conv}(\Omega_0)$ and $\text{conv}(\Omega_1)$, respectively:

$$(\text{ALT}_0) \qquad \sum_{j=1}^{m} (-1)^{j+1} x_{t_j} \leq \left\lfloor \tfrac{S}{2} \right\rfloor \qquad \text{for conv}(\Omega_0) \quad \text{and}$$

$$(\text{ALT}_1) \qquad \sum_{j=1}^{m} (-1)^{j} x_{t_j} \leq \left\lceil \tfrac{S}{2} \right\rceil - 1 \qquad \text{for conv}(\Omega_1)\ .$$

*Proof.* Consider the inequality $(\text{ALT}_0)$ for some index set $\{t_1, ..., t_m\}$ with the required properties. It suffices to prove that the inequality $\sum_{j=1}^{m} (-1)^{j+1} x_{t_j} \leq \lfloor S/2 \rfloor$ is valid for each $x \in \Omega_0$. Observe that for $m < S$, the inequality would be redundant as the number of positive coefficients on the left hand side of $(\text{ALT}_0)$ is at most $\lceil m/2 \rceil$. So suppose that $m \geq S$ and that there is an element $\tilde{x} \in \Omega_0$ such that

$$\sum_{j=1}^{m} (-1)^{j+1} \tilde{x}_{t_j} \geq \left\lfloor \tfrac{S}{2} \right\rfloor + 1\ .$$

In case $S$ is odd and $m$ is even, there must be at least $\lfloor S/2 \rfloor + 1 = S+1/2$ many indices $t_j$, $j \in \{1, 3, 5, ..., m\}$, such that $\tilde{x}_{t_j} = 1$ and $\tilde{x}_{t_{j+1}} = 0$, since the last coefficient on the left hand side is $-1$. Consequently, $\tilde{x}$ has a variation of at least $S + 1$, which contradicts $\tilde{x} \in \Omega_0$.

In case $S$ is even and $m$ is odd, the reasoning is very similar: either there are at least $S/2 + 1$ indices $t_j$, $j \in \{1, 3, 5, ..., m\}$, with $\tilde{x}_{t_j} = 1$ and $\tilde{x}_{t_{j+1}} = 0$, or there are only $S/2$ indices with this property, but $x_{t_m} = 1$. Either way, $\tilde{x}$ has a variation of more than $S$.

The validity of $(\text{ALT}_1)$ for $\text{conv}(\Omega_1)$ immediately follows due the symmetry described in Remark 3.3.1

$$-\left(\sum_{j=1}^{m} (-1)^{j+1} x_{t_j}\right) \leq \lfloor S/2 \rfloor - \sum_{j=1}^{m} (-1)^{j+1} \Leftrightarrow \sum_{j=1}^{m} (-1)^{j} x_{t_j} \leq \lfloor S/2 \rfloor - \begin{cases} 0, & S \text{ odd} \\ 1, & S \text{ even} \end{cases}$$

$$\Leftrightarrow \sum_{j=1}^{m} (-1)^{j} x_{t_j} \leq \begin{cases} \frac{S-1}{2}, & S \text{ odd} \\ \frac{S}{2} - 1, & S \text{ even} \end{cases} \Leftrightarrow \sum_{j=1}^{m} (-1)^{j} x_{t_j} \leq \left\lceil \tfrac{S}{2} \right\rceil - 1\ .$$

□

The proof that these alternating inequalities ($\text{ALT}_0$) and ($\text{ALT}_1$) together with the trivial inequalities completely describe the polytopes $\text{conv}(\Omega_0)$ and $\text{conv}(\Omega_1)$ will be given in the next subsection. The mode in which this will be proven will differ significantly from the previous techniques. In fact, instead of using arguments involving total unimodularity or convex combinations, the following section will rely on LP duality arguments.

### 3.3.2 Merging Algorithm and Complete Description

The purpose of this subsection is twofold: on the one hand, it presents a primal-dual optimization algorithm for ($\text{BND}^{(1)}_{\text{var},0}$), on the other hand, it will show that the trivial bounds on $x$ and the alternating inequalities ($\text{ALT}_0$), ($\text{ALT}_1$) defined in the previous section yield respective perfect formulations of the polytopes $\text{conv}(\Omega_0)$ and $\text{conv}(\Omega_1)$. Note that due to the symmetry reasons, it suffices to consider the version with fixation to zero in the beginning. Thus, this section will focus on the polytope $P_0$ that consists of the alternating inequalities ($\text{ALT}_0$) and the trivial inequalities and it will show $P_0 \subseteq \text{conv}(\Omega_0)$ by proving the integrality of $P_0$. Observe that it suffices to show $P_0 = \text{conv}(\Omega_0)$, since the reverse the inclusion $\text{conv}(\Omega_0) \subseteq P_0$ is obvious by validity of the alternating inequalities ($\text{ALT}_0$).

**Theorem 3.3.3.** Let $P_0$ denote the polytope that is defined by the linear inequalities $x_0 = 0$, $x_t \in [0,1]$ for all $t = 1, ..., T$, and the alternating inequalities ($\text{ALT}_0$). Then,

$$\text{conv}(\Omega_0) = P_0 \ .$$

The above theorem and Remark 3.3.1 immediately imply

**Theorem 3.3.4.** Let $P_1$ denote the polytope that is defined by the linear inequalities $x_0 = 1$, $x_t \in [0,1]$ for all $t = 1, ..., T$, and the alternating inequalities ($\text{ALT}_1$). Then,

$$\text{conv}(\Omega_1) = P_1 \ .$$

The remainder of this subsection is dedicated to the proof of Theorem 3.3.3. This aim is achieved using LP-duality: It will be shown that for any objective $c$ there exists an *integer* optimizer of $c^\top x$ over $P_0$, by simultaneously constructing a dual solution attaining the same objective value. In other words, a primal-dual optimization algorithm will be presented. Proving the correctness of the latter then simultaneously proves Theorem 3.3.3.

Since the variable $x_0$ is fixed to zero in Theorem 3.3.3 and does not appear in any alternating inequality ($\text{ALT}_0$), it may as well be deleted in ($\text{BND}^{(1)}_{\text{var},0}$).

Let then $p$ denote the total number of alternating inequalities and let the system $Ax \leq \lfloor \frac{S}{2} \rfloor \cdot \mathbb{1}_p$ collect all of them, where now $x := (x_1, ..., x_T)^\top$ and $c := (c_1, ..., c_T)^\top$. Then, replace the feasible region of $(\mathrm{BND}^{(1)}_{\mathrm{var},0})$ by $P_0$ and consider the resulting linear problem, which can be written in primal (left) and dual (right) form as follows:

$$(\text{OPT-P}_0) \qquad \begin{cases} \max & c^\top x \\ \text{s.t.} & Ax \leq \lfloor S/2 \rfloor \cdot \mathbb{1}_p \\ & x \leq \mathbb{1}_T,\ x \in \mathbb{R}^T_{\geq 0} \end{cases} \qquad \begin{cases} \min & \lfloor S/2 \rfloor \mathbb{1}_p^\top y + \mathbb{1}_T^\top z \\ \text{s.t.} & A^\top y + z \geq c \\ & y \in \mathbb{R}^p_{\geq 0}, z \in \mathbb{R}^T_{\geq 0} \end{cases}$$

Furthermore, assume that $S < T$, as otherwise the alternating inequalities in (OPT-P$_0$) are redundant, the existence of an integer optimum solution for (OPT-P$_0$) is obvious and, in particular, there is no need for an optimization algorithm. Moreover, the integrality of $P_0$ follows immediately. Similarly, assume $S > 0$ since otherwise, the existence of an optimal integer primal solution is also trivial.
As previously stated, the integrality of $P_0$ is proven by constructing, for a given objective $c \in \mathbb{R}^T$, a feasible integer primal solution and a dual solution with the same objective value. This is achieved by *merging* or *deleting* variables which reduces the given problem instance step-wise up to a recursion root, for which an integer optimum primal and dual solution is easily determined. With the knowledge of these solutions, the previous reduction process is then reversed so as to obtain an integer optimal primal and optimum dual solution for the original instance. The pseudocode below describes the outline of the merging algorithm. The structure of this subsection will now follow the course of the algorithm: one after the other, the lemmas making up the algorithm's steps are presented and proven in the order in which they occur in the algorithm.

**Algorithm 2:** Sketch of primal-dual algorithm

**Input:** $T \in \mathbb{N}_{\geq 0}$, $S \in \{1, ..., T-1\}$, $c \in \mathbb{R}^T$
**Output:** optimal integer solution $x^*$ for (OPT-P$_0$) (and corresponding dual solution $(y^*, z^*)$)

```
apply the preprocessing steps described in Definition 3.3.3;
if S = T − 1 then
    obtain integer primal and dual solution according to Lemma 3.3.9;
else
    apply Merging Rule 3.3.4;
    recursively call Algorithm 2 for the merged instance;
extend the integer primal and dual solution according to the proof of
  Theorem 3.3.6;
```

In order to capture the *merging* of variables and to be able to reverse this merging process, the following method of indexing the variables is introduced.

**Definition 3.3.2.** Refer to instances of (OPT-P$_0$) via $(c, \mathcal{T})$. Here, $c$ denotes the objective, and the finite set $\mathcal{T}$ is a family of sets used to index the dual variables. The elements $J \in \mathcal{T}$ are assumed to be ordered and thus, for $1 \leq t \leq |\mathcal{T}|$, let $\mathcal{T}(t)$ denote the $t$-th element of $\mathcal{T}$. The objective $c$ and the vector $x$ are indexed with the corresponding elements $J \in \mathcal{T}$. As each alternating inequality is uniquely determined by the elements contained in its support $\mathcal{J} \subseteq \mathcal{T}$, the corresponding dual variable $y$ is indexed with the subset $\mathcal{J}$. The other dual variable $z$ is indexed by $J \in \mathcal{T}$.

For a better grasp of Definition 3.3.2, a small example is given below.

**Example 3.3.4.** Consider an instance of (OPT-P$_0$) with $T = 4$, $S = 2$, and the set family $\mathcal{T} = \{\{1\}, \{2\}, \{3\}, \{4\}\}$. Then, the alternating inequality

$$x_{\{1\}} - x_{\{2\}} + x_{\{3\}} \leq 1$$

is uniquely determined by $\mathcal{J} = \{\{1\}, \{2\}, \{3\}\} \subseteq \mathcal{T}$, which is also the index of the corresponding dual variable. The (non-trivial) dual constraints are thus given by:

$$\begin{array}{rrrrrrrrrrrr}
+y_{\{\{1\},\{2\},\{3\}\}} & + y_{\{\{1\},\{2\},\{4\}\}} & +y_{\{\{1\},\{3\},\{4\}\}} & & & +z_{\{1\}} & & & & & & \geq c_{\{1\}} \\
-y_{\{\{1\},\{2\},\{3\}\}} & - y_{\{\{1\},\{2\},\{4\}\}} & & & + y_{\{\{2\},\{3\},\{4\}\}} & & + z_{\{2\}} & & & & & \geq c_{\{2\}} \\
+y_{\{\{1\},\{2\},\{3\}\}} & & -y_{\{\{1\},\{3\},\{4\}\}} & - y_{\{\{2\},\{3\},\{4\}\}} & & & & +z_{\{3\}} & & & & \geq c_{\{3\}} \\
& + y_{\{\{1\},\{2\},\{4\}\}} & +y_{\{\{1\},\{3\},\{4\}\}} & + y_{\{\{2\},\{3\},\{4\}\}} & & & & & + z_{\{4\}} & & & \geq c_{\{4\}}
\end{array}$$

Given an arbitrary instance $(c, \mathcal{T})$, the merging algorithm 2 will first perform the preprocessing steps which are defined next. Their purpose is to transform the given instance into a *simpler* instance, i.e., an instance where the objective has a very specific structure. The properties of this objective will then be the basis of the entire merging idea. While it is obvious that the following preprocessing step can be performed for any given instance, the fact that the resulting instance of (BND$^{(1)}_{\text{var},0}$) is actually equivalent to the given one is not. The proof of this will be contained in the proof of Theorem 3.3.6 which will conclude the description of the algorithm's steps.

**Definition 3.3.3** (Preprocessing)**.** Given an instance of (OPT-P$_0$), use the indexing method described in Definition 3.3.2, i.e., $\mathcal{T} = \{\{1\}, ...., \{T\}\}$ and define corresponding objective coefficients by $c_{\{j\}} := c_j$ for $j = 1, ..., T$. Then,

1. *Delete all zero-entries:*
   Set $\mathcal{T}^{(1)} := \{J \in \mathcal{T} \mid c_J \neq 0\}$ and $c^{(1)}_J := c_J$ for all $J \in \mathcal{T}^{(1)}$.

2. *Delete the first sequence of negative entries in the objective:*
   For the first positive entry $l$ in $c_{\mathcal{T}^{(1)}}$, set

$$\mathcal{T}^{(2)} := \{\mathcal{T}^{(1)}(t) \mid t \geq l\} \text{ and } c_J^{(2)} := c_J^{(1)} \text{ for all } J \in \mathcal{T}^{(2)} .$$

3. *Combine each sequence of entries with the same sign into a single element:*
   For all maximal sequences $c_{\mathcal{T}^{(2)}(k)}, ..., c_{\mathcal{T}^{(2)}(j)}$ of equal sign, define

$$M := \bigcup_{k \leq t \leq j} \mathcal{T}^{(2)}(t) \text{ and } \mathcal{T}^{(3)} := \mathcal{T}^{(2)} \setminus \bigcup_{k \leq t \leq j} \{\mathcal{T}^{(2)}(t)\} \cup \{M\}$$

   as well as $c_M^{(3)} := \sum_{t=k}^{j} c^{(2)}_{\mathcal{T}^{(2)}(t)}$, $c_J^{(3)} := c_J^{(2)}$ for $J \in \mathcal{T}^{(3)}$.

4. *Delete the last sequence of entries with sign* $(-1)^{S+1}$*:*
   For the last $l$ with $\operatorname{sgn}(c_{\mathcal{T}^{(3)}(l)}) = (-1)^S$, set

$$\mathcal{T}^{(4)} := \{I^{(3)}(t) \mid t \leq l\} \text{ and } c_J^{(4)} := c_J^{(3)} \text{ for all } J \in \mathcal{T}^{(4)} .$$

The example below demonstrates the preprocessing steps.

**Example 3.3.5.** Let the following instance with $S = 3$ be given:

$$\begin{aligned} c &= (-3, 1, 3, -6, 5, 0, -7, 4, -2, 3, -9, 11, -1, 2) \\ \mathcal{T} &= \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}, \{9\}, \{10\}, \{11\}, \{12\}, \{13\}, \{14\}\} \end{aligned}$$

The instances in the preprocessing steps then read as follows:

$$\begin{aligned} c^{(1)} &= (-3, 1, 3, -6, 5, -7, 5, -2, 3, -9, 11, -1, 2) , \\ \mathcal{T}^{(1)} &= \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{7\}, \{8\}, \{9\}, \{10\}, \{11\}, \{12\}, \{13\}, \{14\}\} , \\ \\ c^{(2)} &= (1, 3, -6, 5, -7, 5, -2, 3, -9, 11, -1, 2) , \\ \mathcal{T}^{(2)} &= \{\{2\}, \{3\}, \{4\}, \{5\}, \{7\}, \{8\}, \{9\}, \{10\}, \{11\}, \{12\}, \{13\}, \{14\}\} , \\ \\ c^{(3)} &= (4, -6, 5, -7, 5, -2, 3, -9, 11, -1, 2) , \\ \mathcal{T}^{(3)} &= \{\{2, 3\}, \{4\}, \{5\}, \{7\}, \{8\}, \{9\}, \{10\}, \{11\}, \{12\}, \{13\}, \{14\}\} , \\ \\ c^{(4)} &= (4, -6, 5, -7, 5, -2, 3, -9, 11, -1) , \\ \mathcal{T}^{(4)} &= \{\{2, 3\}, \{4\}, \{5\}, \{7\}, \{8\}, \{9\}, \{10\}, \{11\}, \{12\}, \{13\}\} . \end{aligned}$$

The following lemma is an obvious consequence of the preprocessing steps.

**Lemma 3.3.8.** Given any instance, let $(c, \mathcal{T})$ be the resulting instance obtained from the given instance by the preprocessing. Then, this instance $(c, \mathcal{T})$ has the following properties:

$$
\begin{array}{lll}
 & (i) & c_J \neq 0 \text{ for all } J \in \mathcal{T} \\
(\star) & (ii) & \operatorname{sgn}(c_{\mathcal{T}(1)}) = 1 \text{ and } \operatorname{sgn}(c_{\mathcal{T}(t)}) \neq \operatorname{sgn}(c_{\mathcal{T}(t+1)}) \text{ for all } t = 1, ..., |\mathcal{T}| - 1 \\
 & (iii) & \operatorname{sgn}(c_{\mathcal{T}(|\mathcal{T}|)}) = (-1)^S
\end{array}
$$

Recall that for $|\mathcal{T}| \leq S$, an optimal integer solution of (OPT-$\mathrm{P}_0$) is trivially given by $x_{\mathcal{T}(t)} = 1$ if and only if $c_{\mathcal{T}(t)} \geq 0$. However, if an instance has the properties $(\star)$, then also the case $|\mathcal{T}| = S + 1$ will turn out to be fairly easy to solve. Motivated by this, Algorithm 2 repeatedly decreases $|\mathcal{T}|$ as long as $|\mathcal{T}| > S + 1$ (lines 4 to 6 in Algorithm 2) by applying the *Merging Rule* defined below. The latter reduces an instance $(c, \mathcal{T})$ of Problem (OPT-$\mathrm{P}_0$) with dimension $|\mathcal{T}|$ to an instance $(c', \mathcal{T}')$ with dimension $|\mathcal{T}'| = |\mathcal{T}| - 2$ while maintaining the properties $(\star)$.

**Definition 3.3.4** (Merging Rule)**.** Given a vector $c$ with properties $(\star)$ and an index set $\mathcal{T}$, let $\mathcal{T}(\theta)$ be an element of $\mathcal{T}$ such that $|c_{\mathcal{T}(\theta)}|$ is minimal. Now, define a merged instance with objective $c'$ and index set $\mathcal{T}'$, with $|\mathcal{T}'| = |\mathcal{T}| - 2$, as follows:

(a) if $\theta = 1$, set $\mathcal{T}' := \mathcal{T} \setminus \{\mathcal{T}(1), \mathcal{T}(2)\}$ and $\bar{c}_J := c_J$ for $J \in \mathcal{T}'$

(b) if $\theta = |\mathcal{T}|$, set $\mathcal{T}' := \mathcal{T} \setminus \{\mathcal{T}(|\mathcal{T}| - 1), \mathcal{T}(|\mathcal{T}|)\}$ and $c'_J := c_J$ for $J \in \mathcal{T}'$

(c) otherwise, define

$$
\begin{aligned}
&\mathcal{T}' := \mathcal{T} \setminus \{\mathcal{T}(\theta - 1), \mathcal{T}(\theta), \mathcal{T}(\theta + 1)\} \cup \{\mathcal{T}(\theta - 1) \cup \mathcal{T}(\theta) \cup \mathcal{T}(\theta + 1)\}\ , \\
&c'_J := c_J \text{ for } J \in \mathcal{T}' \cap \mathcal{T} \qquad \text{and} \\
&c'_{\mathcal{T}(\theta-1)\cup\mathcal{T}(\theta)\cup\mathcal{T}(\theta+1)} := c_{\mathcal{T}(\theta-1)} + c_{\mathcal{T}(\theta)} + c_{\mathcal{T}(\theta+1)}\ .
\end{aligned}
$$

In words, the above definition can be phrased as folllows: either remove the first two entries as in (a), or remove the last two entries as in (b), or merge three entries to one as in (c), adding up their coefficients. By its definition, it is obvious that the Merging Rule maintains the properties $(\star)$. Below, Example 3.3.5 is continued with the application of the Merging Rule.

**Example 3.3.6.** Let $S = 3$ and consider the instance

$$
\begin{aligned}
c^{(4)} &= (4, -6, 5, -7, 5, -2, 3, -9, 11, -1), \\
\mathcal{T}^{(4)} &= \{\{2, 3\}, \{4\}, \{5\}, \{7\}, \{8\}, \{9\}, \{10\}, \{11\}, \{12\}, \{13\}\}
\end{aligned}
$$

which was the final instance obtained by the preprocessing in Example 3.3.5. Recall that this instance therefore satisfies the properties ($\star$). Now, application of the Merging Rule yields $\theta = 10$ with $\mathcal{T}(\theta) = \{13\}$ and $|c_{\mathcal{T}(\theta)}| = 1$. As a result of applying (b), one obtains

$$\begin{aligned} c' &= (4, -6, 5, -7, 5, -2, 3, -9), \\ \mathcal{T}' &= \{\{2,3\}, \{4\}, \{5\}, \{7\}, \{8\}, \{9\}, \{10\}, \{11\}\} . \end{aligned}$$

A second application of the Merging Rule as in (c) yields $\theta = 6$ with $\mathcal{T}'(\theta) = \{9\}$, $|c_{T'(\theta)}| = 2$ and

$$\begin{aligned} c'' &= (4, -6, 5, -7, 6, -9), \\ \mathcal{T}'' &= \{\{2,3\}, \{4\}, \{5\}, \{7\}, \{8,9,10\}, \{11\}\} . \end{aligned}$$

Finally, with operation (a), one obtains $\theta = 1$ with $\mathcal{T}''(\theta) = \{2,3\}, |c_{\mathcal{T}''(\theta)}| = 4$ and

$$\begin{aligned} c''' &= (5, -7, 6, -9), \\ \mathcal{T}''' &= \{\{5\}, \{7\}, \{8,9,10\}, \{11\}\} . \end{aligned}$$

The effect of the Merging Rule can be interpreted in the primal-dual context as follows: deleting two entries at the beginning of $c$ as in (a) (or at the end as in (b), respectively) corresponds to deleting the first (or last) two corresponding rows from the dual problem, whereas merging three entries of $c$ into a single one as in (c) can be interpreted as summing up the corresponding three dual constraints. In the example above, the final instance $(c''', \mathcal{T}''')$ satisfies $|\mathcal{T}'''| = S + 1$ and as such, represents the root of the recursion in Algorithm 2 (lines 2 and 3). Such an instance can be solved quickly:

**Lemma 3.3.9** (Recursion Root)**.** Consider an instance $(c, \mathcal{T})$ with $|\mathcal{T}| = S + 1$ such that the objective $c$ has properties ($\star$). Let $\theta \in \arg\min_t |c_{\mathcal{T}(t)}|$. Then, the solution

$$x^*_{\mathcal{T}(t)} := \begin{cases} 1 & \text{if } c_{\mathcal{T}(t)} > 0 \text{ and } t \neq \theta \\ 0 & \text{if } c_{\mathcal{T}(t)} < 0 \text{ and } t \neq \theta \end{cases} \qquad x^*_{\mathcal{T}(\theta)} := \begin{cases} 1 & \text{if } c_{\mathcal{T}(\theta)} < 0 \\ 0 & \text{if } c_{\mathcal{T}(\theta)} > 0 \end{cases}$$

is an optimizer of the primal problem, while the solution

$$y^*_{\mathcal{T}} := |c_{\mathcal{T}(\theta)}|, \quad z^*_{\mathcal{T}(t)} := \begin{cases} c_{\mathcal{T}(t)} - y^*_{\mathcal{T}} & \text{if } c_{\mathcal{T}(t)} > 0 \text{ and } t \neq \theta \\ 0 & \text{otherwise} \end{cases}$$

is optimal for the dual problem.

*Proof.* Because of $|\mathcal{T}| = S + 1$, the primal problem contains only one alternating inequality, namely the inequality determined by $\mathcal{T}$. Consequently, the dual problem

contains only a single variable $y_{\mathcal{T}}$ and the $|\mathcal{T}| = S + 1$ variables $z_{\mathcal{T}(t)}, t = 1, ..., |\mathcal{T}|$. The solution $x^*$ has a variation of at most $S$ by its definition and is hence feasible for the primal problem. The feasibility of $y^*, z^*$ is just as easily checked. For the objective values, it holds

$$\begin{aligned} c^\top x^* = \sum_{c_{\mathcal{T}(t)}>0} c_{\mathcal{T}(t)} - |c_{\mathcal{T}(\theta)}| &= \sum_{c_{\mathcal{T}(t)}>0} c_{\mathcal{T}(t)} - y^*_{\mathcal{T}} \\ &= \lfloor \tfrac{S}{2} \rfloor \cdot y^*_{\mathcal{T}} + \sum_{c_{\mathcal{T}(t)}>0} c_{\mathcal{T}(t)} - \Big( \lfloor \tfrac{S}{2} \rfloor + 1 \Big) \cdot y^*_{\mathcal{T}} \\ &= \lfloor \tfrac{S}{2} \rfloor \cdot y^*_{\mathcal{T}} + \sum_{c_{\mathcal{T}(t)}>0} c_{\mathcal{T}(t)} - \#\Big\{ t \mid c_{\mathcal{T}(t)} > 0 \Big\} \cdot y^*_{\mathcal{T}} \\ &= \lfloor \tfrac{S}{2} \rfloor \cdot y^*_{\mathcal{T}} + \sum_{c_{\mathcal{T}(t)}>0} \Big( c_{\mathcal{T}(t)} - y^*_{\mathcal{T}} \Big) \\ &= \lfloor \tfrac{S}{2} \rfloor \cdot y^*_{\mathcal{T}} + \sum_{c_{\mathcal{T}(t)}>0} z^*_{\mathcal{T}(t)} \, . \end{aligned}$$

Thus, the optimality of the solutions follows from weak duality. □

Since the final instance $(c''', \mathcal{T}''')$ from Example 3.3.6 satisfies all conditions for the previous Lemma 3.3.9, it is possible to compute an optimal primal and dual solution for it.

**Example 3.3.7.** Consider the instance

$$\begin{aligned} c''' &= (5, -7, 6, -9), \\ \mathcal{T}''' &= \{\{5\}, \{7\}, \{8, 9, 10\}, \{11\}\} \end{aligned}$$

with $S = 3$ from Example 3.3.6. According to Lemma 3.3.9, $\theta = 1$ and an optimal solution is given by

$$x''' = (0, 0, 1, 0) \text{ with optimal value } 6 \, .$$

The corresponding dual optimal solution is given by $y'''_{\mathcal{T}'''} = 5$, and $z'''_{\{8,9,10\}} = 1$, and $z'''_J = 0$ otherwise, and it achieves an objective value $\lfloor 3/2 \rfloor \cdot 5 + 1 = 6$. The sketch below shows its feasibility by evaluating the (non-trivial) dual constraints; zero-values are omitted for a better readability. It also depicts the primal solution $x'''$ and the elements of the index set $\mathcal{T}'''$ on the very right for the sake of completeness.

| $y'''_{\mathcal{T}'''}$ | $z'''_{\{8,9,10\}}$ | $c'''$ | $x'''$ | $J \in \mathcal{T}'''$ |
|---|---|---|---|---|
| $+5$ | | $\geq +5$ | 0 | $\{5\}$ |
| $-5$ | | $\geq -7$ | 0 | $\{7\}$ |
| $+5$ | $+1$ | $\geq +6$ | 1 | $\{8, 9, 10\}$ |
| $-5$ | | $\geq -9$ | 0 | $\{11\}$ |

Lemma 3.3.9 serves as the root for the recursive application of the following Theorem 3.3.5, which is the core of the merging algorithm. It specifies how to reverse the reduction process in a step-wise manner by constructing a primal integer optimum solution and a corresponding dual optimum solution for each problem instance arising in the course of the reduction. This reversal process will exploit a characteristic of the dual optimum solution that can already be observed in the previous Example 3.3.7: By definition, the dual variable $y'''_{\mathcal{T}'''}$ is assigned the value $|\mathcal{T}'''(\theta)| = 5$. Furthermore, the (non-zero) coefficients for the variables $y'''_{\mathcal{J}}$ that are assigned a positive value have the same sign as the corresponding right hand side $c$ in every constraint of the dual problem. This is a beneficial trait of the dual solution that will be maintained when reversing the merging process.

**Theorem 3.3.5** (Reversing the merging process)**.** Let $c'$ and $\mathcal{T}'$ be obtained from $c$ and $\mathcal{T}$ by applying the Merging Rule. Let $A'$ denote the matrix corresponding to the merged problem and let ${A'}^{\top}_{J,\mathcal{J}}$ refer to the entry in row $J$ and column $\mathcal{J}$ of ${A'}^{\top}$. Assume that, for the merged instance, an integer optimal solution $x'$ for the primal problem is given as well as an optimal solution $(y', z')$ for the dual problem, where $y'$ and $z'$ satisfy the following conditions:

(C1) If $y'_{\mathcal{J}} > 0$ and ${A'}^{\top}_{J,\mathcal{J}} \neq 0$, then $\operatorname{sgn}({A'}^{\top}_{J,\mathcal{J}}) = \operatorname{sgn}(c'_J)$ for all $J \in \mathcal{T}'$.

(C2) The value of $y'_{\mathcal{T}'}$ equals $\min_{J\in\mathcal{T}'} |c'_J|$ .

Then, for the original instance $(c, \mathcal{T})$, there exist an integer optimal primal solution $x$ and an optimal dual solution $(y, z)$ which satisfies (C1) and (C2) again.

*Proof.* The construction of $x$ and $(y, z)$ depends on how the merged instance $(c', \mathcal{T}')$ is obtained from the instance $(c, \mathcal{T})$. Assume first that $\theta = 1$ as in case (a). Then, define a dual solution $(y, z)$ for the original instance by setting

$$y_{\mathcal{T}} := |c_{\mathcal{T}(\theta)}|, \quad y_{\mathcal{T}'} := y'_{\mathcal{T}'} - |c_{\mathcal{T}(\theta)}|, \quad y_{\mathcal{J}} := y'_{\mathcal{J}} \text{ for } \mathcal{J} \subseteq \mathcal{T}' \cap \mathcal{T}, \quad y_{\mathcal{J}} := 0 \text{ otherwise}$$

and

$$z_J := z'_J \text{ for } J \in \mathcal{T}' \cap \mathcal{T}, \quad z_{\mathcal{T}(1)} := z_{\mathcal{T}(2)} := 0 \, .$$

Because $(y', z')$ satisfies (C2) and by the Merging Rule it follows

$$|c_{\mathcal{T}(\theta)}| = \min_{J\in\mathcal{T}} |c_J| \leq \min_{J\in\mathcal{T}'} |c'_J| = y'_{\mathcal{T}'} \quad \Rightarrow \quad y_{\mathcal{T}'} \geq 0 \, .$$

So obviously, $(y, z) \geq 0$. By construction of $(y, z)$, all dual constraints $\mathcal{T}(t)$, $t \geq 3$, in the original instance are satisfied. One only has to make sure that the defined dual solution satisfies the constraints $\mathcal{T}(1)$ and $\mathcal{T}(2)$ as well. This is achieved by setting $y_{\mathcal{T}} := |c_{\mathcal{T}(\theta)}|$ and because of the fact that $\min_{J\in\mathcal{T}} |c_J| = c_{\mathcal{T}(1)}$. Thus, the

constructed solution is a feasible dual solution for the original instance. By *shifting* the value $|c_{\mathcal{T}(\theta)}|$ from $y_{\mathcal{T}'}$ to $y_{\mathcal{T}}$, it is ensured that $(y, z)$ satisfies (C2). Because $(y', z')$ satisfies (C1) and by definition of $y$, $(y, z)$ satisfies (C1) as well. By construction, it holds $\sum_{J\in\mathcal{T}'} {y'}_J = \sum_{J\in\mathcal{T}} y_J$ and $\sum_{J\in\mathcal{T}'} z'_J = \sum_{J\in\mathcal{T}} z_J$, the objective value of the constructed dual solution $(y, z)$ is the same as the optimal value ${c'}^\top x'$ of the merged instance. Finally, define $x$ by extending $x'$ by two zeros in the front, i.e., define

$$x_J := x'_J \text{ for } J \in \mathcal{T}' \quad \text{and} \quad x_{\mathcal{T}(1)} = x_{\mathcal{T}(2)} := 0 \,.$$

Then, $x$ is integer and feasible for the original (primal) problem with $c^\top x = {c'}^\top x'$. Therefore, by weak duality, the constructed solutions $x$ and $(y, z)$ are optimal for the non-merged instance.

Now, assume that $(c', \mathcal{T}')$ is obtained by deleting the last two elements of $c$ and $\mathcal{T}$ as in case (b), so $\theta = |\mathcal{T}|$. Then define a feasible dual solution $(y, z)$ and a feasible integer primal solution for the original instance depending on whether $S$ is even or odd. Note that this is the only case in which this distinction for $S$ is made. However, the definition of the dual variables $y$ remains independent of the parity of $S$: those variables are defined exactly as in the first case above. For the definition of the dual variables $z$ and the integer primal solution $x$, distinguish the following cases:

– If $S$ is even, define $z$ and $x$ via

$$z_J := z'_J \text{ for } J \in \mathcal{T}' \cap \mathcal{T}, \quad z_{\mathcal{T}(|\mathcal{T}|-1)} = z_{\mathcal{T}(|\mathcal{T}|)} := 0$$

and

$$x_J := x'_J \text{ for } J \in \mathcal{T}', \quad x_{\mathcal{T}(|\mathcal{T}|-1)} = x_{\mathcal{T}(|\mathcal{T}|)} := 0 \,.$$

As the vector $x$ is obtained by extending $x'$ by two zeros at the end, it is integer and feasible for the (primal) original problem, since $x'$ contains at most $S$ switchings in case it ends in zero and at most $S-1$ otherwise. By arguing as above, the dual solution $(y, z)$ is feasible for the original instance, it satisfies conditions (C1) and (C2), and

$$c^\top x = {c'}^\top x' = \lfloor \tfrac{S}{2} \rfloor \sum_{\mathcal{J}\subseteq\mathcal{T}'} y'_{\mathcal{J}} + \sum_{J\in\mathcal{T}'} z'_J = \lfloor \tfrac{S}{2} \rfloor \sum_{\mathcal{J}\subseteq\mathcal{T}} y_{\mathcal{J}} + \sum_{J\in\mathcal{T}} z_J \,.$$

Consequently, weak duality implies that $x$ and $(y, z)$ are optimal.

– If $S$ is odd, define $z$ and $x$ via

$$z_J := z'_J \text{ for } J \in \mathcal{T}' \cap \mathcal{T}, \quad z_{\mathcal{T}(|\mathcal{T}|-1)} := c_{\mathcal{T}(|\mathcal{T}|-1)} + c_{\mathcal{T}(|\mathcal{T}|)}, \quad z_{\mathcal{T}(|\mathcal{T}|)} := 0$$

and

$$x_J := x'_J \text{ for } J \in \mathcal{T}', \quad x_{\mathcal{T}(|\mathcal{T}|-1)} = x_{\mathcal{T}(|\mathcal{T}|)} := 1 \,.$$

It is clear by definition of $y$ and $z$ that all constraints of the dual problem for the non-merged instance other than $\mathcal{T}(|\mathcal{T}|-1)$ and $\mathcal{T}(|\mathcal{T}|)$ are satisfied. Since $c$ satisfies the properties $(\star)$ and $S$ is odd, it is $c_{\mathcal{T}(|\mathcal{T}|-1)} > 0$ and $c_{\mathcal{T}(|\mathcal{T}|)} < 0$. Furthermore, if $(c', \mathcal{T}')$ was obtained by deleting the last two entries in $c$ and $\mathcal{T}$, i.e., if $\theta = |\mathcal{T}|$, then $|c_{\mathcal{T}(|\mathcal{T}|)}| = |c_{\mathcal{T}(\theta)}| \leq |c_{\mathcal{T}(|\mathcal{T}|-1)}|$, which implies $c_{\mathcal{T}(|\mathcal{T}|-1)} + c_{\mathcal{T}(|\mathcal{T}|)} \geq 0$, so that $z_{\mathcal{T}(|\mathcal{T}|-1)} \geq 0$. Thus, it is clear that $(y, z)$ defined as above satisfies all dual constraints of the original instance. The vector $x$ is integer and feasible for the (primal) original problem, since $x'$ contains at most $S$ switchings in case it ends in one and at most $S-1$ otherwise. Concerning objective function values, it follows that

$$\begin{aligned} c^\top x &= {c'}^\top x' + c_{\mathcal{T}(|\mathcal{T}|-1)} + c_{\mathcal{T}(|\mathcal{T}|)} \\ &= \lfloor \tfrac{S}{2} \rfloor \sum_{\mathcal{J} \subseteq \mathcal{T}'} y'_{\mathcal{J}} + \sum_{J \in \mathcal{T}'} z'_J + c_{\mathcal{T}(|\mathcal{T}|-1)} + c_{\mathcal{T}(|\mathcal{T}|)} \\ &= \lfloor \tfrac{S}{2} \rfloor \sum_{\mathcal{J} \subseteq \mathcal{T}'} y'_{\mathcal{J}} + \sum_{J \in \mathcal{T}'} z'_J + z_{\mathcal{T}(|\mathcal{T}|-1)} + z_{\mathcal{T}(|\mathcal{T}|)} \\ &= \lfloor \tfrac{S}{2} \rfloor \sum_{\mathcal{J} \subseteq \mathcal{T}} y_{\mathcal{J}} + \sum_{J \in \mathcal{T}} z_J \ . \end{aligned}$$

Consequently, weak duality again shows optimality of $x$ and $(y, z)$. Observe that the latter satisfies conditions (C1) and (C2) for the same reasons as in case (a).

It remains to consider the case in which $(c', \mathcal{T}')$ is obtained by applying the Merging Rule with $\theta \in \{2, ..., |\mathcal{T}| - 1\}$, as in (c). Consider the constraint $M := \mathcal{T}(\theta-1) \cup \mathcal{T}(\theta) \cup \mathcal{T}(\theta+1)$. Recall that one can understand the merging process in this case as the summation of the corresponding three dual constraints $\mathcal{T}(\theta-1)$, $\mathcal{T}(\theta)$, and $\mathcal{T}(\theta+1)$ in the dual problem. This summation shall be undone in such a way that a dual solution $(y, z)$ for the original instance with the same objective function value as $(y', z')$ is obtained. First, define

$$y_{\mathcal{J}} := y'_{\mathcal{J}} \quad \text{for } \mathcal{J} \subseteq \mathcal{T} \cap \mathcal{T}' \quad \text{and} \quad z_J := z'_J \quad \text{for } J \in \mathcal{T} \cap \mathcal{T}' \ .$$

Consider $y'_{\mathcal{J}}$ with $\mathcal{J} \in \mathcal{K} := \{\mathcal{J} \subseteq \mathcal{T}' \mid y'_{\mathcal{J}} > 0,\ M \in \mathcal{J}\}$. For all $\mathcal{J} \in \mathcal{K}$, distribute the value $y'_{\mathcal{J}}$ among the three variables $y_{\mathcal{J}\setminus\{M\}\cup L}$, $y_{\mathcal{J}\setminus\{M\}\cup\{\mathcal{T}(\theta-1)\}}$, and $y_{\mathcal{J}\setminus\{M\}\cup\{\mathcal{T}(\theta+1)\}}$, where $L := \{\mathcal{T}(\theta-1), \mathcal{T}(\theta), \mathcal{T}(\theta+1)\}$, and assign values to the variables $z_{\mathcal{T}(\theta-1)}, z_{\mathcal{T}(\theta)}$, and $z_{\mathcal{T}(\theta+1)}$ according to the following procedure:

1. Assign values to the variables $y_{\mathcal{J}\setminus\{M\}\cup L}$, $\mathcal{J} \in \mathcal{K}$, such that

$$\sum_{\mathcal{J} \in \mathcal{K}} y_{\mathcal{J}\setminus\{M\}\cup L} = |c_{\mathcal{T}(\theta)}|, \quad y_{\mathcal{J}\setminus\{M\}\cup L} \in [0, y'_{\mathcal{J}}] \text{ for } \mathcal{J} \in \mathcal{K} \ .$$

This is possible because the Merging Rule and (C2) imply $y'_{\mathcal{T}'} = \min_t |c'_{T'(t)}| \geq |c_{\mathcal{T}(\theta)}|$, also obviously $M \in \mathcal{T}'$ holds and, therefore, $\mathcal{T}' \in \mathcal{K}$. Furthermore,

there is such an assignment with $y_{\mathcal{T}} = |c_{\mathcal{T}(\theta)}|$. Next, set $z_{\mathcal{T}(\theta)} := 0$ so that the constraint $\mathcal{T}(\theta)$ in the original instance is satisfied with equality.

2. Define $\tilde{c}_{\mathcal{T}(\theta-1)} := |c_{\mathcal{T}(\theta-1)}| - |c_{\mathcal{T}(\theta)}|$ and $\tilde{c}_{\mathcal{T}(\theta+1)} := |c_{\mathcal{T}(\theta+1)}| - |c_{\mathcal{T}(\theta)}|$ for a better overview and assign values to the remaining variables identified above by setting

$$y_{\mathcal{J}\setminus\{M\}\cup\{\mathcal{T}(\theta-1)\}} := (y'_{\mathcal{J}} - y_{\mathcal{J}\setminus\{M\}\cup L}) \cdot \tfrac{\tilde{c}_{\mathcal{T}(\theta-1)}}{\tilde{c}_{\mathcal{T}(\theta-1)}+\tilde{c}_{\mathcal{T}(\theta+1)}} ,$$

$$y_{\mathcal{J}\setminus\{M\}\cup\{\mathcal{T}(\theta+1)\}} := (\bar{y}_{\mathcal{J}} - y_{\mathcal{J}\setminus\{M\}\cup L}) \cdot \tfrac{\tilde{c}_{\mathcal{T}(\theta+1)}}{\tilde{c}_{\mathcal{T}(\theta-1)}+\tilde{c}_{\mathcal{T}(\theta+1)}} ,$$

for all $\mathcal{J} \in \mathcal{K}$, and

$$z_{\mathcal{T}(\theta-1)} := z'_M \cdot \tfrac{\tilde{c}_{\mathcal{T}(\theta-1)}}{\tilde{c}_{\mathcal{T}(\theta-1)}+\tilde{c}_{\mathcal{T}(\theta+1)}}, \quad z_{\mathcal{T}(\theta+1)} := z'_M \cdot \tfrac{\tilde{c}_{\mathcal{T}(\theta+1)}}{\tilde{c}_{\mathcal{T}(\theta-1)}+\tilde{c}_{\mathcal{T}(\theta+1)}} .$$

Then the constraints $\mathcal{T}(\theta-1)$ and $\mathcal{T}(\theta+1)$ in the original instance are satisfied, too.

All other variables $y_{\mathcal{J}}$ are set to zero. With this assignment, it is not difficult to see that the constructed solution $(y, z)$ satisfies conditions (C1) and (C2) again. Moreover,

$$\begin{aligned} y_{\mathcal{J}\setminus\{M\}\cup L} + y_{\mathcal{J}\setminus\{M\}\cup\{\mathcal{T}(\theta-1)\}} + y_{\mathcal{J}\setminus\{M\}\cup\{\mathcal{T}(\theta+1)\}} &= y'_{\mathcal{J}} \quad \text{for } \mathcal{J} \in \mathcal{K} , \\ z_{\mathcal{T}(\theta-1)} + z_{\mathcal{T}(\theta)} + z_{\mathcal{T}(\theta+1)} &= z'_M . \end{aligned}$$

The first equation implies that all other constraints $\mathcal{T}(t)$ with $t \notin \{\theta-1, \theta, \theta+1\}$ in the original instance are satisfied as well. Furthermore, by these two equations it holds $\sum_{\mathcal{J}\subseteq\mathcal{T}'} y'_{\mathcal{J}} = \sum_{\mathcal{J}\subseteq\mathcal{T}} y_{\mathcal{J}}$ and $\sum_{J\in\mathcal{T}'} z'_J = \sum_{J\in\mathcal{T}} z_J$. By looking at the coefficients in the objective function of the dual problem it is clear that the feasible solution $(y, z)$ for the original instance has the same objective function value as the solution $(y', z')$. Finally, define an integer primal solution for the original instance by

$$x_J := x'_J \text{ for } J \in \mathcal{T} \setminus L, \quad x_J := x'_M \text{ for } J \in L .$$

The feasibility of $x$ is obvious since it has the same variation as $x'$, and $x$ has the same objective function value as $x'$ since

$$c^\top x = \sum_{J\in\mathcal{T}\setminus L} c_J x_J + \sum_{J\in L} c_J x_J = \sum_{J\in\mathcal{T}\setminus L} c_J x_J + x'_M \underbrace{\sum_{J\in L} c_J}_{=c'_M} = c'^\top x'$$

The optimality of $x$ and $(y, z)$ then follows from weak duality again. □

The construction in the proof of Theorem 3.3.5 is illustrated by continuing the example from before and the merging process according to the previous proof.

**Example 3.3.8.** Recall that the optimal integer primal and dual solutions for the recursion root instance $(c''', \mathcal{T}''')$ were obtained in Example 3.3.7 and they yield an optimal value of 6. The corresponding solutions are once again depicted below in the same style as in Example 3.3.7, i.e., by evaluating the (non-trivial) dual constraints, omitting zero-valued dual variables and by writing the primal solution and the index set on the very right.

| $y'''_{\mathcal{T}'''}$ | $z'''_{\{8,9,10\}}$ | $c'''$ | $x'''$ | $J \in \mathcal{T}'''$ |
|---|---|---|---|---|
| $+5$ | | $\geq +5$ | 0 | $\{5\}$ |
| $-5$ | | $\geq -7$ | 0 | $\{7\}$ |
| $+5$ | $+1$ | $\geq +6$ | 1 | $\{8,9,10\}$ |
| $-5$ | | $\geq -9$ | 0 | $\{11\}$ |

Further recall that the instance $(c''', \mathcal{T}''')$ was obtained with merging operation (a) from the instance

$$\begin{aligned} c'' &= (4, -6, 5, -7, 6, -9), \\ \mathcal{T}'' &= \{\{2,3\}, \{4\}, \{5\}, \{7\}, \{8,9,10\}, \{11\}\}\ , \end{aligned}$$

where $\theta = 1$ with $\mathcal{T}''(\theta) = \{2,3\}$ and $|c''_{\mathcal{T}''}(\theta)| = 4$.
By following the construction described in the previous proof, the integer primal and dual solution for $(c'', \mathcal{T}'')$ are sketched continuing in the same style.

| $y''_{\mathcal{T}''}$ | $y''_{\mathcal{T}'''}$ | $z''_{\{8,9,10\}}$ | $c''$ | $x''$ | $J \in \mathcal{T}''$ |
|---|---|---|---|---|---|
| $+4$ | | | $\geq +4$ | 0 | $\{2,3\}$ |
| $-4$ | | | $\geq -6$ | 0 | $\{4\}$ |
| $+4$ | $+1$ | | $\geq +5$ | 0 | $\{5\}$ |
| $-4$ | $-1$ | | $\geq -7$ | 0 | $\{7\}$ |
| $+4$ | $+1$ | $+1$ | $\geq +6$ | 1 | $\{8,9,10\}$ |
| $-4$ | $-1$ | | $\geq -9$ | 0 | $\{11\}$ |

The optimal value is given by $\lfloor \frac{3}{2} \rfloor \cdot (4+1) + 1 = 6 \cdot 1$.
Recall that the instance $(c'', \mathcal{T}'')$, in turn, was derived through by application of (c) in the Merging Rule to the following instance

$$\begin{aligned} c' &= (4, -6, 5, -7, 5, -2, 3, -9), \\ \mathcal{T}' &= \{\{2,3\}, \{4\}, \{5\}, \{7\}, \{8\}, \{9\}, \{10\}, \{11\}\}\ , \end{aligned}$$

where $\theta = 6$ with $\mathcal{T}'(\theta) = \{9\}$ and $|c'_{\mathcal{T}'(\theta)}| = 2$.
The optimal primal and dual solutions for the instance $(c', \mathcal{T}')$ that result from the construction described in the previous proof are sketched below still using the same style. However, with respect to readability, not every variable $y'$ is referenced directly at the top.

| $y'_{\mathcal{T}'}$ | | | | | | $z'_{\{8\}}$ | $z'_{\{10\}}$ | $c'$ | $x'$ | $J \in \mathcal{T}'$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $+2$ | $+\frac{6}{4}$ | $+\frac{2}{4}$ | | | | | | $\geq +4$ | 0 | $\{2,3\}$ |
| $-2$ | $-\frac{6}{4}$ | $-\frac{2}{4}$ | | | | | | $\geq -6$ | 0 | $\{4\}$ |
| $+2$ | $+\frac{6}{4}$ | $+\frac{2}{4}$ | $+0$ | $+\frac{3}{4}$ | $+\frac{1}{4}$ | | | $\geq +5$ | 0 | $\{5\}$ |
| $-2$ | $-\frac{6}{4}$ | $-\frac{2}{4}$ | $-0$ | $-\frac{3}{4}$ | $-\frac{1}{4}$ | | | $\geq -7$ | 0 | $\{7\}$ |
| $+2$ | $+\frac{6}{4}$ | | $+0$ | $+\frac{3}{4}$ | | $+\frac{3}{4}$ | | $\geq +5$ | 1 | $\{8\}$ |
| $-2$ | | | $-0$ | | | | | $\geq -2$ | 1 | $\{9\}$ |
| $+2$ | | $+\frac{2}{4}$ | $+0$ | | $+\frac{1}{4}$ | | $+\frac{1}{4}$ | $\geq +3$ | 1 | $\{10\}$ |
| $-2$ | $-\frac{6}{4}$ | $-\frac{2}{4}$ | $-0$ | $-\frac{3}{4}$ | $-\frac{1}{4}$ | | | $\geq -9$ | 0 | $\{11\}$ |

The objective value is $\lfloor \frac{3}{2} \rfloor (2 + \frac{6}{4} + \frac{2}{4} + \frac{3}{4} + \frac{1}{4}) + \frac{3}{4} + \frac{1}{4} = 6 \cdot 1$.
Finally, the instance $(c', \mathcal{T}')$ was obtained by applying the Merging Rule as in (b) with $\theta = 10$ with $\mathcal{T}(\theta) = \{13\}$ and $|c_{\mathcal{T}(\theta)}| = 1$ to the following instance

$$\begin{aligned} c &= (4, -6, 5, -7, 5, -2, 3, -9, 11, -1), \\ \mathcal{T} &= \{\{2,3\}, \{4\}, \{5\}, \{7\}, \{8\}, \{9\}, \{10\}, \{11\}, \{12\}, \{13\}\} . \end{aligned}$$

According the previous proof, the primal solution $x'$ is extended at the end with two one-entries since $S = 3$ is odd. The resulting primal and dual solution for the instance $(c, \mathcal{T})$ are sketched below.

| $y_{\mathcal{T}}$ | $y_{\mathcal{T}'}$ | | | | | | $z_{\{8\}}$ | $z_{\{10\}}$ | $z_{\{13\}}$ | | $x$ | $J \in \mathcal{T}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $+1$ | $+1$ | $+\frac{6}{4}$ | $+\frac{2}{4}$ | | | | | | | $\geq +4$ | 0 | $\{2,3\}$ |
| $-1$ | $-1$ | $-\frac{6}{4}$ | $-\frac{2}{4}$ | | | | | | | $\geq -6$ | 0 | $\{4\}$ |
| $+1$ | $+1$ | $+\frac{6}{4}$ | $+\frac{2}{4}$ | $+0$ | $+\frac{3}{4}$ | $+\frac{1}{4}$ | | | | $\geq +5$ | 0 | $\{5\}$ |
| $-1$ | $-1$ | $-\frac{6}{4}$ | $-\frac{2}{4}$ | $-0$ | $-\frac{3}{4}$ | $-\frac{1}{4}$ | | | | $\geq -7$ | 0 | $\{7\}$ |
| $+1$ | $+1$ | $+\frac{6}{4}$ | | $+0$ | $+\frac{3}{4}$ | | $+\frac{3}{4}$ | | | $\geq +5$ | 1 | $\{8\}$ |
| $-1$ | $-1$ | | | $-0$ | | | | | | $\geq -2$ | 1 | $\{9\}$ |
| $+1$ | $+1$ | | $+\frac{2}{4}$ | $+0$ | | $+\frac{1}{4}$ | | $+\frac{1}{4}$ | | $\geq +3$ | 1 | $\{10\}$ |
| $-1$ | $-1$ | $-\frac{6}{4}$ | $-\frac{2}{4}$ | $-0$ | $-\frac{3}{4}$ | $-\frac{1}{4}$ | | | | $\geq -9$ | 0 | $\{11\}$ |
| $+1$ | | | | | | | | | $+10$ | $\geq +11$ | 1 | $\{12\}$ |
| $-1$ | | | | | | | | | | $\geq -1$ | 1 | $\{13\}$ |

The optimal value is given by

$$\lfloor \tfrac{3}{2} \rfloor (1 + 1 + \tfrac{6}{4} + \tfrac{2}{4} + \tfrac{3}{4} + \tfrac{1}{4}) + \tfrac{3}{4} + \tfrac{1}{4} + 10 = 16 = 5 - 2 + 3 + 11 - 1 .$$

Currently, Theorem 3.3.5 and Lemma 3.3.9 prove that Algorithm 2 returns an integer optimum solution for (OPT-$P_0$) in case the objective $c$ has properties ($\star$) or, in other words, when the preprocessing step is superfluous. Finally, it is shown that this also holds true for an arbitrary objective, which will conclude the proof of integrality for the polytope $P_0$ and show the correctness of Algorithm 2.

**Theorem 3.3.6.** The primal-dual optimization Algorithm 2 computes an optimal integer (primal) solution for (OPT-P$_0$). In particular, the polytope $P_0$ is integer and hence agrees with $\mathrm{conv}(\Omega_0)$ defined in Theorem 3.3.3.

*Proof.* Let $c$ be an arbitrary objective. By performing the preprocessing operations described in Definition 3.3.3, the instance $(c^{(4)}, \mathcal{T}^{(4)})$ results from $(c, \mathcal{T})$. The objective $c^{(4)}$ has the properties $(\star)$, and by Theorem 3.3.5 and Lemma 3.3.9 let there be an optimum integer primal solution $x^{(4)}$ and an optimum dual solution $(y^{(4)}, z^{(4)})$. Define a primal solution $x^{(3)}$ and dual solution $(y^{(3)}, z^{(3)})$ for $(\mathcal{T}^{(3)}, c^{(3)})$ as follows:

$$x_J^{(3)} := x_J^{(4)}, \quad z_J^{(3)} := z_J^{(4)} \quad \text{for } J \in \mathcal{T}^{(3)} \cap \mathcal{T}^{(4)}$$

and, for $J \in \mathcal{T}^{(3)} \setminus \mathcal{T}^{(4)}$, define depending on the parity of $S$

$$\begin{aligned} x_J^{(3)} &:= 0 \quad \text{and} \quad z_J^{(3)} := 0 \quad \text{if } S \text{ is even, and} \\ x_J^{(3)} &:= 1 \quad \text{and} \quad z_J^{(3)} := c_J^{(3)} \quad \text{if } S \text{ is odd.} \end{aligned}$$

The feasibility of $x^{(3)}$ is easy to see. The definition of $y^{(3)}$ is independent of $S$ and its feasibility follows from the definition:

$$y_{\mathcal{J}}^{(3)} := y_{\mathcal{J}}^{(4)} \quad \text{for } \mathcal{J} \subseteq \mathcal{T}^{(3)} \cap \mathcal{T}^{(4)} \quad \text{and} \quad y_{\mathcal{J}}^{(3)} := 0 \text{ otherwise.}$$

By this, it is ${c^{(3)}}^\top x^{(3)} = \lfloor \frac{S}{2} \rfloor \sum_{\mathcal{J} \subseteq \mathcal{T}^{(3)}} y_{\mathcal{J}}^{(3)} + \sum_{J \in \mathcal{T}^{(3)}} z_J^{(3)}$ and the defined solutions are primal and dual optimum solutions for $(\mathcal{T}^{(3)}, c^{(3)})$ by weak duality.

Next, given an optimum integer primal solution $x^{(3)}$ and an optimum dual solution $(y^{(3)}, z^{(3)})$, define an integer solution $x^{(2)}$ and dual solution $(y^{(2)}, z^{(2)})$ for the instance $(I^{(2)}, c^{(2)})$. For this, consider $\mathcal{K} := \{\mathcal{J} \subseteq \mathcal{T}^{(3)} \mid y_{\mathcal{J}}^{(3)} > 0, M \in \mathcal{J}\}$, i.e., the set of index sets $\mathcal{J}$ whose variables $y_{\mathcal{J}}^{(3)}$ go into the dual constraint $M$ with positive value. Then, define $L := \bigcup_{t \le k \le j} \{\mathcal{T}^{(3)}(k)\}$ and

$$y^{(2)}_{\mathcal{J}\setminus\{M\}\cup\{J\}} := y_{\mathcal{J}}^{(3)} \cdot \tfrac{|c_J^{(2)}|}{|c_M^{(3)}|} \quad \text{for } \mathcal{J} \in \mathcal{K}, J \in L \quad \text{and} \quad z_J^{(2)} = z_M^{(3)} \cdot \tfrac{|c_J^{(2)}|}{|c_M^{(3)}|} \text{ for } J \in L\,,$$

and define all other dual variables by

$$y_{\mathcal{J}}^{(2)} := \begin{cases} y_{\mathcal{J}}^{(3)} & \text{for } \mathcal{J} \subseteq \mathcal{T}^{(2)} \cap \mathcal{T}^{(3)} \\ 0 & \text{for } \mathcal{J} \subseteq \mathcal{T}^{(2)} \setminus \mathcal{T}^{(3)} \end{cases} \quad \text{and} \quad z_J^{(2)} := z_J^{(3)} \quad \text{for } J \in \mathcal{T}^{(2)} \cap \mathcal{T}^{(3)}\,.$$

By definition, it holds

$$\sum_{J \in L} y^{(2)}_{\mathcal{J}\setminus\{M\}\cup\{J\}} = \tfrac{y_{\mathcal{J}}^{(3)}}{|c_M^{(3)}|}|c_M^{(3)}| = y_{\mathcal{J}}^{(3)} \quad \text{and} \quad \sum_{J \in L} z_J^{(2)} = \tfrac{z_M^{(3)}}{|c_M^{(3)}|}|c_M^{(3)}| = z_M^{(3)}\,.$$

Thus, it follows $\sum_{\mathcal{J} \subseteq \mathcal{T}^{(2)}} y_{\mathcal{J}}^{(2)} = \sum_{\mathcal{J} \subseteq \mathcal{T}^{(3)}} y_{\mathcal{J}}^{(3)}$ as well as $\sum_{J \in \mathcal{T}^{(2)}} z_J^{(2)} = \sum_{J \in \mathcal{T}^{(3)}} z_J^{(3)}$, so that the objective value of the dual solutions stays the same. For feasibility of

$y^{(2)}, z^{(2)}$ it suffices to show that all dual constraints $J \in L$ are satisfied, as all others are satisfied by definition of the dual solution. So, consider any dual constraint $J \in L$ and let $a_{J,\mathcal{J}}$ denote the coefficient in row $J$ and column $\mathcal{J}$ of the dual constraint matrix for $(c^{(2)}, \mathcal{T}^{(2)})$. Then,

$$\begin{aligned}\sum_{\mathcal{J}\in\mathcal{K}} a_{J,\mathcal{J}} \cdot y^{(2)}_{\mathcal{J}\setminus\{M\}\cup\{J\}} + z^{(2)}_J &= \tfrac{|c^{(2)}_J|}{|c^{(3)}_M|}\underbrace{\Big(\sum_{\mathcal{J}\in\mathcal{K}} a_{J,\mathcal{J}} \cdot y^{(3)}_{\mathcal{J}} + z^{(3)}_M\Big)}_{\geq c^{(3)}_M} \\ &\geq \tfrac{|c^{(2)}_J|}{|c^{(3)}_M|} \cdot c^{(3)}_M = \underbrace{\operatorname{sgn}(c^{(3)}_M)}_{=\operatorname{sgn}(c^{(2)}_J)}|c^{(2)}_J| \,.\end{aligned}$$

Therefore, the constructed solution $(y^{(2)}, z^{(2)})$ is dual feasible for $(c^{(2)}, \mathcal{T}^{(2)})$. Lastly, define a feasible integer primal solution $x^{(2)}$ via

$$x^{(2)}_J := x^{(3)}_M \quad \text{for } J \in L \quad \text{and} \quad x^{(2)}_J := x^{(3)}_J \quad \text{for } J \in \mathcal{T}^{(2)} \cap \mathcal{T}^{(3)} \,.$$

By definition of $c^{(3)}_M$ and $x^{(2)}$, it is clear that ${c^{(2)}}^\top x^{(2)} = {c^{(3)}}^\top x^{(3)}$. Thus $x^{(2)}$ and $(y^{(2)}, z^{(2)})$ are primal and dual optimum solutions for $(c^{(2)}, \mathcal{T}^{(2)})$ by weak duality.

Given an integer optimum solution $x^{(2)}$ and dual solution $(y^{(2)}, z^{(2)})$ for $(c^{(2)}, \mathcal{T}^{(2)})$, now define optimum solutions $x^{(1)}$ and $(y^{(1)}, z^{(1)})$ for $(\mathcal{T}^{(1)}, c^{(1)})$ as

$$x^{(1)}_J := x^{(2)}_J \quad \text{for } J \in \mathcal{T}^{(1)} \cap \mathcal{T}^{(2)} \text{ and } x^{(1)}_J := 0 \text{ otherwise},$$

which is clearly feasible with the same primal objective value as before. Furthermore, define

$$\begin{aligned} y^{(1)}_{\mathcal{J}} &:= y^{(2)}_{\mathcal{J}} \quad \text{for } \mathcal{J} \subseteq \mathcal{T}^{(1)} \cap \mathcal{T}^{(2)} \quad \text{and} \quad y^{(1)}_{\mathcal{J}} := 0 \text{ otherwise} \\ z^{(1)}_J &:= z^{(2)}_J \quad \text{for } J \in \mathcal{T}^{(1)} \cap \mathcal{T}^{(2)} \quad \text{and} \quad z^{(1)}_J := 0 \text{ otherwise.} \end{aligned}$$

The feasibility of the dual solution follows directly from its definition and clearly the dual objective value stays the same. Thus, $x^{(1)}$ and $(y^{(1)}, z^{(1)})$ are primal and dual optimum solutions for $(c^{(1)}, \mathcal{T}^{(1)})$ by weak duality. At last, given an integer optimum solution $x^{(1)}$ and corresponding dual optimum solution $(y^{(1)}, z^{(1)})$, define an integer primal solution $x$ and a feasible dual solution $(y, z)$ for the original instance $(c, \mathcal{T})$. One only has to adapt the dual solution slightly so as to obtain a feasible dual solution for $(c, \mathcal{T})$:

$$y_{\mathcal{J}} := \begin{cases} y^{(1)}_{\mathcal{J}} & \mathcal{J} \subseteq \mathcal{T} \cap \mathcal{T}^{(1)} \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad z_J := \begin{cases} z^{(1)}_J & J \in \mathcal{T} \cap \mathcal{T}^{(1)} \\ 0 & \text{otherwise} \,. \end{cases}$$

Since all constraints $J \in L$ with $L := \{J \in \mathcal{T} \mid c_J = 0\}$ are satisfied with equality, the feasibility follows immediately.

Now, define a feasible integer primal solution for the original instance by copying the closest entry with non-zero coefficient for every component $x_J$ with $c_J = 0$. Clearly, neither the dual objective value nor the primal objective value change, so again weak duality implies the primal and dual optimality of these solutions, thus ending the proof. □

Below, the example from before is concluded by deriving optimal primal and dual solutions for the original instance from Example 3.3.5.

**Example 3.3.9.** Starting with the instance $(c^{(4)}, \mathcal{T}^{(4)})$ considered in Example 3.3.5, the deletion of the last sequence with the wrong sign is reversed as described in the proof of Theorem 3.3.6. This yields primal and dual optimal solutions for the instance $(c^{(3)}, \mathcal{T}^{(3)})$ from Example 3.3.5. Continuing to undo the other preprocessing steps in reversed order finally yields an optimal primal and dual solution for the initial instance $(c, \mathcal{T})$ from Example 3.3.5:

| | | | | | | | | | | | $z_{\{8\}}$ | $z_{\{10\}}$ | $z_{\{13\}}$ | $z_{\{14\}}$ | $c$ | $x$ | $\mathcal{T}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | $\geq -3$ | 0 | $\{1\}$ |
| $+\frac{1}{4}$ | | $+\frac{1}{4}$ | | $+\frac{6}{16}$ | | $+\frac{2}{16}$ | | | | | | | | | $\geq +1$ | 0 | $\{2\}$ |
| | $+\frac{3}{4}$ | | $+\frac{3}{4}$ | | $+\frac{18}{16}$ | | $+\frac{6}{16}$ | | | | | | | | $\geq +3$ | 0 | $\{3\}$ |
| $-\frac{1}{4}$ | $-\frac{3}{4}$ | $-\frac{1}{4}$ | $-\frac{3}{4}$ | $-\frac{6}{16}$ | $-\frac{18}{16}$ | $-\frac{2}{16}$ | $-\frac{2}{16}$ | | | | | | | | $\geq -6$ | 0 | $\{4\}$ |
| $+\frac{1}{4}$ | $+\frac{3}{4}$ | $+\frac{1}{4}$ | $+\frac{3}{4}$ | $+\frac{6}{16}$ | $+\frac{18}{16}$ | $+\frac{2}{16}$ | $+\frac{6}{16}$ | $+0$ | $+\frac{3}{4}$ | $+\frac{1}{4}$ | | | | | $\geq +5$ | 0 | $\{5\}$ |
| | | | | | | | | | | | | | | | $\geq 0$ | 0 | $\{6\}$ |
| $-\frac{1}{4}$ | $-\frac{3}{4}$ | $-\frac{1}{4}$ | $-\frac{3}{4}$ | $-\frac{6}{16}$ | $-\frac{18}{16}$ | $-\frac{2}{16}$ | $-\frac{6}{16}$ | $-0$ | $-\frac{3}{4}$ | $-\frac{1}{4}$ | | | | | $\geq -7$ | 0 | $\{7\}$ |
| $+\frac{1}{4}$ | $+\frac{3}{4}$ | $+\frac{1}{4}$ | $+\frac{3}{4}$ | $+\frac{6}{16}$ | $+\frac{18}{16}$ | | | $+0$ | $+\frac{3}{4}$ | | $+\frac{3}{4}$ | | | | $\geq +5$ | 1 | $\{8\}$ |
| $-\frac{1}{4}$ | $-\frac{3}{4}$ | $-\frac{1}{4}$ | $-\frac{3}{4}$ | | | | | $-0$ | | | | | | | $\geq -2$ | 1 | $\{9\}$ |
| $+\frac{1}{4}$ | $+\frac{3}{4}$ | $+\frac{1}{4}$ | $+\frac{3}{4}$ | | | $+\frac{2}{16}$ | $+\frac{6}{16}$ | $+0$ | | $+\frac{1}{4}$ | | $+\frac{1}{4}$ | | | $\geq +3$ | 1 | $\{10\}$ |
| $-\frac{1}{4}$ | $-\frac{3}{4}$ | $-\frac{1}{4}$ | $-\frac{3}{4}$ | $-\frac{6}{16}$ | $-\frac{18}{16}$ | $-\frac{2}{16}$ | $-\frac{6}{16}$ | $-0$ | $-\frac{3}{4}$ | $-\frac{1}{4}$ | | | | | $\geq -9$ | 0 | $\{11\}$ |
| $+\frac{1}{4}$ | $+\frac{3}{4}$ | | | | | | | | | | | | $+10$ | | $\geq +11$ | 1 | $\{12\}$ |
| $-\frac{1}{4}$ | $-\frac{3}{4}$ | | | | | | | | | | | | | | $\geq -1$ | 1 | $\{13\}$ |
| | | | | | | | | | | | | | | $+2$ | $\geq +2$ | 1 | $\{14\}$ |

Both the correctness of Algorithm 2 and Theorem 3.3.3 are proven. Recall that the efficient solvability of $(\mathrm{BND}^{(1)}_{\mathrm{var}})$ was already established by the polynomial dynamic programming scheme devised in the very beginning of this entire section. However, as it turns out, Algorithm 2 can be implemented to significantly outperform the dynamic programming scheme. As the number of dual variables $y$ with non-zero values can grow exponentially in the course of the construction, the merging algorithm generally does not run in polynomial time if the dual solution is explicitly constructed. But if only the primal solution is required, Algorithm 2 can easily be implemented to run in polynomial time. In the master's thesis [Hal22], supervised by this author, an implementation using heaps with doubly-linked nodes was proposed. The remark below summarizes its findings.

**Remark 3.3.3.** An implementation of Algorithm 2 using heaps with doubly-linked nodes, as proposed in the master's thesis [Hal22], leads to a total running time

of $\mathcal{O}(T \log T)$. The heap is used to identify the entry $\theta$ in constant time in each merging step; see Definition 3.3.4. Note that this is significantly faster than the running time of $\mathcal{O}(T^2)$ required by the dynamic programming scheme; see Lemma 3.3.1. In case $T - S$ is considered a constant, using a doubly-linked list instead of the heap leads to a linear running time; recall that the dynamic programming scheme has linear running time if $S$ is constant. Algorithm 2 also compares favorably in terms of required space: while the dynamic programming approach requires $\mathcal{O}(S \cdot T)$ space, which is quadratic if $S$ is part of the input, the merging algorithm only needs $\mathcal{O}(T)$.

Recall that the extended models (EXT-BND$_0$), (EXT-BND$_1$) and (EXT-BND) presented in Subsection 3.3.1 were defined in dimension $2T + 1$ and $3T + 1$. Thus, they were only slightly larger than the original models (BND$^{(1)}_{\text{var},0}$), (BND$^{(1)}_{\text{var},1}$) and (BND$^{(1)}_{\text{var}}$). Moreover, the number of constraints in all extended models was linear in $T$ as well. On the contrary, the corresponding perfect formulations in the $x$-space that were determined thereafter generally consist of an exponential number of linear constraints. One might hope that these formulations contain unnecessary inequalities that may still be removed. This, however, is not the case as the next section will show that these formulations are in fact minimal.

### 3.3.3 Facets and Separation

As was shown in Subsection 3.3.1 and 3.3.2, the trivial constraints and the respective alternating inequalities (ALT$_0$), (ALT$_1$), and (ALT$^{\Omega}_0$), (ALT$^{\Omega}_1$) yield perfect formulations of the respective polytopes conv($\Omega_0$), conv($\Omega_1$) and conv($\Omega$). It will turn out that these polyhedral descriptions are in fact minimal, as each of the inequalities induces a facet of the corresponding polytope. Again, it suffices to show the claims for conv($\Omega_0$), since the symmetric results for conv($\Omega_1$) are implied, see Remark 3.3.1, and together, they imply the corresponding results for conv($\Omega$).

#### Facets

This paragraph is dedicated to the proof that the trivial inequalities and the alternating inequalities (ALT$_0$) induce facets of conv($\Omega_0$). It begins with the consideration of an easy special case, namely the case in which $S = 1$. In this case, the set of feasible incidence vectors is given by

$$\bar{\Omega}_0 := \left\{(x_0, ..., x_T) \in \{0,1\}^{T+1} \,:\, 0 = x_0 \leq x_1 \leq x_2 \leq ... \leq x_{T-1} \leq x_T \leq 1\right\} .$$

As the set $\bar{\Omega}_0$ consists of $T + 1$ affinely independent elements, conv($\bar{\Omega}_0$) is a simplex of dimension $T$ and its facets are defined by any $T$-element subset of $\bar{\Omega}_0$. Note that $\dim(\text{conv}(\bar{\Omega}_0)) = T$ implies $\dim(\Omega_0) = T$ for any $S \geq 1$ due to $\bar{\Omega}_0 \subseteq \Omega_0$.

**Theorem 3.3.7.** For $S = 1$, the polytope $\text{conv}(\Omega_0)$ is completely described by the linear constraints $0 = x_0 \leq x_1 \leq x_2 \leq ... \leq x_{T-1} \leq x_T \leq 1$. Each of these inequalities defines a facet of $\text{conv}(\Omega_0)$.

*Proof.* Each of the inequalities $x_T \leq 1$ and $x_{t-1} - x_t \leq 0$ for $t \geq 1$ is obviously satisfied with equality for all elements in $\bar{\Omega}_0$ except for one, respectively. Since these $T$ elements are affinely independent, the claim follows. □

Note that the polyhedron $\text{conv}(\bar{\Omega}_0)$ can also be described by the trivial inequalities and the alternating inequalities ($\text{ALT}_0$) as per Theorem 3.3.3. Indeed, each facet-defining inequality $x_{t-1} - x_t \leq 0$ for $t = 1, ..., T$ can be considered an alternating inequality with the index set $\{t_1, t_2\} = \{t - 1, t\}$, $m = 2$ and $\lfloor S/2 \rfloor = \lfloor 1/2 \rfloor = 0$. All of the other alternating inequalities are obviously redundant in this special case. In other words, for $S = 1$, the linear inequality description of $\text{conv}(\bar{\Omega}_0)$ by the inequalities in the definition of $P_0$ is not minimal.

However, it will turn out that the case $S = 1$ is an exception in this regard. Indeed, in the following it will be proven that for $S \geq 2$, the description $P_0$ is in fact a minimal description of $\text{conv}(\Omega_0)$ since all inequalities from Theorem 3.3.3 are in fact facet-inducing.

**Lemma 3.3.10.** For $2 \leq S \leq T$, the trivial inequalities $0 \leq x_t \leq 1$, $t = 1, ..., T$, induce facets of $\text{conv}(\Omega_0)$.

*Proof.* The inequality $x_t \geq 0$ is satisfied with equality by the zero vector and by all unit vectors $e_j \in \mathbb{R}^{T+1}$ for $j \in \{1, ..., T\} \setminus \{t\}$. As $S \geq 2$, these vectors all belong to $\Omega_0$. Thus, $x_t \geq 0$ defines a facet of $\text{conv}(\Omega_0)$.

For the inequality $x_t \leq 1$, define

$$x_j^{(\ell)} = \begin{cases} 0 & \text{if } j < \ell \\ 1 & \text{otherwise} \end{cases} \quad \text{for } \ell = 1, ..., t \quad \text{and}$$

$$x_j^{(\ell)} = \begin{cases} 1 & \text{if } j < \ell \\ 0 & \text{otherwise} \end{cases} \quad \text{for } \ell = t + 1, ..., T\ .$$

Because of $S \geq 2$, extending these elements with $x_0 = 0$ yields $T$ affinely independent elements of $\Omega_0$, all satisfying $x_t = 1$. Thus, $x_t \leq 1$ is a facet-inducing inequality for $\text{conv}(\Omega_0)$. □

The same also holds for the alternating inequalities.

**Lemma 3.3.11.** For $S \geq 2$, each alternating inequality ($\text{ALT}_0$) induces a facet of the switching polytope $\text{conv}(\Omega_0)$.

*Proof.* Given some alternating inequality as in ($\text{ALT}_0$) with index set $\{t_1, ..., t_m\}$, this proof constructs $T$ affinely independent elements of $\Omega_0$ for which the inequality holds with equality. Recall that by definition, $m > S$. For a better overview, the construction is divided into two main steps. First, the focus lies on the support of ($\text{ALT}_0$), recursively constructing $m$ affinely independent, $m$-dimensional vectors so that some alternating inequality in the reduced space holds with equality, afterwards transferring those vectors into $T$-dimensional space. Then, $m-T$ affinely independent vectors are added which satisfy ($\text{ALT}_0$). To begin, consider the case of an odd $S$.

The first step is again subdivided into the following substeps:

1. Consider the following $S+1$ vectors $\hat{x}^{(\ell)} \in \{0,1\}^{S+1}$:

$$\begin{array}{rlll}
\hat{x}^{(1)} = & (\mathbf{0}\ ,\ \mathbf{0}\ ,\ 1\ ,\ 0\ ,\ 1\ ,\ 0\ ,\ 1\ ,\ 0\ , & \dots & 1\ ,\ 0\ ,\ 1\ ,\ 0) \\
\hat{x}^{(2)} = & (1\ ,\ 0\ ,\ \mathbf{0}\ ,\ \mathbf{0}\ ,\ 1\ ,\ 0\ ,\ 1\ ,\ 0\ , & \dots & 1\ ,\ 0\ ,\ 1\ ,\ 0) \\
\hat{x}^{(3)} = & (1\ ,\ 0\ ,\ 1\ ,\ 0\ ,\ \mathbf{0}\ ,\ \mathbf{0}\ ,\ 1\ ,\ 0\ , & \dots & 1\ ,\ 0\ ,\ 1\ ,\ 0) \\
\vdots & \vdots & & \vdots \\
\hat{x}^{(\frac{S+1}{2})} = & (1\ ,\ 0\ ,\ 1\ ,\ 0\ ,\ 1\ ,\ 0\ ,\ 1\ ,\ 0\ , & \dots & 1\ ,\ 0\ ,\ \mathbf{0}\ ,\ \mathbf{0}) \\
\hat{x}^{(\frac{S+1}{2}+1)} = & (\mathbf{1}\ ,\ \mathbf{1}\ ,\ 1\ ,\ 0\ ,\ 1\ ,\ 0\ ,\ 1\ ,\ 0\ , & \dots & 1\ ,\ 0\ ,\ 1\ ,\ 0) \\
\hat{x}^{(\frac{S+1}{2}+2)} = & (1\ ,\ 0\ ,\ \mathbf{1}\ ,\ \mathbf{1}\ ,\ 1\ ,\ 0\ ,\ 1\ ,\ 0\ , & \dots & 1\ ,\ 0\ ,\ 1\ ,\ 0) \\
\hat{x}^{(\frac{S+1}{2}+3)} = & (1\ ,\ 0\ ,\ 1\ ,\ 0\ ,\ \mathbf{1}\ ,\ \mathbf{1}\ ,\ 1\ ,\ 0\ , & \dots & 1\ ,\ 0\ ,\ 1\ ,\ 0) \\
\vdots & \vdots & & \vdots \\
\hat{x}^{(S+1)} = & (1\ ,\ 0\ ,\ 1\ ,\ 0\ ,\ 1\ ,\ 0\ ,\ 1\ ,\ 0\ , & \dots & 1\ ,\ 0\ ,\ \mathbf{1}\ ,\ \mathbf{1})
\end{array}$$

   The vectors $\hat{x}^{(1)}, ..., \hat{x}^{(S+1)} \in \mathbb{R}^{S+1}$ are affinely independent:

   For every $\ell = 1, ..., S+1$ there is an equation which holds for all $\hat{x}^{(j)}$, $j \neq \ell$, but not for $\hat{x}^{(\ell)}$. For $1 \leq \ell \leq \frac{S+1}{2}$, this equation is $x_{2\ell-1} = 1$, while for $\frac{S+1}{2} + 1 \leq \ell \leq S+1$, it is $x_{2\ell-(S+1)} = 0$.

2. If $m > S+1$, construct $m$ vectors $\tilde{x} \in \{0,1\}^m$ with the help of $\hat{x}^{(1)}, ..., \hat{x}^{(S+1)}$ from the previous step; if $m = S+1$, this step can be skipped. Use the following recursive rule for the construction:

   Let $\hat{x}^{(1)}, ..., \hat{x}^{(S+t)}$ denote $S+t$ affinely independent vectors in $\{0,1\}^{S+t}$, for some odd $t$. Define $S+t+2$ affinely independent vectors $\tilde{x}^{(1)}, ..., \tilde{x}^{(S+t+2)} \in \{0,1\}^{S+t+2}$ by

$$\tilde{x}_j^{(\ell)} := \begin{cases} 0 & j = 1, 2 \\ \hat{x}_{j-2}^{(\ell)} & j \geq 3 \end{cases} \quad \text{for} \quad \ell = 1, ...S+t\ ,$$

and

$$\tilde{x}_j^{(S+t+1)} := \begin{cases} 1 & j \text{ odd}, j \leq \lfloor \frac{S}{2} \rfloor \\ 0 & \text{otherwise} \end{cases}, \quad \tilde{x}_j^{(S+t+2)} := \begin{cases} 1 & j = 1, 2 \\ 1 & t = 3, 5, ..., \lfloor \frac{S}{2} \rfloor + 2 \\ 0 & \text{otherwise} \end{cases}.$$

As $\hat{x}^{(1)}, ..., \hat{x}^{(S+t)}$ are affinely independent, so are $\tilde{x}^{(1)}, ..., \tilde{x}^{(S+t)}$. Moreover, all of the latter vectors satisfy $\tilde{x}_1 = 0$, but $\tilde{x}^{(S+t+1)}$ does not, and all $\tilde{x}^{(\ell)}$, $l \neq S + t + 2$, satisfy $\tilde{x}_2 = 0$, but $\tilde{x}^{(S+t+2)}$ does not. Thus, $\tilde{x}^{(1)}, ..., \tilde{x}^{(S+t+2)}$ are affinely independent.

3. As of now, $m$ affinely independent vectors $\tilde{x}^{(\ell)} \in \{0, 1\}^m$ have been constructed, all of which satisfy $\sum_{j=1}^{m} (-1)^{j+1} \tilde{x}_j^{(\ell)} = \lfloor \frac{S}{2} \rfloor$. It remains to transform the constructed vectors into $T$-dimensional vectors feasible for $\Omega_0$. Recall that $t_1, ..., t_m$ denote the indices in the given alternating inequality, and define

$$\begin{aligned} B_1 &:= \{t \in \{1, ..., T\} \mid t < t_1\}, \\ B_j &:= \{t \in \{1, ..., T\} \mid t_{j-1} < t < t_j\} \text{ for } j = 2, ..., m, \text{ and} \\ B_{m+1} &:= \{t \in \{1, ..., T\} \mid t > t_m\}. \end{aligned}$$

Now construct a vector $x^{(\ell)} \in \{0, 1\}^T$ with $(0, x^{(\ell)}) \in \Omega_0$ out of each $\tilde{x}^{(\ell)} \in \{0, 1\}^m$ by using the following rules:

- $x_{t_j}^{(\ell)} := \tilde{x}_j^{(\ell)}$, $j = 1, ..., m$
  (at index $t_j$ vector $x^{(\ell)}$ has the value $\tilde{x}_j^{(\ell)}$)
- $x_t^{(\ell)} := 1$ for all $t \in B_j$, $j = 2, ..., m$, with $x_{t_{j-1}}^{(\ell)} = x_{t_j}^{(\ell)} = 1$
  (gaps between two one-entries are filled with ones)
- $x_t^{(\ell)} := 1$ for all $t \in B_{m+1}$ with $x_{t_m}^{(\ell)} = 1$
  (if there is a one at $t_m$ in $\tilde{x}^{(\ell)}$, then fill all following entries with ones)
- $x_t^{(\ell)} := 0$ otherwise.

Note that these index sets are not necessarily nonempty. However, they contain all indices which do not arise in the considered alternating inequality.

This concludes the first step (for $S$ odd). The second step makes use of the constructed vectors $x^{(1)}, ..., x^{(m)}$ in order to construct additional $T - m$ affinely independent vectors which satisfy the alternating inequality with equality. To this end, use again the index sets defined before.

1. Choose any vector $x^{(\ell)}$ with $x_{t_1}^{(\ell)} = 1$ and construct $|B_1|$ new vectors from $x^{(\ell)}$ by changing the entries contained in $B_1$ as follows (the underlined entry marks $t_1$):

$$(0, 0, \ldots, 0, 0, 1, \underline{1}, \ldots), (0, 0, \ldots 0, 1, 1, \underline{1}, \ldots), \ldots, (1, 1, \ldots, 1, 1, 1, \underline{1})$$

The constructed vectors are affinely independent of the already constructed vectors since for each vector there is an equation which is satisfied by all other vectors but not this particular one, namely, in order of the vectors above:

$$x_{t_1-2} - x_{t_1-1} = 0,\ x_{t_1-3} - x_{t_1-2} = 0,\ ...,\ x_1 = 0$$

2. For each odd $j \in \{3, ..., m\}$, choose any vector $x^{(\ell)}$ with $x^{(\ell)}_{t_{j-1}} = 0$ and $x^{(\ell)}_{t_j} = 1$ and construct $|B_j|$ new vectors from $x^{(\ell)}$ by changing the entries in $B_j$ as follows (the underlined entries mark $t_{j-1}$ and $t_j$):

   $$(\dots, \underline{0}, 1, 1, \dots, 1, 1, \underline{1}, \dots), (\dots, \underline{0}, 0, 1, \dots, 1, 1, \underline{1}, \dots), \dots, (\dots, \underline{0}, 0, 0, \dots, 0, 1, \underline{1}, \dots)$$

   The constructed vectors are affinely independent of the already constructed vectors, the corresponding equations are

   $$x_{t_{j-1}} - x_{t_{j-1}+1} = 0,\ x_{t_{j-1}+1} - x_{t_{j-1}+2} = 0,\ \dots,\ x_{t_j-2} - x_{t_j-1} = 0\ .$$

3. For each even $j \in \{2, ..., m\}$, choose any vector $x^{(\ell)}$ with $x^{(\ell)}_{t_{j-1}} = 1$ and $x^{(\ell)}_{t_j} = 0$ and construct $|B_j|$ new vectors from $x^{(\ell)}$ by changing the entries in $B_j$ as follows (the underlined entries mark $t_{j-1}$ and $t_j$):

   $$(\dots, \underline{1}, 1, 0, \dots, 0, 0, \underline{0}, \dots), (\dots, \underline{1}, 1, 1, \dots, 0, 0, \underline{0}, \dots), \dots, (\dots, \underline{1}, 1, 1, \dots, 1, 1, \underline{0}, \dots)$$

   For the constructed vectors, the corresponding equations are

   $$x_{t_{j-1}+1} - x_{t_{j-1}+2} = 0,\ x_{t_{j-1}+2} - x_{t_{j-1}+3} = 0,\ ...,\ x_{t_j-1} - x_{t_j} = 0\ .$$

4. Choose any $x^{(\ell)}$ with $x^{(\ell)}_{t_m} = 0$ and construct $|B_{m+1}|$ new vectors from $x^{(\ell)}$ by changing the entries in $B_{m+1}$ as follows (the underlined entry marks $t_m$):

   $$(\dots, \underline{0}, 1, 1, \dots, 1, 1), (\dots, \underline{0}, 0, 1, \dots, 1, 1), \dots, (\dots, \underline{0}, 0, 0, \dots, 0, 1)$$

   The constructed vectors are affinely independent of the already constructed vectors, the corresponding equations are

   $$x_{t_m} - x_{t_m+1} = 0,\ x_{t_m+1} - x_{t_m+2} = 0,\ ...,\ x_{T-1} - x_T = 0\ .$$

By extending all constructed vectors with zero in the beginning, one obtains $T$ binary, affinely independent vectors with at most $S$ switchings each, all of which satisfy the alternating inequality with equality. The latter thus induces a facet of $\mathrm{conv}(\Omega_0)$.

It remains to discuss the case of an even $S$. For this, the construction of the $S+1$ vectors in the first step has to be replaced. Consider the following $S+1$ vectors $\hat{x}^{(\ell)} \in \{0,1\}^{S+1}$:

$$
\begin{array}{rcccc}
\hat{x}^{(1)} = & (\mathbf{0}\,,\,\mathbf{0}\,,\,1\,,\,0\,,\,1\,,\,0\,,\,1\,,\,0\,, & \dots & 1\,,\,0\,,\,1) \\
\hat{x}^{(2)} = & (1\,,\,0\,,\,\mathbf{0}\,,\,\mathbf{0}\,,\,1\,,\,0\,,\,1\,,\,0\,, & \dots & 1\,,\,0\,,\,1) \\
\hat{x}^{(3)} = & (1\,,\,0\,,\,1\,,\,0\,,\,\mathbf{0}\,,\,\mathbf{0}\,,\,1\,,\,0\,, & \dots & 1\,,\,0\,,\,1) \\
\vdots & \vdots & & \vdots \\
\hat{x}^{(\frac{S}{2})} = & (1\,,\,0\,,\,1\,,\,0\,,\,1\,,\,0\,,\,1\,,\,0\,, & \dots & \mathbf{0}\,,\,\mathbf{0}\,,\,1) \\
\hat{x}^{(\frac{S}{2}+1)} = & (1\,,\,0\,,\,1\,,\,0\,,\,1\,,\,0\,,\,1\,,\,0\,, & \dots & 1\,,\,0\,,\,\mathbf{0}) \\
\hat{x}^{(\frac{S}{2}+2)} = & (\mathbf{1}\,,\,\mathbf{1}\,,\,1\,,\,0\,,\,1\,,\,0\,,\,1\,,\,0\,, & \dots & 1\,,\,0\,,\,1) \\
\hat{x}^{(\frac{S}{2}+3)} = & (1\,,\,0\,,\,\mathbf{1}\,,\,\mathbf{1}\,,\,1\,,\,0\,,\,1\,,\,0\,, & \dots & 1\,,\,0\,,\,1) \\
\hat{x}^{(\frac{S}{2}+4)} = & (1\,,\,0\,,\,1\,,\,0\,,\,\mathbf{1}\,,\,\mathbf{1}\,,\,1\,,\,0\,, & \dots & 1\,,\,0\,,\,1) \\
\vdots & \vdots & & \vdots \\
\hat{x}^{(S+1)} = & (1\,,\,0\,,\,1\,,\,0\,,\,1\,,\,0\,,\,1\,,\,0\,, & \dots & \mathbf{1}\,,\,\mathbf{1}\,,\,1)
\end{array}
$$

The vectors $\hat{x}^{(1)}, ..., \hat{x}^{(S+1)} \in \mathbb{R}^{S+1}$ are affinely independent:

For $1 \leq \ell \leq \frac{S}{2}+1$, all $\hat{x}^{(j)}$ with $j \neq \ell$ satisfy $\hat{x}_{2\ell-1} = 1$, but $\hat{x}^{(\ell)}$ does not, and for $\frac{S}{2}+2 \leq \ell \leq S+1$, all $\hat{x}^{(j)}$ with $j \neq \ell$ satisfy the equation $\hat{x}_{2\ell-(S+2)} = 0$, but not $\hat{x}^{(\ell)}$. Furthermore, in the second step, replace the construction of the $|B_{m+1}|$ new vectors with the following:

Choose any vector $x^{(\ell)}$ with $x^{(\ell)}_{t_m} = 1$ and construct $|B_{m+1}|$ new vectors from $x^{(\ell)}$ by changing the entries in $B_{m+1}$ as follows (the underlined entry marks $t_m$):

$$(\dots, \underline{1}, 0, 0, \dots, 0, 0), (\dots, \underline{1}, 1, 0, \dots, 0, 0), \dots, (\dots, \underline{1}, 1, 1, \dots, 1, 0)$$

The constructed vectors are affinely independent of the other constructed vectors, the corresponding equations stay the same. The rest of the construction is the same as for an odd $S$. □

Since by Theorem 3.3.3, the polytope $\text{conv}(\Omega_0)$ is completely described by the trivial inequalities and the alternating inequalities ($\text{ALT}_0$) where each of the latter induces a facet for $S \geq 2$, it follows:

**Corollary 3.3.1.** For $S \geq 2$, the trivial inequalities and the alternating inequalities ($\text{ALT}_0$) yield a minimal perfect formulation of $\text{conv}(\Omega_0)$.

Furthermore, the symmetry between $\Omega_0$ and $\Omega_1$ (see Remark 3.3.1) yields

**Theorem 3.3.8.** For $S = 1$, the polytope $\text{conv}(\Omega_1)$ is completely described by the linear constraints $1 = x_0 \geq x_1 \geq x_2 \geq ... \geq x_{T-1} \geq x_T \geq 0$. Each of these inequalities defines a facet of $\text{conv}(\Omega_1)$.

For $S \geq 2$, each of the trivial inequalities $0 \leq x_t \leq 1$, $t = 1, ..., T$ and each alternating inequality ($\text{ALT}_1$) induces a facet of the polytope $\text{conv}(\Omega_1)$. Hence, the description of $\text{conv}(\Omega_1)$ by these inequalities is minimial.

By using the same idea behind the construction of the affinely independent vectors in the proof of Lemma 3.3.11, one can also show

**Theorem 3.3.9.** For $S \geq 2$, each of the alternating inequalities $(\mathrm{ALT}_0^\Omega)$ and $(\mathrm{ALT}_1^\Omega)$ induces a facet of the polytope $\mathrm{conv}(\Omega)$.

*Proof.* The proof is only briefly sketched, since it follows with analogous reasoning as for Lemma 3.3.11. First note that the dimension of $\mathrm{conv}(\Omega)$ is $T+1$ because all $T+1$-dimensional unit vectors are feasible.

Now, consider an alternating inequality $(\mathrm{ALT}_0^\Omega)$ as defined in Lemma 3.3.2. It is possible to determine $T+1$ affinely independent points by using the same strategy as in the previously described construction:

For $S$ odd, the construction begins with $S+2$ affinely independent points in $\{0,1\}^{S+2}$. In fact, one may use the $S+1$-dimensional points $\hat{x}^{(j)}$ defined in the first step of the proof of Lemma 3.3.11 and extend all of them by appending zero in the first position, i.e., $x^{(j)} := (0, \hat{x}^{(j)})$ for $j = 1, ..., S+1$. Additionally, define one more point by appending a one in the beginning of $\hat{x}^{(\frac{S+1}{2}+1)}$, i.e., $(1, \hat{x}^{(\frac{S+1}{2}+1)})$.

If $S$ is even, then define $S+2$ affinely independent points $x^{(j)} \in \{0,1\}^{S+2}$ with $j = 1, ..., S+2$ by $x^{(j)} := (0, \hat{x}^{(j)})$ for $j = 1, ..., S+1$, where the latter points are the elements in the first step for $S$ even in the proof of Lemma 3.3.11. Define one additional point via $x^{(S+2)} := (1, \hat{x}^{(\frac{S}{2}+1)}) - (\mathbb{0}_{S+1}, 1)$.

Inductively constructing $T+1$ elements as described before shows that each inequality $(\mathrm{ALT}_0^\Omega)$ induces a facet. The respective symmetric vectors then immediately show that the corresponding symmetric inequalities $(\mathrm{ALT}_1^\Omega)$ induce facets as well. □

It is obvious that each of the trivial constraints $0 \leq x_t$ and $x_t \leq 1$, $t = 0, ..., T$, also induces a facet of $\mathrm{conv}(\Omega)$ if $S \geq 2$. Hence, the description of $\mathrm{conv}(\Omega)$ by the inequalities defined in Theorem 3.3.2 is minimal if $S \geq 2$, too.

As a result of this subsection, for $S \geq 2$, the perfect formulations for $\mathrm{conv}(\Omega)$, $\mathrm{conv}(\Omega_0)$, and $\mathrm{conv}(\Omega_1)$ as stated in Theorems 3.3.2, 3.3.3, and 3.3.4 are in fact minimal, i.e., they do not contain any redundant inequalities. However, the number of their linear inequalities is generally exponential.

**Remark 3.3.4.** For each $m > S$ with the opposite parity of $S$, the number of index sets $\{t_1, ..., t_m\}$ for an alternating inequality $(\mathrm{ALT}_0)$, or $(\mathrm{ALT}_1)$, is given by $\binom{T}{m} = \mathcal{O}(\frac{1}{m!}\, T^{T-m})$, which is exponential. In fact, note that this number is still exponential, even if $S$ is constant and thus, the number of feasible solutions to $(\mathrm{BND}^{(1)}_{\mathrm{var},0})$, or $(\mathrm{BND}^{(1)}_{\mathrm{var},1})$, is polynomial in $T$.
If, however, the difference $T - S$ is constant instead, then the number of index sets

is polynomial in $T$, since $m = \mathcal{O}(S)$. Moreover, since there are approximately $\frac{T-S}{2}$ elements $m \in \mathbb{Z}$ with $m > S$ so that $m$ and $S$ do not have the same parity, the overall number of alternating inequalities ($\mathrm{ALT}_0$), or ($\mathrm{ALT}_1$), is polynomial.

Clearly, the same holds for the alternating inequalities ($\mathrm{ALT}_0^\Omega$) and ($\mathrm{ALT}_1^\Omega$).

Due to the exponential number of linear inequalities, deciding the feasibility of some $x \in [0,1]^{T+1}$ cannot be done efficiently by substitution in the linear inequality description $P_0$ (or $P_1$, or $P$). Of course, the feasibility can instead be checked by substitution in the extended model with the help of the auxiliary variables. Nonetheless, the feasibility can also be checked only in the original $x$-space via the corresponding separation problem, which will be studied the next subsection.

### Separation

This subsection is dedicated to the study of the separation problem regarding the polytope $P_0$ (or $P_1$, or $P$), which, as a result of the preceding sections, is known to be a minimal perfect formulation of $\mathrm{conv}(\Omega_0)$ (or $\mathrm{conv}(\Omega_1)$, or $\mathrm{conv}(\Omega)$) with an exponential number of linear inequalities. The separation algorithm that will be devised in this section shows that these inequalities can still be separated efficiently. Due to the symmetry described in Remark 3.3.1, it once again suffices to consider the polytope $P_0$, since the symmetric results for $P_1$ and then, in turn, $P$ will be implied.

For the separation problem regarding the polytope $P_0$, one may assume that all trivial inequalities are satisfied, since this is easily checked. Given some $\bar{x} \in [0,1]^{T+1}$ with $\bar{x}_0 = 0$, the aim is thus to determine an alternating inequality for $P_0$ which is (most) violated by $\bar{x}$, or to determine that $\bar{x} \in P_0$.

**Assumption 3.3.1.** For the following, assume that $\bar{x}$ does not contain any pair of consecutive entries which have the same value.

This assumption only serves the purpose to simplify the reasoning in the following proofs. At the end of this section, the proposed separation technique is adapted to the general case. For the separation, the local extrema of $\bar{x}$ play an important role. A *local maximum* of $\bar{x}$ is defined as an index $t \in \{1, ..., T\}$ such that $\bar{x}_t \geq \bar{x}_{t-1}$ and, in case $t < T$, also $\bar{x}_t \geq \bar{x}_{t+1}$. A *local minimum* is defined analogously.

First, for an even $S$, the following result is proven step-by-step.

**Theorem 3.3.10.** For $\bar{x} \in [0,1]^{T+1}$ with $\bar{x}_0 = 0$, let $\{\theta_1, \theta_2, ..., \theta_\ell\}$ denote the set of all local maxima and minima of $\bar{x}$, read from left to right, excluding zero and up

to the last local maximum. Let $S$ be even. Then, an alternating inequality ($\text{ALT}_0$) most violated by $\bar{x}$ is given by

$$(\text{ALT}_0^*) \qquad x_{\theta_1} - x_{\theta_2} + x_{\theta_3} - x_{\theta_4} + ... - x_{\theta_{\ell-1}} + x_{\theta_\ell} \leq {}^S/_2 \ .$$

In particular, this yields that $\bar{x} \in P_0$ if and only if ($\text{ALT}_0^*$) is satisfied by $\bar{x}$. Note that local maxima and minima arise alternately in $\{\theta_1, \theta_2, ..., \theta_\ell\}$, starting with a local maximum since $\bar{x}_0 = 0$. In particular, $\ell$ must be odd and therefore, ($\text{ALT}_0^*$) is of the form ($\text{ALT}_0$) and hence valid for $P_0$.

Theorem 3.3.10 is proven in several steps. The first step shows that a most violated alternating inequality must contain all local maxima in the given vector $\bar{x}$:

**Lemma 3.3.12.** Let $S$ be even and let the indices $t_1, ..., t_m$ define an alternating inequality most violated by $\bar{x}$. Then, $\theta_f \in \{t_1, ..., t_m\}$ for all $f$ odd.

*Proof.* Let the indices $t_1, ..., t_m$ define an alternating inequality most violated by $\bar{x}$. assume that the claim does not hold, i.e., let $f$ be odd with $\theta_f \notin \{t_1, ..., t_m\}$. In the following case distinction, each case results in a contradiction as it derives an alternating inequality more violated than

$$(3.1) \qquad x_{t_1} - x_{t_2} + x_{t_3} - ... - x_{t_{m-1}} + x_{t_m} \leq \frac{S}{2} \ .$$

First assume that there exists some $j \in \{1, ..., m-1\}$ with $t_j < \theta_f < t_{j+1}$.

If $j$ is odd, consider the index $\theta_h$ with $t_j \leq \theta_h$ where $h$ is odd and minimal. In words, $\theta_h$ refers to the local maximum which is closest to the right of $t_j$, so it is $t_j \leq \theta_h \leq \theta_f$. In case $\theta_h = \theta_f$, one derives an alternating inequality which is more violated than (3.1) by replacing the term $+x_{t_j}$ with $+x_{\theta_f}$ in (3.1). Otherwise, there is an index $\theta_{f-1}$ such that $t_j \leq \theta_h < \theta_{f-1} < \theta_f < t_{j+1}$, where $\theta_{f-1}$ is a local minimum which is not part of (3.1) either. Replacing the term $+x_{t_j}$ with $+x_{\theta_h} - x_{\theta_{f-1}} + x_{\theta_f}$ yields an alternating inequality which is more violated by $\bar{x}$ than (3.1).

If $j$ is even, the argument is similar: consider the index $\theta_h$ with $\theta_h \leq t_{j+1}$ where $h$ is odd and maximal. The index $\theta_h$ refers to the local maximum which is closest to the left of $t_{j+1}$, so, it is $\theta_f \leq \theta_h \leq t_{j+1}$. In case $\theta_f = \theta_h$, replace the term $+x_{t_{j+1}}$ in (3.1) by $+x_{\theta_f}$ and thus obtain a more violated inequality. Otherwise, there is an index $\theta_{f+1}$ with $t_j < \theta_f < \theta_{f+1} < \theta_h \leq t_{j+1}$, so that $\theta_{f+1}$ is a local minimum which is not part of (3.1) either. Substituting the term $+x_{t_{j+1}}$ in (3.1) with $+x_{\theta_f} - x_{\theta_{f+1}} + x_{\theta_h}$ yields an alternating inequality which is violated more than (3.1).

This concludes the proof in case $t_j < \theta_f < t_{j+1}$ for some $j \in \{1, ..., m-1\}$. For the case $\theta_f < t_1$, replace $t_{j+1}$ by $t_1$ and argue the same way as before for an even $j$; in case $\theta_f > t_m$, replace $t_j$ by $t_m$ and argue as before for an odd $j$. □

So according to Lemma 3.3.12, a most violated alternating inequality has to contain all local maxima in $\bar{x}$ in its support. Now, all local minima lying between two local maxima in $\bar{x}$ must be contained in a most violated inequality as well:

**Lemma 3.3.13.** Let $S$ be even and let the indices $t_1, ..., t_m$ define an alternating inequality most violated by $\bar{x}$. Then, $\theta_f \in \{t_1, ..., t_m\}$ for all $f$ even.

*Proof.* Assume that the claim does not hold, i.e., let $f$ be even with $\theta_f \notin \{t_1, ..., t_m\}$. By Lemma 3.3.12, there is some $j \in \{1, ..., m-1\}$ with $t_j < \theta_f < t_{j+1}$. Again, consider different cases and either find a more violated alternating inequality or obtain a contradiction to Lemma 3.3.12 in each case.

If $j$ is odd, consider the index $\theta_h$ with $\theta_h \leq t_{j+1}$ where $h$ is even and maximum, so $\theta_h$ is the local minimum closest to the left of $t_{j+1}$ and it is $\theta_f \leq \theta_h \leq t_{j+1}$. In case $\theta_f = \theta_h$, replace the term $-x_{t_{j+1}}$ in (3.1) with $-x_{\theta_f}$ and obtain a more violated inequality of the form ($\text{ALT}_0$). Otherwise there exists an index $\theta_{f+1}$ with $t_j < \theta_f < \theta_{f+1} < \theta_h \leq t_{j+1}$, so there is a local maximum in $\bar{x}$ which is not contained in (3.1) which contradicts Lemma 3.3.12.

If $j$ is even, the argument is similar: consider the index $\theta_h$ with $h$ even and minimal such that $t_j \leq \theta_h \leq \theta_f$. If $\theta_f = \theta_h$, replace the term $-x_{t_j}$ in (3.1) with $-x_{\theta_f}$ and obtain a more violated inequality of the form ($\text{ALT}_0$). Otherwise, there exists an index $\theta_{f-1}$ with $t_j \leq \theta_h < \theta_{f-1} < \theta_f < t_{j+1}$, so there is a local maximum in $\bar{x}$ which is not part of (3.1), contradicting Lemma 3.3.12. □

Due to Lemma 3.3.12 and Lemma 3.3.13, a most violated alternating inequality must contain all indices $\theta_1, \theta_2, \theta_3, ..., \theta_l$. However, it is not clear that such a most violated inequality does not contain any other indices. The next lemma implies that in a most violated alternating inequality, each variable corresponding to the local extrema at indices $\theta_1, ..., \theta_l$ is assigned the *correct* sign.

**Lemma 3.3.14.** Let $S$ be even and let the indices $t_1, ..., t_m$ define an alternating inequality most violated by $\bar{x}$. If $\theta_f = t_j$, then $j$ is even if and only if $f$ is even.

*Proof.* Assume that the claim is not true, i.e., let $f$ be odd, so that $\theta_f$ is a local maximum in $\bar{x}$, and assume that $x_{\theta_f}$ appears with coefficient $-1$ in (3.1). If $\theta_f + 1 \in \{t_1, ..., t_m\}$, delete $-x_{\theta_f} + x_{\theta_f+1}$ from (3.1) and obtain a more violated alternating inequality than (3.1), since $\bar{x}_{\theta_f} > \bar{x}_{\theta_f+1}$. If $\theta_f < n$ and $\theta_f + 1 \notin \{t_1, ..., t_m\}$, replace $-x_{\theta_f}$ by $-x_{\theta_f+1}$ in (3.1) to obtain a more violated inequality. If $\theta_f = n$, the same construction is possible with $\theta_f - 1$ instead of $\theta_f + 1$. In case $f$ is even, so that $\theta_f$ is a local minimum in $\bar{x}$, the proof is analogous. □

Now, Lemma 3.3.12, Lemma 3.3.13, and Lemma 3.3.14 are exploited to prove that a most violated alternating inequality cannot contain any indices other than $\theta_1, ..., \theta_l$.

**Lemma 3.3.15.** Let $S$ be even and let the indices $t_1, ..., t_m$ define an alternating inequality most violated by $\bar{x}$. Then, $\{\theta_1, \theta_2, ..., \theta_l\} = \{t_1, t_2, ..., t_m\}$.

*Proof.* The inclusion $\subseteq$ was shown in Lemma 3.3.12 and Lemma 3.3.13. For the reverse inclusion, let $j \in \{1, 2, ..., m\}$ be minimal such that $t_j \notin \{\theta_1, \theta_2, ..., \theta_l\}$. Then, by Lemma 3.3.14, also $t_{j+1} \notin \{\theta_1, \theta_2, ..., \theta_l\}$, and deleting $x_{t_j}$ and $x_{t_{j+1}}$ from (3.1) leads to an alternating inequality more violated by $\bar{x}$ than (3.1). □

Consequently, Theorem 3.3.10 is proven to be correct. The following similar result for the case of an odd $S$ can be proven with the same logic.

**Theorem 3.3.11.** For $\bar{x} \in [0, 1]^{T+1}$ with $\bar{x}_0 = 0$, let $\{\theta_1, \theta_2, ..., \theta_\ell\}$ denote the set of all local maxima and minima of $\bar{x}$, read from left to right, excluding zero and up to the last local minimum. Let $S$ be odd. Then, an alternating inequality most violated by $\bar{x}$ is given by

$$x_{\theta_1} - x_{\theta_2} + x_{\theta_3} - x_{\theta_4} + ... + x_{\theta_{\ell-1}} - x_{\theta_\ell} \leq \lfloor \tfrac{S}{2} \rfloor \, .$$

Again, all indices $\theta_1, \theta_2, ..., \theta_\ell$ must appear in the most violated inequality, by the following two lemmas.

**Lemma 3.3.16.** Let $S$ be odd and let the indices $t_1, ..., t_m$ define an alternating inequality most violated by $\bar{x}$. Then, $\theta_f \in \{t_1, ..., t_m\}$ for all $f$ odd.

*Proof.* Suppose there is some odd $f$ with $\theta_f \notin \{t_1, ..., t_m\}$. If $\theta_f < t_m$, the proof is analogous to the proof of Lemma 3.3.12, it thus suffices to consider the case $\theta_f > t_m$. Then, the local minimum $\theta_{f+1}$ does not belong to $\{t_1, ..., t_m\}$ as well, and a more violated alternating inequality can be obtained by adding the term $+x_{\theta_f} - x_{\theta_{f+1}}$ to the left hand side. □

**Lemma 3.3.17.** Let the indices $t_1, ..., t_m$ define an alternating inequality most violated by $\bar{x}$. Then, $\theta_f \in \{t_1, ..., t_m\}$ for all $f$ even.

*Proof.* Let $f$ be even with $\theta_f \notin \{t_1, ..., t_m\}$. Then, there exists a local maximum $\theta_{f-1}$. Lemma 3.3.16 implies that $\theta_{f-1} \in \{t_1, ..., t_m\}$. If $\theta_f < t_m$, argue as in Lemma 3.3.13, so assume that $\theta_f > t_m$.

The reasoning is similar to before: consider the index $\theta_h$ with $h$ even and $t_m \leq \theta_h$ so that $h$ is minimum. So, $\theta_h$ refers to a local minimum which is closest to the right of $t_m$ and it is $\theta_h \leq \theta_f$. If $\theta_h = \theta_f$, then replacing the term $-x_{t_m}$ by $-x_{\theta_f}$ results in an alternating inequality more violated by $\bar{x}$ than the original inequality. Otherwise, there is a local maximum $\theta_{f-1}$ with $t_m \leq \theta_h < \theta_{f-1} < \theta_f$, so $\theta_{f-1}$ is not included in the original inequality, contradicting Lemma 3.3.16. □

As a result, all indices $\theta_1, ..., \theta_\ell$ must be contained in a most violated alternating inequality. With Lemmas 3.3.16, 3.3.17, 3.3.18 and the same reasoning as in the proof of Lemma 3.3.15, it follows that for $S$ odd, a most violated alternating inequality cannot contain any indices other than $\theta_1, ..., \theta_\ell$.

**Lemma 3.3.18.** Let $S$ be odd and let the indices $t_1, ..., t_m$ define an alternating inequality most violated by $\bar{x}$. If $\theta_f = t_j$, then $j$ is even if and only if $f$ is even.

This concludes the proof of Theorem 3.3.11. According to Theorems 3.3.10 and 3.3.11, a most violated alternating inequality is always given by the local minima and maxima of $\bar{x}$. By Assumption 3.3.1, local extrema are strictly larger than the neighboring entries. In this case, there is a unique most violated inequality. However, if this assumption is omitted, it may not be possible to identify unique local maxima and minima in $\bar{x}$. Nevertheless, all statements and proofs remain valid by simply choosing an arbitrary index out of each consecutive sequence of indices with an equal and locally extreme value in $\bar{x}$.

**Example 3.3.10.** Let $S = 2$, $T = 7$, and $\bar{x} = (0, 0.1, 0.9, 0.2, 0.7, 0.5, 0.3, 0.1)$. Then, Theorem 3.3.15 yields $\theta_1 = 2$, $\theta_2 = 3$, $\theta_3 = 4$ with the unique most violated alternating inequality $x_2 - x_3 + x_4 \leq 1$.

Now let $S = 2, T = 7$, and $\bar{x} = (0, 0.9, 0.9, 0.2, 0.7, 0.7, 0.7, 0.5)$. Then, as $\theta_1$, one may choose between 1 and 2, while for $\theta_3$, one can choose 4, 5, or 6. Thus, there are six most violated inequalities, given as follows:

$$\begin{aligned} x_1 - x_3 + x_4 &\leq 1 & x_2 - x_3 + x_4 &\leq 1 \\ x_1 - x_3 + x_5 &\leq 1 & x_2 - x_3 + x_5 &\leq 1 \\ x_1 - x_3 + x_6 &\leq 1 & x_2 - x_3 + x_6 &\leq 1 \end{aligned}$$

However, one may also consider 5 as a local minimum between the local maxima 4 and 6, yielding two other most violated inequalities

$$\begin{aligned} x_1 - x_3 + x_4 - x_5 + x_6 &\leq 1 \\ x_2 - x_3 + x_4 - x_5 + x_6 &\leq 1 \, . \end{aligned}$$

Theorems 3.3.15 and 3.3.11 and the above considerations then give the main result of this subsection:

**Corollary 3.3.2.** The separation problem for $P_0$ can be solved in $\mathcal{O}(T)$ time, which is the time required to determine the local maxima and minima in $\bar{x}$.

Symmetry also yields the following:

**Theorem 3.3.12.** For $\bar{x} \in [0,1]^{T+1}$ with $\bar{x}_0 = 1$, let $\{\theta_1, \theta_2, ..., \theta_l\}$ denote the set of all local maxima and minima of $\bar{x}$, read from left to right, excluding zero and up to the last local minimum, if $S$ is even, or to the last local maximum otherwise. Then, an alternating inequality ($\text{ALT}_1$) most violated by $\bar{x}$ is given by

$$(\text{ALT}_1^*) \qquad -x_{\theta_1} + x_{\theta_2} - x_{\theta_3} + ...(-1)^m x_{\theta_l} \leq \lceil \tfrac{S}{2} \rceil - 1 \ .$$

**Corollary 3.3.3.** The separation problem for $P_1$ can be solved in $\mathcal{O}(T)$ time.

With analogous reasoning, one can show that the separation problem for $\text{conv}(\Omega)$ can also be solved efficiently by identifying local maxima and minima in some given $\bar{x} \in [0,1]^{T+1}$. Note the presence of an additional element $\theta_0$ in the theorem below and that the first entry $\bar{x}_0$ of some given $\bar{x}$ is no longer excluded in the consideration of local extrema.

**Theorem 3.3.13.** For $\bar{x} \in [0,1]^{T+1}$, let $\{\theta_0, \theta_1, ..., \theta_l\}$ denote the set of all local maxima and minima of $\bar{x}$, read from left to right, beginning with the first local minimum and up to the last local maximum/minimum, if $S$ is even/odd. Then, an alternating inequality ($\text{ALT}_0^\Omega$) most violated by $\bar{x}$ is given by

$$-x_{\theta_0} + x_{\theta_1} - x_{\theta_2} + x_{\theta_3} + ...(-1)^{m+1} x_{\theta_l} \leq \lfloor \tfrac{S}{2} \rfloor \ .$$

A most violated alternating inequality ($\text{ALT}_1^\Omega$) is given by

$$+x_{\theta_0} - x_{\theta_1} + x_{\theta_2} - x_{\theta_3} + ...(-1)^m x_{\theta_l} \leq \lceil \tfrac{S}{2} \rceil \ ,$$

where $\{\theta_0, \theta_1, ..., \theta_l\}$ contains the set of all local maxima and minima, beginning with the first local maximum and up to the last local minimum/maximum if $S$ is even/odd.

This concludes the study of the special case $n = 1$ conducted in this chapter. It turned out that in this special case where there is only one switch, both the penalty

problem and the bounded problem are tractable for all three types of variation which are considered in this thesis. The tractability of all problem versions was proven by presenting a corresponding compact extended formulation, which was shown to be integral when continuously relaxed. However, the focus lay on the investigation of the corresponding polyhedral structure. As a result, for each problem version, a perfect formulation in the original variable space was obtained. All these perfect formulations required an exponential number of linear inequalities, with time-wise variation being the only exception. This motivated the study of the respective separation problem, which in turn was proven to be solvable in polynomial time. Based on the idea to merge consecutive entries in a particular way, the chapter also devised an efficient combinatorial optimization algorithm for the bounded problem with total or switch-wise variation, which turned out to significantly outperform a previous naive dynamic programming scheme.

The next chapter will extend the current setting to more than one switch and additional combinatorial constraints for each stage. Recall that the study of the bounded problem with switch-wise/total variation turned out to be the most sophisticated among the versions studied in this chapter. In fact, the results of this chapter might even be seen as an indication of a trend that will continue on a broader scale: Indeed, the bounded problem with switch-wise variation will turn out to be intractable already for a very easy combinatorial structure, whereas the bounded problem with time-wise variation and the penalty case remain polynomially solvable.

# Chapter 4

# Multi-Switch Selection subject to Bounded Variation

In the previous chapter, the single switch problem (BND) with $n = 1$ has been studied for all types of variation. Although the respective problems differed depending on the type of variation and whether the penalized or bounded problem was considered, all problem versions turned out to be tractable. In the special case where $n = 1$, there was no combinatorial structure that was worth studying other than the entire binary space $X_t := \{0, 1\}$, and it will not be very challenging to show that without any further restrictions on the sets $X_t \subseteq \{0, 1\}^n$, all of the resulting problem versions remain tractable for $n > 1$. It is unclear, however, how the respective problem's complexity is impacted if an additional constraint for each of the combinatorial structures $X_t$ is involved. Of course, in this context, it does not make much sense to study a constraint that obviously renders the problem $\mathcal{NP}$-hard, therefore it is reasonable to require that this additional constraint be *simple*, in the sense that linear optimization over the respective feasible set is possible in polynomial time. Probably the simplest such nontrivial constraint is a SELECTION constraint where the sets $X_t$ are given as follows:

$$\text{(SEL)} \qquad X_t := \left\{ x_t \in \{0,1\}^n \,:\, \sum_{i=1}^{n} x_t^{(i)} = K_t \right\} \qquad \text{for all } t = 0, ..., T \,,$$

with $K_t \in \mathbb{N}_{\geq 0}$ for all $t = 0, ..., T$. Recall that this set is a simple set since any linear objective can be efficiently optimized over it by just determining the best $K_t$ elements.

This chapter is driven by the question how the complexity of each of the problem versions behaves if the problems (P) and (BND) include this particular selection constraint (SEL). It will turn out that the complexity of the resulting problem depends heavily on the considered type of variation.

Motivated by the previous chapter, i.e., in the hope that the penalty problem is somehow easier to deal with, the problem (P) will be studied first. Afterwards, the

bounded problem (BND) with time-wise variation will be considered. Although this problem version will become more involved compared to the special case of a single switch (see Section 3.2), it will still remain tractable. In the subsequent section, the problem (BND) with switch-wise variation will be investigated. Since $n > 1$, switch-wise variation and total variation now no longer agree. In fact, the problem (BND) with switch-wise variation will be proven to be strongly $\mathcal{NP}$-hard, if an additional selection constraint as in (SEL) is present. The section thereafter will briefly tackle the problem (BND) with total variation and an additional selection constraint. However, the complexity of this problem will remain an open research question. The final section will collect a few related problems and present some interesting possible generalizations.

## 4.1 Penalized Variation

This section will deal with the penalty problem (P) and will settle its complexity both for the case $X_t = \{0,1\}^n$, as well as for the case in which a selection constraint as in (SEL) is involved. Both problem versions will turn out to be tractable via linear programming. The approach in this section, although slightly more complicated, can still be thought of as an extension of the ideas already used in Section 3.1 for the (unconstrained) single switch problem.

To begin, the unconstrained penalty problem, i.e., (P) with $X_t = \{0,1\}^n$ for all $t$ is discussed. Recall that this unconstrained problem for multiple switches reads

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} \sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} - \lambda^\top \mathrm{Var}(x) \\ \text{s.t.} \quad & x_t \in \{0,1\}^n \qquad t = 0, ..., T \ , \end{aligned}$$

where depending on the variation, $\lambda \in \mathbb{R}^r_{>0}$ denotes a penalty vector of the appropriate dimension. Similar to Section 3.1, the above problem can be linearized by introducing auxiliary variables $u_t^{(i)}$ and $d_t^{(i)}$ for all $i = 1, ..., n$ and $t = 1, ..., T$, which indicate the activations and deactivations of a switch $i \in \{1, ..., n\}$ at stage $t$. Since these auxiliary variables $u$ and $d$ receive individual penalties $\mu$ and $\delta$ in the objective

function, the resulting problem is more general than (P) as the deactivations and activations can incur different penalties. The resulting problem is thus given by

$$\text{(PEN-LIN)} \quad \begin{cases} \max & \displaystyle\sum_{i=1}^{n}\sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} - \sum_{i=1}^{n}\sum_{t=1}^{T} \left(\mu_t^{(i)} u_t^{(i)} + \delta_t^{(i)} d_t^{(i)}\right) & \\ \text{s.t.} & -x_{t-1}^{(i)} + x_t^{(i)} - u_t^{(i)} \leq 0 & t = 1,...,T,\ i = 1,...,n \\ & +x_{t-1}^{(i)} - x_t^{(i)} - d_t^{(i)} \leq 0 & t = 1,...,T,\ i = 1,...,n \\ & u_t^{(i)}, d_t^{(i)} \geq 0 & t = 1,...,T,\ i = 1,...,n \\ & x_t^{(i)} \in \{0,1\} & t = 0,...,T,\ i = 1,...,n\ . \end{cases}$$

It is easy to recognize that the above problem is actually an equivalent reformulation of (P) with $X_t = \{0,1\}^n$ if $\mu_t^{(i)}$ and $\delta_t^{(i)}$ for all $t = 1,...,T$ and all $i = 1,...,n$ are defined as follows

$$\mu_t^{(i)} := \delta_t^{(i)} := \begin{cases} \lambda & \text{for total variation,} \\ \lambda_t & \text{for time-wise variation,} \\ \lambda_i & \text{for switch-wise variation .} \end{cases}$$

Furthermore, observe that the above problem decomposes into $n$ separate single-switch penalty problems ($P^{(1)}$-LIN) from Section 3.1. Thus, the total unimodularity of the constraint matrix of ($P^{(1)}$-LIN) immediately implies the total unimodularity of the constraint matrix of (PEN-LIN). Hence, the unconstrained penalty problem can be solved as an LP corresponding to the continuous relaxation of (PEN-LIN).

**Theorem 4.1.1.** The penalty problem (P) with $X_t = \{0,1\}^n$ for all $t = 0,...,T$ can be solved by linear programming.

*Proof.* The corresponding constraint matrix of (PEN-LIN) can be considered a diagonal block matrix, where the $i$-th matrix on the diagonal line models the corresponding constraints between $x_t^{(i)}$,$u_t^{(i)}$ and $d_t^{(i)}$ for all $t = 1,...,T$. More precisely, the $i$-th diagonal matrix is given by

$$\begin{pmatrix} A & -I_T & 0 \\ -A & 0 & -I_T \end{pmatrix},$$

where the columns of $A \in \{0,1\}^{T\times(T+1)}$ correspond to the variables $x^{(i)}$. The latter is exactly the constraint matrix of ($P^{(1)}$-LIN) and hence, totally unimodular. Since the constraint matrix of (PEN-LIN) is a diagonal matrix and because integer lower and upper bounds on the variables do not destroy total unimodularity, this implies the total unimodularity of the entire constraint matrix of (PEN-LIN). All right hand sides are integer, thus the corresponding continuous relaxation is integral. □

Now, take a look at the penalty problem when an additional selection constraint as in (SEL) is involved. This selection constraint can easily be integrated into the unbounded linearized problem (PEN-LIN), giving rise to the following problem:

$$
(\text{PEN-LIN}^{\text{SEL}}) \quad \begin{cases}
\max & \displaystyle\sum_{i=1}^{n}\sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} - \sum_{i=1}^{n}\sum_{t=1}^{T} \left(\mu_t^{(i)} u_t^{(i)} + \delta_t^{(i)} d_t^{(i)}\right) & & \\
\text{s.t.} & \displaystyle\sum_{i=1}^{n} x_t^{(i)} = K_t & t = 0, ..., T & (1) \\
& -x_{t-1}^{(i)} + x_t^{(i)} - u_t^{(i)} \leq 0 & t = 1, ..., T,\ i = 1, ..., n & (2) \\
& +x_{t-1}^{(i)} - x_t^{(i)} - d_t^{(i)} \leq 0 & t = 1, ..., T,\ i = 1, ..., n & (3) \\
& u_t^{(i)}, d_t^{(i)} \geq 0 & t = 1, ..., T,\ i = 1, ..., n & (4) \\
& x_t^{(i)} \in \{0, 1\} & t = 0, ..., T,\ i = 1, ..., n & (5)
\end{cases}
$$

Here, the problem no longer decomposes into $n$ separate single switch problems, since the switches are connected via the selection constraints. Moreover, the presence of the SELECTION constraints has the effect that it is no longer obvious whether the corresponding constraint matrix is totally unimodular. The main result of this section will be that this matrix is indeed still totally unimodular. Because of the integrality of the right hand sides, this will then imply that (PEN-LIN$^{\text{SEL}}$) can be solved efficiently by linear programming.

**Lemma 4.1.1.** The constraint matrix $M$ corresponding to the constraints (1),(2) and (3) in (PEN-LIN$^{\text{SEL}}$) is totally unimodular.

*Proof.* Let $M$ be the constraint matrix that models the constraints (1), (2), and (3) and let $R$ be any subset of its rows. The claim is proven by an induction over $T$, showing that there exists a partition $R_1 \dot\cup R_2 = R$ that satisfies:

(i) $\sum_{r\in R_1} m_{rj} - \sum_{r\in R_2} m_{rj} \in \{-1, 0, 1\}$ for all columns $j$ of $M$

(ii) if $R$ contains a row $r_1$ of constraint (2) for some $i_1 \in \{1, ..., n\}$ and $t \in \{1, ..., T\}$, and a row $r_2$ of constraint (3) for some $i_2 \in \{1, ..., n\}$ and the same $t$ such that $i_1 \neq i_2$, then $r_1$ and $r_2$ belong to different parts of the partition.

Actually, it suffices to show this for all sets $R$ satisfying the following condition:

($\star$) The set $R$ does not contain both the row of constraint (2) and the row of constraint (3) for the same pair $(i, t)$ of indices.

Indeed, the same statement then follows for all row sets $R$: for all pairs $(i, t)$ as in ($\star$), assign both corresponding rows to the same part, say, $R_1$. Clearly, the

resulting partition still satisfies (i) and (ii) then.

Note that the base case $T = 0$ for the induction is obvious, since then $M$ only consists of the single constraint (1) for $T = 0$ and it is assigned to the set $R_1$, while $R_2 = \emptyset$. Then, (i) is trivially satisfied. Since the constraints (2) and (3) do not exist, (ii) holds as well.

Now, let $\tilde{R}$ refer to the subset of rows in $R$ that correspond to constraints (1), (2), and (3) for $i = 1, ..., n$ and $t = 1, ..., T-1$. Let $\tilde{M}$ denote the restriction of $M$ to the rows in $\tilde{R}$. Observe that the assumption $(\star)$ carries over to $\tilde{R}$. Then, by the induction hypothesis, there exists a partition $\tilde{R} = \tilde{R}_1 \dot{\cup} \tilde{R}_2$ that satisfies (i), (ii) and $(\star)$.

For the induction step, an extension of the partition $\tilde{R}_1 \dot{\cup} \tilde{R}_2$ of $\tilde{R}$ to a partition $R_1 \dot{\cup} R_2$ of $R$ is described. Thus, define $R_1 := \tilde{R}_1$ and $R_2 := \tilde{R}_2$, so that it remains to assign the rows $R \setminus \tilde{R}$, which represent constraints (1), (2), and (3) for $i = 1, ..., n$ and $t = T$. By the induction hypothesis (ii), it follows that all rows that correspond to constraints (2) for $t = T - 1$ are assigned to one partition set, say $R_1$, and all rows corresponding to constraints (3) for $t = T - 1$ are assigned to the other set $R_2$. Next, assign the rows in $R \setminus \tilde{R}$ to the partition sets $R_1$ and $R_2$ as follows:

- If $R$ contains the row $\tilde{r}$ of constraint (1) for $t = T-1$, then $\tilde{r}$ is assigned to $R_2$, as anything else would generate a conflict between (i) and $(\star)$. Then, assign all rows in $R \setminus \tilde{R}$ of constraints (2) to $R_2$, all rows in $R \setminus \tilde{R}$ of constraints (3) to $R_1$ and the row of constraint (1) in $R \setminus \tilde{R}$ to $R_1$.

- Otherwise, assign all rows in $R \setminus \tilde{R}$ of constraints (2) to $R_1$, all rows in $R \setminus \tilde{R}$ of constraints (3) to $R_2$ and the row of constraint (1) in $R \setminus \tilde{R}$ to $R_2$.

It can be verified that, in both cases, the resulting partition $R_1 \dot{\cup} R_2$ satisfies (i),(ii) and $(\star)$. This concludes the proof, since condition (i) implies total unimodularity by Ghouila-Houri's criterion. □

Since the right-hand side of (PEN-LIN$^{\text{SEL}}$) is integer and all constraints in (PEN-LIN$^{\text{SEL}}$) other than (1) to (3) impose only upper or lower bounds, the binarity constraint for $x$ may be relaxed. This yields the following result.

**Theorem 4.1.2.** Problem (PEN-LIN$^{\text{SEL}}$) reduces to a linear program and, hence, can be solved in polynomial time.

**Remark 4.1.1.** Consider the constraint matrix in (PEN-LIN$^{\text{SEL}}$). Then including constraints which fix $x_t^{(i)}$ to some $b \in \{0,1\}$ for any switch $i \in \{1,...,n\}$ and any stage $t \in \{0,...,T\}$ does not destroy the total unimodularity of the resulting constraint matrix. Hence, the corresponding continuous relaxation is also integral.

The next example demonstrates the inductive construction of a partition of the rows that satisfies (i),(ii) and ($\star$) as described in the proof of Lemma 4.1.1.

**Example 4.1.1.** Consider an instance of (PEN-LIN$^{\text{SEL}}$) with $n = 3$ and $T = 2$. The matrix that corresponds to all constraints (1),(2) and (3) (without the columns for $u$ and $d$) is given by

| | $x_0^{(1)}$ | $x_0^{(2)}$ | $x_0^{(3)}$ | $x_1^{(1)}$ | $x_1^{(2)}$ | $x_1^{(3)}$ | $x_2^{(1)}$ | $x_2^{(2)}$ | $x_2^{(3)}$ |
|---|---|---|---|---|---|---|---|---|---|
| $r_1$ | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $r_2$ | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| $r_3$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| $r_4$ | -1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| $r_5$ | 0 | -1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| $r_6$ | 0 | 0 | -1 | 0 | 0 | 1 | 0 | 0 | 0 |
| $r_7$ | 1 | 0 | 0 | -1 | 0 | 0 | 0 | 0 | 0 |
| $r_8$ | 0 | 1 | 0 | 0 | -1 | 0 | 0 | 0 | 0 |
| $r_9$ | 0 | 0 | 1 | 0 | 0 | -1 | 0 | 0 | 0 |
| $r_{10}$ | 0 | 0 | 0 | -1 | 0 | 0 | 1 | 0 | 0 |
| $r_{11}$ | 0 | 0 | 0 | 0 | -1 | 0 | 0 | 1 | 0 |
| $r_{12}$ | 0 | 0 | 0 | 0 | 0 | -1 | 0 | 0 | 1 |
| $r_{13}$ | 0 | 0 | 0 | 1 | 0 | 0 | -1 | 0 | 0 |
| $r_{14}$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | -1 | 0 |
| $r_{15}$ | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | -1 |

Let $R = \{r_1, r_3, r_4, r_6, r_8, r_{11}, r_{13}, r_{15}\}$ be the given subset of the rows. To begin, consider the stage $t = 0$ as the base case. Since in this case, the matrix consists only of constraint (1) for $t = 0$, and $r_1 \in R$, row $r_1$ is assigned to the set $R_1$ (red) arbitrarily. The other set $R_2$ (blue) remains empty.

| | | | |
|---|---|---|---|
| $r_1$ | 1 | 1 | 1 |

Now to the next stage $t = 1$: Since $R$ contains the row $r_1$ for constraint (1) for stage $t = 0$, assign the rows $r_4$ and $r_6$ to the color red as well, while the row $r_8$ is assigned to the color blue. This yields:

| | | | | | | |
|---|---|---|---|---|---|---|
| $r_1$ | 1 | 1 | 1 | 0 | 0 | 0 |
| $r_4$ | -1 | 0 | 0 | 1 | 0 | 0 |
| $r_6$ | 0 | 0 | -1 | 0 | 0 | 1 |
| $r_8$ | 0 | 1 | 0 | 0 | -1 | 0 |

Following the inductive argument for $t = 2$, observe that $R$ does not contain the row $r_2$ of constraint (1) for the stage $t = 2$. Thus, assign row $r_{11}$ to the color red and rows $r_3$,$r_{13}$ and $r_{15}$ to the color blue.

$$\left(\begin{array}{c|ccccccccc}
r_1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
r_3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
\hline
r_4 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
r_6 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 \\
\hline
r_8 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
\hline
r_{11} & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 \\
\hline
r_{13} & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\
r_{15} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1
\end{array}\right)$$

This section closes by collecting some interesting remarks and observations that are implied by the previously obtained results.

**Remark 4.1.2.** Observe that the model (PEN-LIN$^{\text{SEL}}$) can easily be adapted to involve variation constraints by additionally including the constraints

$$\sum_{i=1}^{n} u_t^{(i)} + d_t^{(i)} \leq S_t \qquad t = 1, ..., T \qquad \text{for time-wise variation, or}$$

$$\sum_{t=1}^{T} u_t^{(i)} + d_t^{(i)} \leq S_i \qquad i = 1, ..., n \qquad \text{for switch-wise variation, or}$$

$$\sum_{i=1}^{n} \sum_{t=1}^{T} u_t^{(i)} + d_t^{(i)} \leq S \qquad \text{for total variation.}$$

However, the total unimodularity of Lemma 4.1.1 is immediately lost, once a variation constraint is added, as the next example shows.

**Example 4.1.2.** Consider (PEN-LIN$^{\text{SEL}}$) for $n = 2$ and $T = 1$. Now, include a time-wise variation constraint as proposed in the previous remark. The matrix representing constraints (1), (2), and (3) as well as the constraint for time-wise variation reads

$$\left(\begin{array}{cccc|cccc}
x_0^{(1)} & x_0^{(2)} & x_1^{(1)} & x_1^{(2)} & u_1^{(1)} & u_1^{(2)} & d_1^{(1)} & d_1^{(2)} \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
\hline
-1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\
0 & -1 & 0 & 1 & 0 & -1 & 0 & 0 \\
1 & 0 & -1 & 0 & 0 & 0 & -1 & 0 \\
0 & 1 & 0 & -1 & 0 & 0 & 0 & -1 \\
\hline
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1
\end{array}\right).$$

Then, the submatrix built from rows $1, 3, 6, 7$ and columns $1, 2, 5, 8$ has

$$\det \begin{pmatrix} 1 & 1 & 0 & 0 \\ -1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 1 \end{pmatrix} = 2$$

and is thus not totally unimodular.

Not only is the total unimodularity lost, but more importantly, the integrality of the continuous relaxation is lost as well:

**Example 4.1.3.** Consider the continuous relaxation of (PEN-LIN$^{\text{SEL}}$) with $T = 1$, $n = 2$ and include a time-wise variation constraint as described in Remark 4.1.2. Let the bound on the time-wise variation be given by $S = 1 \in \mathbb{N}$ and let the selection bounds be given by $K_0 := K_1 := 1$. Define the objective by

$$\begin{aligned} c_0^{(1)} &= 1 \quad c_1^{(1)} = 0 \\ c_0^{(2)} &= 0 \quad c_1^{(2)} = 1 \quad . \end{aligned}$$

It is easy to see that the best objective value which can be achieved by a feasible *integer* solution is given by 1, since the bound on the variation and the selection constraints prevent the choice of any integer solution with a larger objective value. However, consider the solution

$$\begin{aligned} x_0^{(1)} &= \tfrac{1}{2} \quad x_1^{(1)} = 0 \\ x_0^{(2)} &= \tfrac{1}{2} \quad x_1^{(2)} = 1 \ , \end{aligned}$$

which is feasible for the continuous relaxation, since it has a time-wise variation of 1 and it satisfies the selection constraint for each stage. Since this solution has an objective value of $3/2$, the continuous relaxation cannot be integral.

Similarly, including a switch-wise or total variation constraint into the continuous relaxation of (PEN-LIN$^{\text{SEL}}$) as in Remark 4.1.2 destroys the integrality, even if there is only a single switch. Recall that for $n = 1$, switch-wise and total variation are equivalent so that the model (PEN-LIN$^{\text{SEL}}$) with a switch-wise/total variation constraint as in Remark 4.1.2 is equivalent to the model obtained by the intersection of ($\text{P}^{(1)}$-S) with the hyperplane fixing the former variable $S$ to the given bound. The corresponding continuous relaxation was already shown to be non-integral in Example 3.3.2.

As Examples 4.1.2 and 4.1.3 show, the inclusion of the time-wise variation constraint not only destroys the total unimodularity, but also the integrality of the corresponding LP-relaxation. However, for related, but slightly different bounds on the auxiliary variables $u$ and $d$, the total unimodularity of the corresponding constraint matrix is maintained. More precisely, if there is an individual (integer) upper bound $U_t$ on the sum $\sum_{i=1}^{n} u_t^{(i)}$ and/or an (integer) upper bound $D_t$ on the sum $\sum_{i=1}^{n} d_t^{(i)}$ for each stage $t = 0, ..., T$, then incorporating these particular variation constraints into the model (PEN-LIN$^{\text{SEL}}$) yields a totally unimodular constraint matrix. Of course, this observation is interesting in itself, but mainly, it will help to settle the complexity of (BND) with time-wise variation and a selection constraint as in (SEL), see Theorem 4.2.1 later on.

**Lemma 4.1.2.** The integer problem

$$\begin{aligned} \max \quad & \sum_{i=1}^{n}\sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} - \sum_{i=1}^{n}\sum_{t=1}^{T} \left(\mu_t^{(i)} u_t^{(i)} - \lambda_t^{(i)} d_t^{(i)}\right) \\ \text{s.t.} \quad & (1), (2), (3), (4), (5) \\ & \sum_{i=1}^{n} u_t^{(i)} \leq U_t, \ \sum_{i=1}^{n} d_t^{(i)} \leq D_t \qquad t = 1, ..., T \end{aligned}$$

reduces to a linear problem and hence, can be solved in polynomial time.

*Proof.* Indeed, the partition obtained from the proof of Lemma 4.1.1 can easily be extended to work for subsets $R$ of the rows that may also include the rows of these additional *new* variation constraints. Recall that the characteristic (ii) leads to a partition so that all constraints (2) for $i \in \{1, ..., n\}$ and for some fixed $t$ are assigned to the same partition set, say $R_1$, while all constraints (3) for $i \in \{1, ..., n\}$ and the same $t$ are assigned to the other set $R_2$. Note that each row corresponding to a constraint (2) or (3) contains exactly one non-zero entry $-1$ in the column symbolizing either variable $u_t^{(i)}$ or $d_t^{(i)}$. Now, assign the row corresponding to constraint $\sum_{i=1}^{n} u_t^{(i)} \leq U_t$ to the same partition set to which all constraints (2) in $R$ for the stage $t$ have been assigned. Conversely, assign the row of constraint $\sum_{i=1}^{n} d_t^{(i)} \leq D_t$ to the other partition set, to which all rows corresponding to constraints (3) in $R$ for stage $t$ have been assigned. It is quickly verified that this yields a partition for any given subset of rows that satisfies Ghouila-Houri's criterion. □

The next example continues Example 4.1.1 and extends the partition onto the additional constraints for this *new* time-wise variation.

**Example 4.1.4.** Consider an instance as in Lemma 4.1.2 for $n = 3$, $T = 2$ and let the row set $R' = R \cup \{r_{16}, r_{17}, r_{18}, r_{19}\}$ be given, where $R$ is the row set considered in Example 4.1.1. Now, starting with the row partition that was determined in Example 4.1.1, this partition is extended onto the additional rows numbered $r_{16}, r_{17}, r_{18}$ and $r_{19}$ expressing the newly defined variation constraints from Lemma 4.1.2. The assignment of these additional rows to the partition sets is indicated through the respective row colors in Figure 4.1 below.

| | | | | | | | | | | $u_1^{(1)}$ | $u_1^{(2)}$ | $u_1^{(3)}$ | $d_1^{(1)}$ | $d_1^{(2)}$ | $d_1^{(3)}$ | $u_2^{(1)}$ | $u_2^{(2)}$ | $u_2^{(3)}$ | $d_2^{(1)}$ | $d_2^{(2)}$ | $d_2^{(3)}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $r_1$ | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $r_3$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $r_4$ | -1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | -1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $r_6$ | 0 | 0 | -1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | -1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $r_8$ | 0 | 1 | 0 | 0 | -1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $r_{11}$ | 0 | 0 | 0 | 0 | -1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -1 | 0 | 0 | 0 | 0 | 0 |
| $r_{13}$ | 0 | 0 | 0 | 1 | 0 | 0 | -1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -1 | 0 |
| $r_{15}$ | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | -1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -1 |
| $r_{16}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $r_{17}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $r_{18}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| $r_{19}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Figure 4.1: Assignment of additional rows in Example 4.1.4

## 4.2 Bounded Variation

This section is dedicated to the problem (BND) with a hard bound on the variation and an additional selection constraint as in (SEL) for each stage. As it will turn out, the complexity of the respective problem variant drastically changes depending on the type of variation considered. Each following section will deal with one of the problem variants and will settle its complexity. The first section will investigate the problem with time-wise variation. Previously, it was already shown that there is no obvious approach to this problem variant with an extended linear reformulation, since the corresponding continuous relaxation is not integral (see Example 4.1.3). Nonetheless, the problem will be proven to be efficiently solvable by means of a compact LP formulation as well as by a polynomial reduction to a min cost flow problem. This reduction will yield the possibility to solve the problem algorithmically by network flow algorithms. In the subsequent Section 4.2.2, the problem with a bound on the switch-wise variation will be studied. It will be shown that this problem almost immediately becomes much harder. In fact, the problem will be proven to become strongly $\mathcal{NP}$-hard, even if the bound on the (switch-wise) variation is uniformly equal to two and all selection bounds are uniformly equal to one. However, as soon as either the number of stages $T$, or the number of switches $n$ is considered a fixed constant, the problem will turn out to be efficiently solvable by dynamic programming. Furthermore, the problem is shown to be tractable if the bound on the variation is uniformly one.

In the third section, the problem will be studied with a bound on the total variation. While the unbounded problem version without constraints on the feasible sets will be shown to be polynomially solvable by extension of the dynamic programming scheme for a single switch from the previous chapter, the general complexity of

the problem with an additional selection constraint will remain unknown. However, as soon as the number of stages, the number switches, or the selection bounds $K_t$, $t = 0, ..., T$ are fixed, the problem will be proven to be tractable.

Finally, the last section will briefly explore some interesting related problems and generalizations.

### 4.2.1 Time-Wise Variation

This section will prove the tractability of the problem (BND) with bounded time-wise variation and an additional selection constraint as in (SEL) at each stage. Observe that the unconstrained problem version can be solved by linear programming, since the inclusion of a time-wise variation constraint as described in Remark 4.1.2 in (PEN-LIN) clearly yields a totally unimodular constraint matrix according to Ghouila-Houri's criterion. Hence, this section will only consider the version with an additional selection constraint which reads as follows:

$$(\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{tw}}) \qquad \begin{cases} \max & \displaystyle\sum_{i=1}^{n}\sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} & \\ \text{s.t.} & \displaystyle\sum_{i=1}^{n} |x_{t+1}^{(i)} - x_t^{(i)}| \leq S_t & t = 0, ..., T-1 \\ & \displaystyle\sum_{i=1}^{n} x_t^{(i)} = K_t & t = 0, ..., T \\ & x_t \in \{0,1\}^n & i = 1, ..., n,\ t = 0, ..., T \end{cases}$$

As already mentioned in the introduction, $(\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{tw}})$ is tractable. However, it simplifies the argument to assume that all selection bounds are uniform. Indeed, this may be assumed without loss of generality:

**Lemma 4.2.1.** Without loss of generality, one may assume that all selection bounds are uniform, i.e., that $K_t := K$ for all $t = 0, ..., T$.

*Proof.* Let an instance of $(\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{tw}})$ with non-uniform be given. By introducing additional auxiliary switches so that their states are fixed, and adjusting the selection bounds accordingly, this instance can be reduced to an instance with uniform selection bounds. To this end, define

$$K_{max} := \max_{t=0,...,T} K_t, \quad K_{min} := \min_{t=0,...,T} K_t \qquad \text{and} \qquad m := K_{max} - K_{min}\ .$$

Now, define a new instance of $(\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{tw}})$ by extending the given instance through introducing further switches numbered $n+1, ..., n+m$ and enforcing a uniform

selection bound of $K := K_{max}$ for all of the stages. The idea behind these new switches is to fix their state in a particular way, therefore the objective function is defined as follows

$$c_t^{(n+i)} := \begin{cases} C & \text{if } i \leq K_{max} - K_t \\ -C & \text{otherwise} \end{cases} \qquad \text{for } i = 1, ..., m \ ,$$

where $C$ is a constant large enough, for example $C := \sum_{i=1}^{n} \sum_{t=0}^{T} |c_t^{(i)}| + 1$. Then, for any optimal solution of this new instance, exactly $K_{max} - K^{(t)}$ of the auxiliary switches must be active at stage $t$. Consequently, exactly $K_t$ of the original switches $1, ..., n$ must be active at any stage $t$. Finally, enlarge the original bounds $S_t$ on the time-wise variation by $|K_t - K_{t-1}|$ for $t = 1, ..., T$. Since the latter term equals the increase in the time-wise variation caused by the auxiliary switches, the former bound $S_t$ applies to the original switches $1, ..., n$. This reduction is clearly of polynomial size, as $m \leq n$. Hence follows the claim. □

This last observation is helpful to settle the complexity of $(\text{BND}_{\text{tw}}^{\text{SEL}})$.

**Theorem 4.2.1.** The problem $(\text{BND}_{\text{tw}}^{\text{SEL}})$ reduces to a linear problem and, consequently, can be solved in polynomial time.

*Proof.* Consider an instance of $(\text{BND}_{\text{tw}}^{\text{SEL}})$ where, according to Lemma 4.2.1, the selection bounds may be assumed to be uniform without loss of generality. Then, this selection constraint enforces that the number of switches that are active must be equal to $K$ for each stage and, more importantly, it enforces that the number of deactivations must equal the number of activations for each stage. Note that, as a result, the time-wise variation between two consecutive stages will always be even. Therefore, in order to ensure that the time-wise variation between two consecutive stages $t-1$ and $t$ does not exceed the given bound $S_t$, it actually suffices to impose an upper bound of $\lfloor S_t/2 \rfloor$ on the number of deactivations or the number of activations, since the time-wise variation between those stages is given by twice the number of deactivations (or activations, respectively) due to the selection constraint. Thus, the claim follows from Lemma 4.1.2 with $U_t := \lfloor S_t/2 \rfloor$ and/or $D_t := \lfloor S_t/2 \rfloor$. □

However, there also exists an interesting connection to combinatorial optimization in the context of network flows, which once again shows the tractability of the problem $(\text{BND}_{\text{tw}}^{\text{SEL}})$.

**Theorem 4.2.2.** The problem $(\text{BND}_{\text{tw}}^{\text{SEL}})$ polynomially reduces to a min cost flow problem. Therefore, it can be solved in polynomial time.

*Proof.* According to Lemma 4.2.1, one may assume that all selection bounds are uniform, i.e., $K_t := K$ for all $t = 0, ..., T$. So, let $I = (c, (S_0, ..., S_T), K)$ be such an instance of $(\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{tw}})$.

In the following, a suitable network $(V, A)$ for a min cost flow problem is defined: First, introduce a source $q$ with supply $b_q = K$ and a sink $r$ with demand $b_r = -K$. For each switch $i = 1, ..., n$ and each time point $t = 0, ..., T$, let there be a node $v_t^{(i)}$ as well as a corresponding copy node $\bar{v}_t^{(i)}$. These nodes are connected via the arc $(v_t^{(i)}, \bar{v}_t^{(i)})$. A positive flow on such an arc then represents the respective switch being active at stage $t$. The source is connected with each the nodes $v_0^{(1)}, ..., v_0^{(n)}$, while each of the copies $\bar{v}_T^{(1)}, ..., \bar{v}_T^{(n)}$ is connected with the sink. The bound on the variation is incorporated by arcs $(\alpha_t, \beta_t)$ for $t = 1, ..., T$ that have a capacity limit of $\lfloor S_t/2 \rfloor$. All of the copies $\bar{v}_t^{(1)}, ..., \bar{v}_t^{(n)}$ are connected with $\alpha_t$, while the nodes $\beta_t$ are connected to all of the nodes $v_{t+1}^{(1)}, ..., v_{t+1}^{(n)}$ of the next stage. The idea is to model the problem $(\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{tw}})$ as a $q, r$-flow of strength $K$. More formally, the set of nodes is defined by

$$\begin{aligned} V :=& \{v_t^{(i)} \mid i = 1, ..., n,\ t = 0, ..., T\} \cup \{\bar{v}_t^{(i)} \mid i = 1, ..., n,\ t = 1, ..., T\} \cup \\ & \{\{\alpha_t, \beta_t \mid t = 1, ..., T\} \cup \{q, r\} \end{aligned}$$

and the set of arcs is defined by

$$\begin{aligned} A :=& \{(v_t^{(i)}, \bar{v}_t^{(i)}) \mid i = 1, ..., n,\ t = 0, ..., T\} \cup \\ & \{(\bar{v}_t^{(1)}, \alpha_{t+1}), \ldots, (\bar{v}_t^{(n)}, \alpha_{t+1}) \mid t = 0, ..., T-1\} \cup \\ & \{(\beta_t, v_t^{(1)}), \ldots, (\beta_t, v_t^{(n)}), (\alpha_t, \beta_t) \mid t = 1, ..., T\} \cup \\ & \{(q, v_0^{(1)})), ..., (q, v_0^{(n)})\} \cup \{(\bar{v}_T^{(1)}, r), ..., (\bar{v}_T^{(n)}, r)\}\ , \end{aligned}$$

The capacities $u : A \to \mathbb{N}$ and the costs $c : A \to \mathbb{R}$ of the arcs are given by:

$$u(a) := \begin{cases} 1 & \text{if } a = (v_t^{(i)}, \bar{v}_t^{(i)}) \\ \lfloor S_t/2 \rfloor & \text{if } a = (\alpha_t, \beta_t) \\ K & \text{otherwise} \end{cases} \qquad c(a) := \begin{cases} -c_t^{(i)} & \text{if } a = (v_t^{(i)}, \bar{v}_t^{(i)}) \\ 0 & \text{otherwise} \end{cases} .$$

This construction is clearly polynomial in its size. Figure 4.2 below illustrates the resulting network graph for an instance of $(\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{tw}})$ with $n = 3$ and $T = 2$.

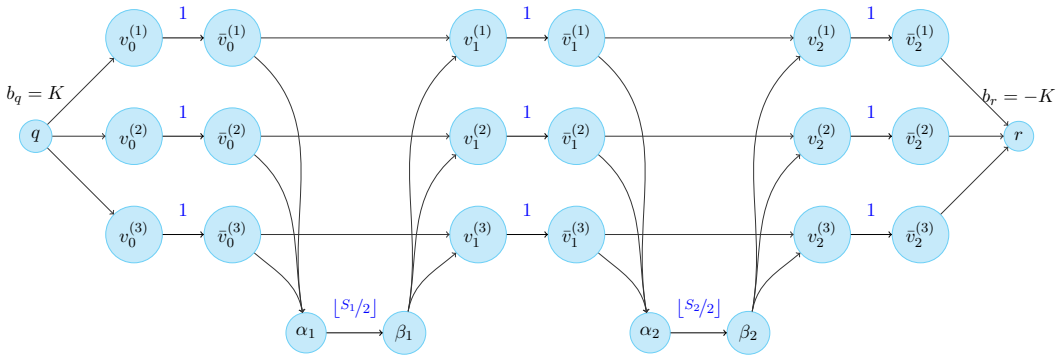



Figure 4.2: Sketch of the network for $n = 3, T = 2$, where arc capacities (that are not $K$) are represented in blue and arc costs are not depicted

Since all arc capacities of the network are integer, there always exists an integer optimal flow for the min cost flow instance. Therefore, it suffices to show that feasible solutions of $(\mathrm{BND}_{\mathrm{tw}}^{\mathrm{SEL}})$ bijectively correspond to integer feasible solutions of the min cost flow problem, while attaining the respective (negated) objective value.

To this end, let $x$ be a feasible solution of $(\mathrm{BND}_{\mathrm{tw}}^{\mathrm{SEL}})$. Now, define a feasible integer flow $f$ as follows:

$$f_{(v_t^{(i)}, \bar{v}_t^{(i)})} := x_t^{(i)} \qquad i = 1, ..., n,\ t = 0, ..., T$$
$$f_{(\alpha_t, \beta_t)} := \sum_{i=1}^{n} |x_t^{(i)} - x_{t+1}^{(i)}| \qquad t = 1, ...., T\ .$$

With these definitions, the values of the (integer) flow on all remaining edges are already determined. The flow must emanate from the source to the nodes $v_0^{(i)}$ which correspond to the switches selected in the first stage, while the flow leaving the nodes $\bar{v}_T^{(i)}$ for the switches selected in the last stage must be absorbed by the target:

$$f_{(q, v_t^{(i)})} := x_0^{(i)},\ f_{(\bar{v}_T^{(i)}, r)} := x_T^{(i)} \qquad i = 1, ...., n\ .$$

Clearly, the *direct* arc $(\bar{v}_t^{(i)}, v_{t+1}^{(i)})$ is more convenient than the detour along the arcs $(\bar{v}_t^{(i)}, \alpha_{t+1}), (\alpha_{t+1}, \beta_{t+1}), (\beta_{t+1}, v_{t+1}^{(i)})$, since both have cost zero, but the latter unnecessarily uses up capacity on the arc $(\alpha_{t+1}, \beta_{t+1})$. Therefore, the flow may be assumed to always use these *direct* arcs, if possible. As a result, define

$$f_{(\bar{v}_t^{(i)}, v_{t+1}^{(i)})} := 1 \quad \text{if and only if } x_t^{(i)} = 1,\ x_{t+1}^{(i)} = 1 \qquad i = 1, ...n, \quad t = 0, ..., T-1\ ,$$
$$f_{(\beta_t, v_t^{(i)})} := 1 \quad \text{if and only if } x_t^{(i)} = 0,\ x_{t+1}^{(i)} = 1 \qquad i = 1, ...n, \quad t = 1, ..., T\ ,$$
$$f_{(\bar{v}_t^{(i)}, \alpha_{t+1})} := 1 \quad \text{if and only if } x_t^{(i)} = 1,\ x_{t+1}^{(i)} = 0 \qquad i = 1, ...n, \quad t = 0, ..., T-1\ .$$

The fact that this flow has value $K$ and that it is flow preserving is ensured by the selection constraint at each stage. It is easily verified that this flow is feasible, as it is flow-preserving, it does not exceed the capacity limits and it satisfies the supply and demand constraints. The definition of the costs in the network then yields

$$-\sum_{i=1}^{n}\sum_{t=0}^{T-1} c((v_t^{(i)}, v_{t+1}^{(i)})) = \sum_{i=1}^{n}\sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} \ .$$

Conversely, given any feasible integer flow $f$, define a feasible solution $x$ for ($\text{BND}_{\text{tw}}^{\text{SEL}}$) by $x_t^{(i)} := f_{(v_t^{(i)}, \bar{v}_t^{(i)})}$ for all $t = 0, ..., T$. Since the flow is preserved, it has strength $K$ and it can be at most one on each arc $(v_t^{(i)}, \bar{v}_t^{(i)})$, the selection constraint is satisfied at each stage. In particular, it is not possible to select a node/switch more than once for stage $t$. Furthermore, since the flow on the arcs $(\alpha_t, \beta_t)$ is limited by the capacity $\lfloor S_t/2 \rfloor$ for each $t = 1, ..., T$, and since the flow on this arc must be at least as large as the variation $\sum_{i=1}^{n} |x_t^{(i)} - x_{t+1}^{(i)}|$ between stages $t$ and $t+1$, the solution $x$ is feasible for the given instance of ($\text{BND}_{\text{tw}}^{\text{SEL}}$). The fact that the (negated) objective values agree is easy to see. This concludes the proof. □

**Remark 4.2.1.** With suitable adjustments to the network defined in the proof of Theorem 4.2.2, it is possible to directly incorporate non-uniform selection bounds into the construction. The key element lies in the introduction of further sources and sinks. More precisely, define $T$ additional nodes $q_t$ for $t = 1, ..., T$ with a demand/supply of $b_{q_t} := K_{t+1} - K_t$. The sign of $b_{q_t}$ indicates whether the node is a source (positive) or a sink (negative). Each source is then included into the existing network with the arcs $(q_t, \beta_t)$, while each sink is included into the network with the arcs $(\alpha_t, q_t)$. The supply and demand of the source and sink are redefined as $b_q := K_1$ and $b_r := -K_T$, the capacities on the arcs $(\alpha_t, \beta_t)$ are redefined as $c((\alpha_t, \beta_t)) := \lfloor \frac{S_t - |K_{t-1} - K_t|}{2} \rfloor$ in order to correctly take the rise or decline in the selection bounds into account. Figure 4.3 below shows the necessary adjustments in the network graph depicted in Figure 4.2 for $K_0 = 2, K_1 = 1$ and $K_2 = 3$.

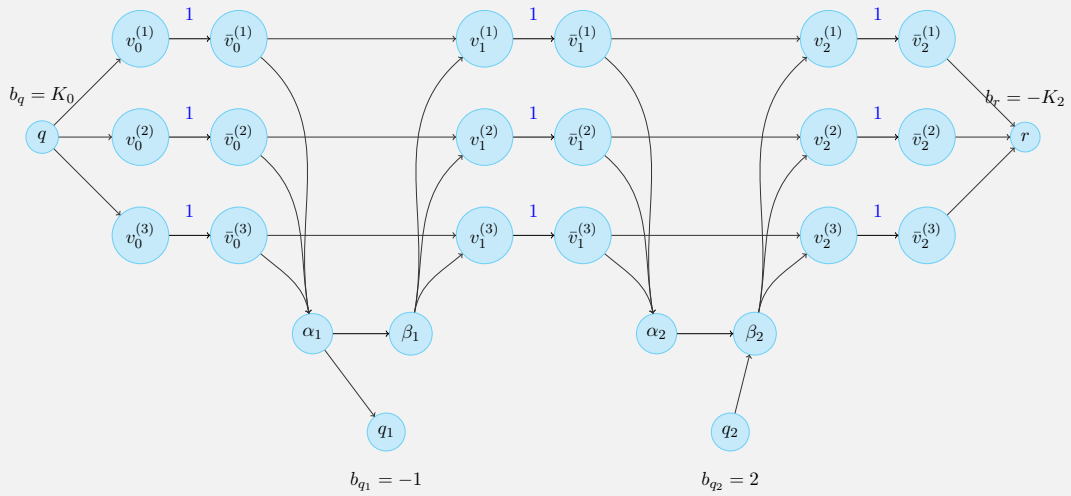



Figure 4.3: Sketch of the flow network for $n = 3, T = 2$ for the general problem ($\text{BND}_{\text{tw}}^{\text{SEL}}$) with non-uniform selection bounds

The problems studied in the previous Chapter 3, as well as in the preceding Section 4.1, were mainly investigated from a polyhedral perspective. This was justified by the fact that these problems quickly turned out to be tractable. However, as soon as there is more than one switch and an additional combinatorial constraint, such as (SEL), as is the case in this chapter, the complexity of the resulting problem versions is less obvious. The focus of this chapter therefore primarily lies on settling the complexity of the respective problem versions. If one of the problem versions turns out to be tractable, as ($\text{BND}_{\text{tw}}^{\text{SEL}}$) here, then one might contemplate a further polyhedral study. Recall that the intuitive candidate for an extended formulation of ($\text{BND}_{\text{tw}}^{\text{SEL}}$) as considered in Example 4.1.3 did not work out. However, as was argued for Theorem 4.2.1, there exists a compact extended formulation which may be continuously relaxed as described in Lemma 4.1.2. Actually, Theorem 4.2.2 also yields a polyhedral result as a by-product:

**Remark 4.2.2.** Recall that a min cost flow problem can be modeled by a linear program with a totally unimodular constraint matrix. Consequently, if the right hand side is integer, this LP always has an integer optimal solution. The previous reduction of ($\text{BND}_{\text{tw}}^{\text{SEL}}$) to a min cost flow problem relied on the fact that it produces an integer right hand side in the LP model corresponding to the flow problem. Therefore, the latter is an extended formulation for ($\text{BND}_{\text{tw}}^{\text{SEL}}$). In particular, this once again shows that ($\text{BND}_{\text{tw}}^{\text{SEL}}$) can be solved by linear programming.

A compact extended LP formulation of ($\text{BND}_{\text{tw}}^{\text{SEL}}$) can be used to obtain a linear inequality description of the convex hull of feasible solutions of ($\text{BND}_{\text{tw}}^{\text{SEL}}$) in the original space of only the $x$-variables, e.g., by projection. However, it is likely that the result of the projection is exponentially large and does not offer much additional insight. Moreover, the process of the projection itself is likely to be very complex and convoluted. Therefore, this thesis refrains from polyhedral study beyond this point.

**Remark 4.2.3.** As in the previous chapter, it might be the case that for some stage $t$, the solution $x_t$ is fixed to a particular element in $X_t$, so that the selection of switches at this stage $t$ is predetermined. For the penalty problem in Section 4.1, it sufficed to include the respective constraints ensuring the desired fixation in the LP formulation, since the latter had a totally unimodular constraint matrix, see Remark 4.1.1. The same approach works out here:
Consider ($\text{BND}_{\text{tw}}^{\text{SEL}}$) and let there be some additional fixation constraint $x_t^{(i)} = b$ for some $b \in \{0,1\}$ and $i \in \{1,...,n\}$, $t \in \{0,...,T\}$. Then, the constraint can either be directly included in the corresponding LP formulation described in Lemma 4.1.2, or, it can be included in the form of the constraint $f_{(v_t^{(i)},\bar{v}_t^{(i)})} = b$ in the corresponding min cost flow problem. The latter is equivalent to adapting the lower or upper bound for the flow on the arc $(v_t^{(i)}, \bar{v}_t^{(i)})$ as follows: either raise its lower bound from zero to one if $b = 1$, otherwise lower its upper bound from one to zero.

In the next section, the problem (BND) with switch-wise variation will be studied. The unconstrained case where $X_t = \{0,1\}^n$ for all $t = 0,...,T$ will immediately decompose into $n$ separate single switch problems as considered in Section 3.3, and is therefore tractable. However, this will change if a slightly more complicated combinatorial set $X_t$ is considered. In fact, the problem will turn out to become strongly $\mathcal{NP}$-hard if a selection constraint as in (SEL) is included.

### 4.2.2 Switch-Wise Variation

This section is dedicated to (BND) with switch-wise variation and a selection constraint as in (SEL). Indeed, the unconstrained problem (BND) with switch-wise variation is already completely covered with the results from Section 3.3, as this problem decomposes along the indices $i = 1,...,n$ into $n$ disjoint single switch problems ($\text{BND}_{\text{var}}^{(1)}$). Consequently, a compact extended formulation is given by the cartesian product of all corresponding single switch extended formulations (EXT-BND). Analogously, the convex hull of all feasible solutions is completely described by the cartesian product of all corresponding alternating inequalities and the trivial inequalities. The separation problem can be solved by solving the separation problem for each of the single-switch problems in time $\mathcal{O}(n \cdot T)$. Algorithmically, the problem can be solved by application of the merging algorithm 2 for each of the switches $i = 1,....,n$

in time $\mathcal{O}(n \cdot T \log T)$, or by application of the dynamic programming scheme for each of the switches $i = 1, ...., n$ in time $\mathcal{O}(n \cdot S \cdot T)$.

However, this decomposition only happens if the switches are entirely independent of one another, which is no longer the case if additional selection constraints as in (SEL) are present. The remainder of this section will thus deal with the problem

$$(\mathrm{BND}_{\mathrm{sw}}^{\mathrm{SEL}}) \qquad \begin{cases} \max & \displaystyle\sum_{i=1}^{n}\sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} & \\ \text{s.t.} & \displaystyle\sum_{t=0}^{T-1} |x_t^{(i)} - x_{t+1}^{(i)}| \leq S_i & i = 1, ..., n \\ & \displaystyle\sum_{i=1}^{n} x_t^{(i)} = K_t & t = 0, ..., T \\ & x_t \in \{0,1\}^n & t = 0, ..., T\ . \end{cases}$$

As was argued before, dropping the selection constraints $\sum_{i=1}^{n} x_t^{(i)} = K_t$, $t = 0, ..., T$, from $(\mathrm{BND}_{\mathrm{sw}}^{\mathrm{SEL}})$ yields a tractable problem. Although there are a few tractable special cases which will be presented later, generally, the inclusion of the selection constraints renders the problem $(\mathrm{BND}_{\mathrm{sw}}^{\mathrm{SEL}})$ strongly $\mathcal{NP}$-hard. Actually, the problem remains hard, even if $K_t = 1$ for all $t = 0, ..., T$ and $S_i = 2$ for all $i = 1, ..., n$, i.e., if at any stage at most one switch may be selected and every switch may have a variation of at most two. This is the main result of this section, it will be proven by a reduction from the following decision problem discussed in [BDMW13]:

**Problem 4.2.1.** Interval Selection

**Input:** Consider a scheduling problem with $m$ machines and $n$ jobs.
A job consists of $m$ open intervals on the real line, each of the intervals is associated with exactly one machine and each machine has exactly one interval per job. To schedule a job, exactly one of its intervals has to be selected. To schedule several jobs, no two selected intervals on the same machine must intersect.

**Task:** Decide whether it is possible to schedule all jobs.

With a reduction from $(\leq 3, 3)$-SAT – which is a special case of the satisfiability problem in which every clause is restricted to have no more than three literals and each variable appears in the formula at most three times, once as a negative literal and at most twice as a positive literal – the authors of [BDMW13] prove the following result:

**Theorem 4.2.3.** Interval Selection is $\mathcal{NP}$-complete, even if the number of machines is fixed to three and all intervals have the same length.

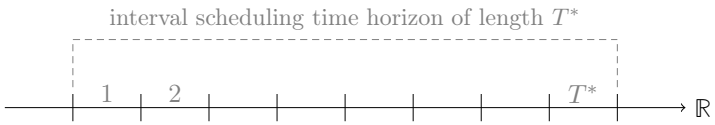
Through sharp observation of the proof presented in [BDMW13], it becomes apparent that the transformation of an instance of $(\leq 3, 3)$-SAT to an instance of Interval Selection produces unit intervals of length two and a time horizon big enough to contain $2s + 1$ of these intervals that do not intersect. Here, $s$ denotes the number of Boolean variables in the $(\leq 3, 3)$-SAT instance. Consequently, a time horizon of length $T^* := (2s+1) \cdot 2$ is sufficient. As a result, it is possible to strengthen the previous theorem as follows:

**Theorem 4.2.4.** Interval Selection is $\mathcal{NP}$-complete, even if the number of machines is fixed to three, all intervals have integer endpoints, and each interval has length two.

Theorem 4.2.4 then implies the $\mathcal{NP}$-hardness of $(\text{BND}_{\text{sw}}^{\text{SEL}})$.

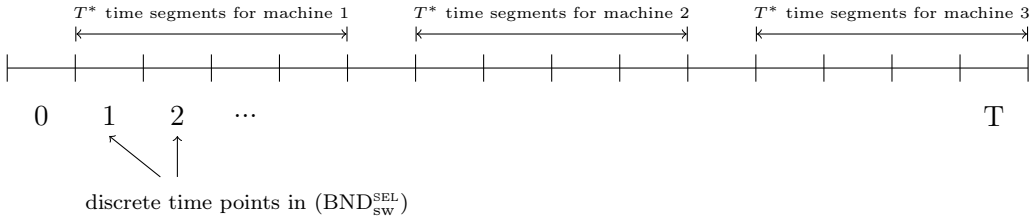
**Theorem 4.2.5.** Problem $(\text{BND}_{\text{sw}}^{\text{SEL}})$ is $\mathcal{NP}$-hard in the strong sense, even if $K_t = 1$ for all $t = 0, ..., T$ and $S_i = 2$ for all $i = 1, ..., n$.

*Proof.* Let an instance of Interval Selection as in Theorem 4.2.4 be given, consisting of $n$ jobs, $m = 3$ machines, and intervals $I_{ij}$ having length two and integer endpoints for each machine $i = 1, \ldots, n$ and $j = 1, 2, 3$. Let $T^*$ denote the smallest length of the time horizon on the real line containing all these intervals.




Then, divide this time horizon into $T^*$ segments of length one and observe that $T^*$ is bounded by $6n$ without loss of generality, as each interval has length two. The intervals $I_{ij}$ can thus be interpreted as subsets of $\{1, ..., T^*\}$.

Now, define an instance of $(\text{BND}_{\text{sw}}^{\text{SEL}})$. Let this instance have $3(T^* + 1)$ stages. This can be understood as concatenating $m = 3$ copies of the time horizon obtained from Interval Scheduling horizontally and inserting an additional time point before each of the copies. The idea behind this is that the scheduling decisions on the respective machines are modeled within the respective copies of the discrete time horizon. Recall that the time horizon in the instance of $(\text{BND}_{\text{sw}}^{\text{SEL}})$ begins with the stage $t = 0$, therefore define $T := 3(T^* + 1) - 1$.




For each of job $i \in \{1, ..., n\}$, define a switch $i$ as well as $T + 1$ dummy switches $i \in \{n+1, ..., n+1+T\}$. The variation of each switch is uniformly bounded by two, i.e., $S_i = 2$ for $i = 1, ..., n + 1 + T$. It remains to define the objective coefficients. For $i = 1, ..., n$, the coefficients $c_t^{(i)}$ will serve to model the intervals $I_{ij}$. More precisely, for $i \in \{1, ..., n\}$ define

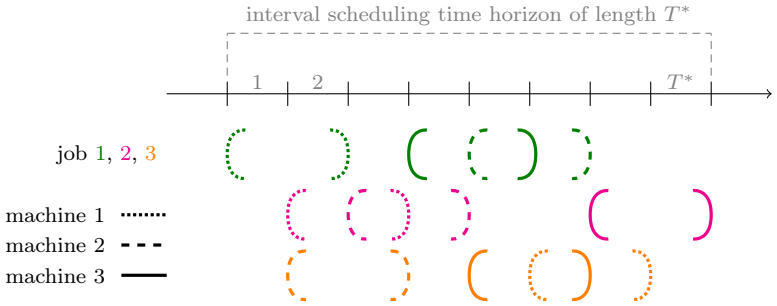
$$c_t^{(i)} = \begin{cases} 1, & \text{if } t \in \bigcup_{j=1,2,3} \bar{I}_{ij} \\ -3 & \text{otherwise,} \end{cases}$$

where $\bar{I}_{i1} := I_{i1}$ and $\bar{I}_{ij} := I_{ij} + (j-1)(T^* + 1)$ for $j = 2, 3$ is the interval $I_{ij}$ shifted to the $j$-th block. For the dummy switches, all objective coefficients are defined as zero, i.e., $c_t^{(i)} = 0$ for all $t = 0, ..., T$ and $i = n + 1, ..., n + 1 + T$. Finally, set the selection bound time-wise to one, i.e., $K_t := 1$ for all $t = 0, ..., T$.

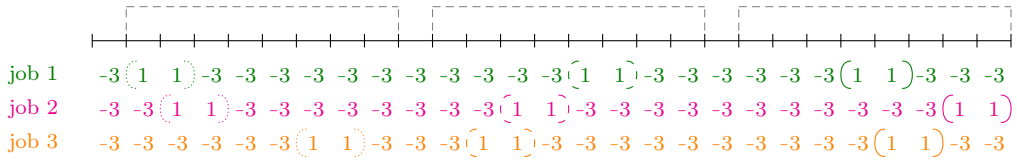
Observe that the dummy switches ensure that the instance of Problem ($\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{SW}}$) has an optimal value of at least zero. Indeed, the solution in which the first $n$ switches are inactive over all periods and each dummy switch $n + 1 + t$ for $t = 0, ..., T$ is only active at stage $t$ is feasible with objective value zero. Moreover, any partial solution on the first $n$ switches that satisfies the variation constraints but leaves slack in the selection constraint at some stage $t$ can uniquely be extended without increasing the objective value by defining the dummy switch $n + 1 + t$ to be active at stage $t$ only. As a result, the definition of $c_t^{(i)}$, $i = 1, \ldots, n$, together with $S_i = 2$ implies that in an optimum solution of ($\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{SW}}$), each switch can only be active within at most one of these three intervals $\bar{I}_{ij}$, $j = 1, 2, 3$. Consequently, this yields an upper bound of $2n$ for the optimum value of Problem ($\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{SW}}$).

Then, the original instance of Interval Scheduling is a yes-instance if and only if the optimum value of the constructed instance of Problem ($\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{SW}}$) is exactly $2n$, i.e., it reaches this upper bound. The construction and the idea behind the definition of the objective is illustrated in Figure 4.4, where the dummy switches are omitted.

Indeed, given a yes-instance of Interval Scheduling, let $I_{ij_i}$ be the selected interval for job $i$. Then, set $x_t^{(i)} = 1$ for $i \in \{1, ..., n\}$ if and only if $t \in \bar{I}_{ij_i}$ and extend this partial solution to a solution over all switches by defining the dummy switch $n+1+t$ to only be active at stage $t$ if the selection constraint at this stage is not already satisfied by the partial solution over the first $n$ switches. Otherwise, define the dummy switch $n + 1 + t$ as inactive over all stages. By definition of the objective,




(a) An instance of Interval Selection with three jobs




(b) Illustration of the objective of the instance of ($\text{BND}^{\text{SEL}}_{\text{sw}}$)

Figure 4.4: Illustration of the reduction in the proof of Theorem 4.2.5

this extension does not change the objective value. This is a feasible solution for Problem ($\text{BND}^{\text{SEL}}_{\text{sw}}$), since each switch has a variation of at most two. Moreover, since the selected intervals are conflict-free and by definition of the states of the dummy switches, the solution satisfies the selection-constraint at each stage. The objective value is $2n$ and thus agrees with the upper bound.

Conversely, assume that this upper bound is attained. Then, as argued before, each of the first $n$ switches must contribute exactly 2 to the objective value, hence each switch $i \in \{1, ..., n\}$ is active for exactly two consecutive stages, namely those in $\bar{I}_{ij_i}$ for some $j_i \in \{1, 2, 3\}$. Then, define a feasible solution for Interval Scheduling by scheduling job $i$ on the respective machine $i_j$. Since the selection constraint is satisfied for each stage, no two intervals corresponding to jobs scheduled on the same machine can intersect. Hence, the instance of Interval Scheduling is a yes-instance. □

Now, one might hope that maybe further constraints fixing the selection of the switches for some stage $t$ to a given element in $X_t$ could simplify the resulting problem. However, observe that by deleting the dummy switch number $n+1$, the construction in the former proof can be adapted to also work for $K_0 = 0$ and $K_t = 1,\ t > 0$. In other words, the proof actually also implies the strong $\mathcal{NP}$-hardness of the special case in which all switches are fixed to zero in the first stage. In the following, let ($\text{BND}^{\text{SEL}}_{0,\text{sw}}$) denote this special case of ($\text{BND}^{\text{SEL}}_{\text{sw}}$).

**Remark 4.2.4.** In the setting of Theorem 4.2.5, the size of each feasible vertical set $X_t$ is given by $n$. So, Theorem 4.2.5 shows that even a polynomial size of $X_t$ does not necessarily lead to tractability of $(\mathrm{BND}_{\mathrm{sw}}^{\mathrm{SEL}})$.

Due to the previous results, one cannot expect polynomial-time algorithms for $(\mathrm{BND}_{\mathrm{sw}}^{\mathrm{SEL}})$ even for small values of $S$ and for polynomially large sets $X_t$. This draws attention to the case of a small number of switches $n$ or a small number of stages $T$.

**Theorem 4.2.6.** If the number of switches $n$ is constant, then $(\mathrm{BND}_{\mathrm{sw}}^{\mathrm{SEL}})$ can be solved efficiently with a dynamic programming scheme. A straightforward implementation of this dynamic programming scheme leads to a running time of $\mathcal{O}\left(\left(\max_{t=0,\dots,T} |X_t|\right)^2 \cdot (T+1)^{n+1}\right)$.

*Proof.* The dynamic programming scheme is devised in the following. First, introduce the vector $\sigma \in \times_{i=1}^{n}\{0,1,2,\dots,S_i\}$, where $S_i$ is the $i$-th component of $S \in \mathbb{N}_{\geq 0}^{n}$ giving the upper bound on the variation of switch $i$. The vector $\sigma$ will represent an adjusted upper bound on the variation for all switches. Moreover, introduce a vector $b \in \{0,1\}^n$ that represents the states of the switches at the currently considered stage $\bar{t}$. Define the shorthand notation $|x-y| := (|x_1-y_1|,\dots,|x_n-y_n|)^\top$ for two vectors $x$, $y$ of the same dimension. The dynamic programming scheme recursively computes the optimum values of the following subproblems:

$$
\begin{aligned}
c^*(\bar{t},\sigma,b) := \max \quad & \sum_{t=0}^{\bar{t}} c_t^\top x_t & \\
\text{s.t.} \quad & \sum_{t=0}^{\bar{t}-1} |x_{t+1}^{(i)} - x_t^{(i)}| \leq \sigma_i & i = 1,\dots,n \\
& x_t \in X_t & t = 0,\dots,\bar{t} \\
& x_{\bar{t}} = b &
\end{aligned}
$$

The initial values are defined by:

$$
c^*(0,\sigma,b) := \begin{cases} c_0^\top b & \text{if } \sigma \in \times_{i=1}^{n}\{0,1,\dots,S_i\},\ b \in X_1 \\ -\infty & \text{otherwise.} \end{cases}
$$

Then, for $\bar{t} = 1,\dots,T$, the scheme recursively computes

$$
c^*(\bar{t},\sigma,b) := \max_{\substack{x \in X_{\bar{t}-1}, \\ |x-b| \leq \sigma}} c^*(\bar{t}-1, \sigma - |x-b|, x) + \sum_{i=1}^{n} c_{\bar{t}}^{(i)} b^{(i)}
$$

for all $\sigma \in \times_{i=1}^{n}\{0,1,...,S_i\}$ and $b \in X_{\bar{t}}$. The optimum value of Problem ($\mathrm{BND}_{\mathrm{sw}}^{\mathrm{SEL}}$) is then given by

$$\max_{x \in X_T} c^*(T,S,x) \ .$$

Finally, the running time of the scheme is analyzed: for fixed $\bar{t}$ and $\sigma$, it is possible to compute the values $c^*(\bar{t},\sigma,b)$ for all $b \in X_{\bar{t}}$ by enumeration of all elements in $X_{\bar{t}-1}$ and $X_{\bar{t}}$. Note that the size of each set $|X_{\bar{t}}|$ is bounded by $2^n$. However, up to $|\times_{i=1}^{n}\{0,1,...,S_i\}|$ of such values need to be computed. In summary, the running time of the scheme can roughly be estimated by

$$\mathcal{O}\left(\left(\max_{t=0,...,T}|X_t|\right)^2 \cdot \left(\max_{i=1,...,n} S_i+1\right)^n \cdot (T+1)\right) \ .$$

Since $S_i \leq T$ without loss of generality, the running time can be bounded by

$$\mathcal{O}\left(\left(\max_{t=0,...,T}|X_t|\right)^2 \cdot (T+1)^{n+1}\right) \ .$$

In particular, this yields a polynomial running time if the number of switches $n$ is constant. □

**Remark 4.2.5.** It is obvious that the previous dynamic programming scheme can be adapted to work for the special case ($\mathrm{BND}_{0,\mathrm{sw}}^{\mathrm{SEL}}$) by simply defining $K_0 := 0$. Actually, this dynamic programming scheme is even more general: It can be adapted to work for any other combinatorial structure, as long as the membership in a set $X_t$ for any given element can be verified efficiently. This will be studied further in the following chapter.

Similarly, one can devise a dynamic programming scheme that leads to a polynomial running time, if instead of $n$, the number of stages $T$ is fixed.

**Theorem 4.2.7.** If the number of stages $T$ is constant, then ($\mathrm{BND}_{\mathrm{sw}}^{\mathrm{SEL}}$) can be solved efficiently with a dynamic programming scheme. A straightforward implementation of this dynamic programming scheme leads to a running time of $\mathcal{O}\left(n \cdot 2^{T+1} \cdot (n+1)^{T+1}\right)$.

*Proof.* This second dynamic programming scheme is based on the approach used in [BET22] for the multistage knapsack problem, and transfers the author's idea to this setting:

Consider a switch $i \in \{1,...,n\}$, then let $\Gamma_i \subseteq \{0,1\}^T$ be the set that contains all of its feasible states, i.e.

$$(x_0^{(i)},...,x_T^{(i)}) \in \Gamma_i \ \Leftrightarrow \ \sum_{i=1}^{T} |x_t^{(i)} - x_{t-1}^{(i)}| \leq S_i \ .$$

Note that if $T$ is fixed, this set can be obtained in constant time by enumerating all elements in $\{0,1\}^{T+1}$. Next, define the following subproblem for $k_t \in \{0,1,...,K_t\}$ and $j \in \{1,...,n\}$

$$\begin{aligned} c^*(k_0,...,k_T,j) = \quad & \max \quad \sum_{i=1}^{j}\sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} \\ & \text{s.t.} \quad \sum_{t=1}^{T} |x_t^{(i)} - x_{t-1}^{(i)}| \leq S_i \qquad i = 1,...,j \\ & \qquad \sum_{i=1}^{j} x_t^{(i)} = k_t \qquad\qquad t = 0,....,T \\ & \qquad x_t^{(i)} \in \{0,1\} \qquad\qquad \forall i = 1,...,j,\ t = 0,...,T \end{aligned}$$

The above problem only considers the first $j$ of the $n$ switches and for each stage $t$, there is a selection constraint with selection bound $k_t$. The initial values for $j = 1$ and for all $(k_0,...,k_T) \in \times_{t=1}^{T}\{0,...,K_t\}$ are defined by:

$$c^*(k_0,...,k_T,1) := \max\left\{\sum_{t=0}^{T} c_t^{(1)} x_t^{(1)} \colon (x_0^{(1)},...,x_T^{(1)}) \in \Gamma_1, x_t^{(1)} = k_t\ \forall t = 0,...,T\right\}.$$

Now, for $j = 2,...,n$ the respective optimal values are computed with the following recursion formula for all $(k_0,...,k_T) \in \times_{t=0}^{T}\{0,...K_t\}$:

$$c^*(k_0,...,k_T,j) = \max_{(x_0^{(j)},...,x_T^{(j)})\in\Gamma_j} c^*(k_0 - x_0^{(j)},...,k_T - x_T^{(j)}, j-1) + \sum_{t=0}^{T} c_t^{(j)} x_t^{(j)}$$

The optimal value of Problem ($\text{BND}_{\text{sw}}^{\text{SEL}}$) is then given by $c^*(K_0,...,K_T,n)$.

Now, analyze the running time of the previously described dynamic programming scheme. The maximum in the recursion formula can be computed by enumeration of all feasible states, whose number is at most $\max_{i=1,...,n}|\Gamma_i| \leq 2^{T+1}$. This maximum needs to be computed for all $(k_0,...,k_T) \in \times_{t=0}^{T}\{0,...K_t\}$, the number of which can be bounded by $\left(\max_{t=0}^{T} K_t + 1\right)^{T+1}$. Note that $K_t \leq n$ for all $t$. In total, the running time can be estimated by $\mathcal{O}\left(n \cdot 2^{T+1} \cdot (n+1)^{T+1}\right)$. □

**Remark 4.2.6.** Unlike the first dynamic programming scheme, this second scheme is customized for a combinatorial structure $X_t$ as in (SEL). Therefore, it will generally not work for other feasible sets $X_t$. However, it can be adapted to work for the special case ($\text{BND}_{0,\text{sw}}^{\text{SEL}}$) . To this end, it suffices to replace the definition of the feasible switching patterns so that $\Gamma_i$ denotes the set of feasible states for switch $i$ with the

fixation, i.e.,

$$(x_0^{(i)}, ..., x_T^{(i)}) \in \Gamma_i \;\Leftrightarrow\; \sum_{t=0}^{T-1} |x_{t+1}^{(i)} - x_t^{(i)}| \leq S_i, x_0^{(i)} = 0 \;,$$

while also including the constraints $x_0^{(i)} = 0$, $i = 1, ..., n$ in all subproblems. Alternatively, one may define $K_0 := 0$.

Summing up, both the general problem ($\text{BND}_{\text{sw}}^{\text{SEL}}$) as well as its special case ($\text{BND}_{0,\text{sw}}^{\text{SEL}}$) are tractable if either the number of switches, or the number of stages is not considered part of the input. In general, however, already the special case ($\text{BND}_{0,\text{sw}}^{\text{SEL}}$) is $\mathcal{NP}$-hard in the strong sense – even when $S_i = 2$ for all $i = 1, ..., n$ and $K_t = 1$ for all $t = 1, ..., T$. This raises the question whether the complexity changes if the bound on the variation is even lower. The remainder of this chapter deals with such tractable special cases, starting with the simplest one and then slowly generalizing the setting.

**Corollary 4.2.1.** The Problem ($\text{BND}_{0,\text{sw}}^{\text{SEL}}$) with $S_i = 1$ for $i = 1, ..., n$ and $K := K_t$ for $t = 1, ..., T$ polynomially reduces to a classical selection problem. Hence, it is tractable and can be solved in time $\mathcal{O}(n \log n)$.

Indeed, the combination of uniform selection bounds together with a switch-wise variation of one for each switch immediately let the problem ($\text{BND}_{0,\text{sw}}^{\text{SEL}}$) collapse to a classical selection problem over $\{1, ..., n\}$ with $C_i := \sum_{t=1}^{T} c_t^{(i)},\ i = 1, ..., n$.

Now, consider ($\text{BND}_{0,\text{sw}}^{\text{SEL}}$) with $S_i = 1$ for all $i = 1, ..., n$ with non-uniform selection bounds. Since each switch is required to be inactive in the beginning, i.e., $x_0^{(i)} = 0$ for all $i = 1, ..., n$, a bound of $S_i = 1$ implies that each switch either has no variation at all, or that it is activated exactly once and then remains active until the last stage. The best possible value that any switch $i$ can achieve is thus given by:

$$\text{val}(i) := \max_{\bar{t}=0,...,T} \left\{ \sum_{t=\bar{t}}^{T} c_t^{(i)},\ 0 \right\} \;.$$

However, this observation does not help, since a switch $i$ that is selected in an optimal solution may not achieve its individual optimum value $\text{val}(i)$, as the example below shows.

**Example 4.2.1.** Consider the following instance of ($\text{BND}_{0,\text{sw}}^{\text{SEL}}$) with $n = 2$, $T = 2$ and $S_1 = S_2 = 1$, $K_1 = 1, K_2 = 2$. The objective $c$ is given by

$$c = \begin{pmatrix} 0 & 3 & 0 \\ 0 & 1 & 1 \end{pmatrix} \;,$$

where the row $i = 1, 2$ contains the vector $(c_0^{(i)}, c_1^{(i)}, c_2^{(i)})$. Using the same notation, the (unique) optimal solution $x^*$ is given by

$$x^* = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} .$$

But, for switch 2 on its own, it is $\mathrm{val}(2) = 2$ for $(x_0^{(2)}, x_1^{(2)}, x_2^{(2)}) = (0, 1, 1)$, which does not agree with the optimal solution.

Nonetheless, the problem $(\mathrm{BND}^{\mathrm{SEL}}_{0,\mathrm{sw}})$ can be polynomially reduced to a different tractable problem:

**Theorem 4.2.8.** The Problem $(\mathrm{BND}^{\mathrm{SEL}}_{0,\mathrm{sw}})$ with $S_i = 1$ for all $i = 1, ..., n$ polynomially reduces to a min cost flow problem and hence, can be solved efficiently.

*Proof.* Let an instance of $(\mathrm{BND}^{\mathrm{SEL}}_{0,\mathrm{sw}})$ be given so that $S_i = 1$ for all $i = 1, ..., n$. In the following, the construction of the min cost flow network is described:
Introduce a source $\alpha$ and a sink $\omega$, as well as nodes $\bar{S}_i$ for all $i = 1, ..., n$ and nodes $s_t^{(i)}$ for all $i = 1, ..., n,\ t = 0, ..., T$. Each node $\bar{S}^{(i)}$ is connected with both the source $\alpha$ by the arc $(\alpha, \bar{S}^{(i)})$ and each of the nodes $s_t^{(i)}$ for $t = 1, ..., T$ by the arc $(\bar{S}^{(i)}, s_t^{(i)})$. A positive flow on arc $(\alpha, \bar{S}^{(i)})$ will correspond to the switch $i$ being activated at some stage. Further define nodes $v_t$ for $t = 0, ..., T$ and connect the nodes $s_t^{(i)}$ for all $i = 1, ..., n$ with those through the arcs $(s_t^{(i)}, v_t)$. Then, a positive flow on arc $(\bar{S}^{(i)}, s_t^{(i)})$ will correspond to switch $i$ being activated at stage $t$ (and remaining on). Connect the nodes $v_t$ to each other via the edges $(v_t, v_{t+1})$ for all $t = 0, ..., T-1$. At last, the vertex $v_T$ is connected with the sink $\omega$ via the arc $(v_T, \omega)$. This yields the following set of vertices

$$V := \{\cup\{\bar{S}^{(i)} \colon i = 1, ..., n\}\ \cup \{s_t^{(i)} \colon i = 1, ..., n,\ t = 0, ..., T\}\ \cup \{v_t \colon t = 1, ..., T\}$$

and the following set of arcs

$$\begin{aligned} A := \{(\alpha, \bar{S}_i) \colon i = 1, ..., n\} \cup \{(\bar{S}^{(i)}, s_t^{(i)}), (s_t^{(i)}, v_t) \colon i = 1, ..., n,\ t = 0, ..., T\} \\ \cup \{(v_t, v_{t+1}) \colon t = 0, ..., T-1\} \\ \cup \{(v_T, \omega)\} . \end{aligned}$$

The costs $c$ of the arcs are defined as zero for all arcs, except for

$$c((\bar{S}^{(i)}, s_t^{(i)})) = C_t^{(i)} := \sum_{j=t}^{T} -c_j^{(i)} \qquad \forall i = 1, ..., n,\ t = 0, ..., T .$$

The upper capacity bound $u$ of all arcs is defined to be one, expect for

$$u((\alpha,\omega)) := K_T, \quad u((v_t,v_{t+1})) := K_t \quad \forall t = 0,...,T-1,\ u((v_T,\omega)) := K_T$$

while the lower capacity bound is defined as $K_t$ for the arcs $(v_t, v_{t+1})$ and zero otherwise. At last, define the supply $b_\alpha := K_T$ and the demand $b_\omega := -K_T$, while all other nodes in the network have demand/supply zero. The resulting network is sketched in Figure 4.5 for $n = 3$ and $T = 3$.

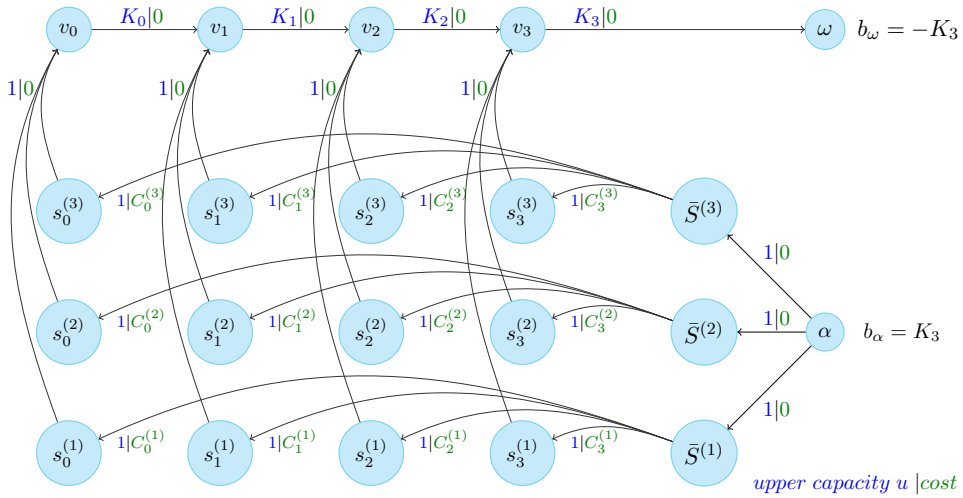



Figure 4.5: Sketch of the constructed min cost flow network for $n = 3$ and $T = 3$

Due to the definition of the capacities in the min cost flow network, any feasible integer flow $f$ must satisfy the following constraints:

$$f_{(\alpha,S^{(i)})} = f_{(S^{(i)},s_t^{(i)})} = f_{(s_t^{(i)},v_t)} \in \{0,1\} \qquad i = 1,...,n,\ t = 0,...,T$$

$$f_{(v_t,v_{t+1})} = \sum_{j=1}^{t}\sum_{i=1}^{n} f_{(s_j^{(i)},v_j)} = K_t \qquad t = 0,...,T-1$$

$$f_{(v_T,\omega)} = \sum_{j=1}^{T}\sum_{i=1}^{n} f_{(s_j^{(i)},v_j)} = \sum_{i=1}^{n} f_{(a,\bar{S}^{(i)})} = K_T$$

Therefore, a feasible solution $x$ of $(\mathrm{BND}^{\mathrm{SEL}}_{0,\mathrm{sw}})$ can be transformed into a feasible integer flow $f$, and vice versa, as follows:

$$f_{(\alpha,\bar{S}^{(i)})} = f_{(\bar{S}^{(i)},s_t^{(i)})} = f_{(s_t^{(i)},v_t)} = 1 \quad \Leftrightarrow \quad x_{t-1}^{(i)} = 0,\ x_t^{(i)} = 1\ .$$

By definition of the costs in the network, both solutions yield the same objective value (disregarding the sign). From the literature, it follows that there always exists

an optimal integer flow $f^*$, as all capacity bounds in the network are integer. Thus, $(\mathrm{BND}^{\mathrm{SEL}}_{0,\mathrm{sw}})$ polynomially reduces to the described min cost flow problem. □

**Remark 4.2.7.** Just as in the previous section (see Remark 4.2.2), the polynomial reduction of $(\mathrm{BND}^{\mathrm{SEL}}_{0,\mathrm{sw}})$ to a min cost flow problem also yields an extended LP formulation for $(\mathrm{BND}^{\mathrm{SEL}}_{0,\mathrm{sw}})$ as a by-product. Then, a description of the convex hull of feasible solutions for $(\mathrm{BND}^{\mathrm{SEL}}_{0,\mathrm{sw}})$ in the original space of only the $x$-variables can be obtained, for example, by projection.

In the more general problem $(\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{sw}})$ with $S_i = 1$, $i = 1, ..., n$, a switch can also be active in the beginning and may be deactivated (and stay deactivated). Since it is not obvious how, or whether at all, it is possible to adapt the previous min cost flow construction to capture this new setting, the tractability of $(\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{sw}})$ will be proven with the help of the following variant of the matching problem:

**Definition 4.2.1.** Let $G = (V, E)$ be a graph with edge weights $w : E \to \mathbb{R}$. The following problem is called CARDINALITY-CONSTRAINED MAXIMUM WEIGHT MATCHING PROBLEM:

$$\text{(CMWM)} \qquad \begin{cases} \max & \displaystyle\sum_{e \in M} w(e) \\ \text{s.t.} & M \subseteq E \text{ is a matching in } G \\ & |M| = k \ . \end{cases}$$

**Lemma 4.2.2.** The Problem (CMWM) can be solved in polynomial time.

*Proof.* The tractability of (CMWM) can be proven by a reduction to the MINIMUM WEIGHT PERFECT MATCHING PROBLEM (MWPM).

Let $(G = (V, E), w, k)$ be an instance of (CMWM), where $V := \{v_1, ..., v_n\}$. Then, define an instance $(\bar{G} = (\bar{V}, \bar{E}), \bar{w})$ of (MWPM) by

$$\begin{aligned} \bar{V} &:= V \cup \{u_1, ..., u_{n-2k}\} \\ \bar{E} &:= E \cup \{(v_i, u_j) \mid i = 1, ..., n;\ j = 1, ..., n - 2k\} \\ \bar{w}(e) &:= -w(e) \quad \forall e \in E \\ \bar{w}(e) &:= 0 \quad \forall e \in \bar{E} \setminus E \end{aligned}$$

Thus, any perfect matching $\bar{M} \subseteq \bar{E}$ in $\bar{G}$ must contain exactly $n-2k$ edges $(v_i, u_j)$ in order to match the nodes $u_j$. The remaining $|V| - (n-2k) = 2k$ nodes in $V$ that are not matched to the nodes $u_j$ must therefore be matched using $k$ edges from $E$. Since

all of the edges $(v_i, u_j)$ have weight zero, a minimum weight perfect matching $\bar{M}$ in $\bar{G}$ thus induces a maximum-weight perfect matching $M := \bar{M} \cap E$ in $G$.

Conversely, any maximum weight matching $M$ in $G$ with $|M| = k$ can be transformed into a minimum weight perfect matching $\bar{M}$ in $\bar{G}$: W.l.o.g. let $v_1, ..., v_{n-2k}$ be the nodes in $V$ that are not incident to an edge in $M$. Additional inclusion of the edges $(v_i, u_i)$ for $i = 1, ..., n - 2k$ yields a perfect matching $\bar{M}$ in $\bar{G}$ while achieving the negated objective value.
Since (MWPM) can be solved in polynomial time, e.g., by Edmond's blossom algorithm [Edm65a], the claim follows. □

As will be shown soon, ($\text{BND}_{\text{sw}}^{\text{SEL}}$) with $S_i = 1$ for all $i = 1, ..., n$ polynomially reduces to (CMWM), which implies the tractability of ($\text{BND}_{\text{sw}}^{\text{SEL}}$). Since this reduction gets less intricate for instances of ($\text{BND}_{\text{sw}}^{\text{SEL}}$) with a uniform selection bound, the reduction is delayed until after the next lemma.

**Lemma 4.2.3.** The problem ($\text{BND}_{\text{sw}}^{\text{SEL}}$) polynomially reduces to an instance of ($\text{BND}_{\text{sw}}^{\text{SEL}}$) with uniform selection bounds. As a result, the selection bounds in ($\text{BND}_{\text{sw}}^{\text{SEL}}$) may be assumed to be uniform without loss of generality.

*Proof.* The problem ($\text{BND}_{\text{sw}}^{\text{SEL}}$) with $S_i = 1$ for all $i = 1, ..., n$ and $K_t \in \{0, ..., n\}$ for all $t = 0, ..., T$ polynomially reduces to ($\text{BND}_{\text{sw}}^{\text{SEL}}$) with $S_i = 1$ for all $i = 1, ..., n$ and uniform selection bounds $K := K_t$ for all $t = 0, ..., T$ as follows:
Let an instance $I = (c, (K_0, ..., K_T), S = 1_n)$ of ($\text{BND}_{\text{sw}}^{\text{SEL}}$) be given, where $n$ denotes the number of switches. As long as the selection bound is not uniform, identify an index $\ell \in \{0, ...., T-1\}$ with $K_\ell \neq K_{\ell+1}$ and define a new instance of ($\text{BND}_{\text{sw}}^{\text{SEL}}$) by adding $m := |K_\ell - K_{\ell+1}|$ new switches which are indexed by $n+1, ...., n+m$. In addition, define $C := \sum_{i=1}^{n} \sum_{t=0}^{T} |c_t^{(i)}| + 1$. The construction of this new instance depends on the sign of $K_\ell - K_{\ell+1}$:

- If $K_{\ell+1} - K_\ell > 0$, then for all new switches $i = n+1, ..., n+m$ define $c_t^{(i)} = C$ for $t \leq \ell$ and $c_t^{(i)} = -C$ for $t > \ell$. Furthermore, replace the former selection bound $K_t$ for all $t \leq \ell$ by $K_t + m$. All other parameters remain unchanged. This way, any optimal solution of the new instance satisfies $x_t^{(i)} = 1$ for all new switches $i$ if and only if $t \leq \ell$. Consequently, the former switches satisfy the original selection bounds.

- If $K_{\ell+1} - K_\ell < 0$, then for the new switches $i = n+1, ..., n+m$ define $c_t^{(i)} = -C$ for $t \leq \ell$ and $c_t^{(i)} = C$ for $t > \ell$. Replace the former selection bound $K_t$ for $t > \ell$ by $K_t + m$. All other parameters remain the same. Then any optimal solution of the new instance satisfies $x_i^{(t)} = 1$ for the new switches if and only if $t > \ell$.

As a result of the construction, the selection bound at stages $\ell$ and $\ell + 1$ is equal. At most $\mathcal{O}(T)$ recursive applications are necessary to yield an instance of ($\text{BND}_{\text{sw}}^{\text{SEL}}$)

in which the upper bound on the selection constraint is uniform. Furthermore, said final value $K$ is bounded from above by $\sum_{t=0}^{T-1} |K_{t+1} - K_t|$, so that the presented construction is of polynomial size. □

**Theorem 4.2.9.** The Problem ($\text{BND}_{\text{sw}}^{\text{SEL}}$) with $S_i = 1$ for all $i = 1, ..., n$ can be solved in polynomial time.

*Proof.* Due to Lemma 4.2.3, it suffices to prove that an instance of ($\text{BND}_{\text{sw}}^{\text{SEL}}$) with $S_i = 1$ for all $i = 1, ..., n$ and $K := K^{(t)}$ for all $t = 0, ..., T$ can be polynomially reduced to an instance of (CMWM). Thus, given such an instance of ($\text{BND}_{\text{sw}}^{\text{SEL}}$), define a graph $G = (V, E)$ by

$$V := \{v_1, ..., v_n\} \cup \{u_1, ..., u_n\} \text{ and}$$
$$E := \{(v_i, v_j) \mid i, j \in \{1, ..., n\}, i \neq j\} \cup \{(v_i, u_i) \mid i = 1, ..., n\} .$$

Furthermore, define weights $w : E \to \mathbb{R}$ for the edges as follows:

$$c((v_i, v_j)) := \max_{p<q} \left\{ \sum_{t=0}^{p} c_t^{(i)} x_t^{(i)} + \sum_{t=q}^{T} c_t^{(j)} x_t^{(j)}, \sum_{t=1}^{p} c_t^{(j)} x_t^{(j)} + \sum_{t=q}^{T} c_t^{(i)} x_t^{(i)} \right\}$$
$$c((v_i, u_i)) := \max_{p=0,...,T} \left\{ \sum_{t=0}^{p} c_t^{(i)} x_t^{(i)}, \sum_{t=p}^{T} c_t^{(i)} x_t^{(i)} \right\} .$$

The size $k'$ that a matching in the instance of (CMWM) has to meet is defined by $k' := K$.

The idea behind this definition can be described as follows: Consider two states of switches $i, j$, $i \neq j$, which are *overlap-free*, i.e., where there is no stage $t \in \{0, ..., T\}$ at which both switches are active simultaneously. Then, the weight $w((v_i, v_j))$ of an edge $(v_i, v_j)$ can be interpreted as the optimal value that the pair $i, j$, $i \neq j$ can achieve under the constraint that it is overlap-free. The weight $w((v_i, u_i))$ of an edge $(v_i, u_i)$ represents the optimal value that the switch $i$ can achieve on its own.

The following property will enable a precise description of the reduction:

Claim: Let $x$ be feasible for the instance of ($\text{BND}_{\text{sw}}^{\text{SEL}}$) and call a switch $i$ *active* with respect to $x$ if the corresponding vector $x^{(i)}$ is not entirely zero. Then, the set of active switches in $\{1, ..., n\}$ with respect to $x$ can be partitioned into a set $A \subseteq \{1, ..., n\}^2$ and a set $B \subseteq \{1, ..., n\}$ so that $|A| + |B| = K$. Here, the set $A$ consists of pairs $(i, j)$, $i \neq j$, of active switches with respect to $x$, where each pair $(i, j)$ is non-overlapping. The set $B$ contains all remaining switches that are active with respect to $x$.

Indeed, this follows from the fact that exactly $K$ switches must be active at any stage. Therefore, each deactivation at some stage $t$ implies the existence of an

activation at the same stage $t$ that balances out the cardinality of the set of active switches, and vice versa.

Figure 4.6a gives an example for a solution $x$ of ($\text{BND}_{\text{sw}}^{\text{SEL}}$) with $n = 7$ and $T = 7$ and a possible partition. The pairs of switches in $A$ are illustrated by a common color, while the switches in $B$ receive an individual color. Now, the proof of the theorem continues.

Given a feasible solution $x$ of ($\text{BND}_{\text{sw}}^{\text{SEL}}$), a feasible solution $M$ of (CMWM) can be constructed as follows: For each pair $(i, j) \in A$, add the edge $(v_i, v_j)$ to $M$ and for each element $i$ in $B$, add the edge $(v_i, u_i)$. By construction, $M$ is a matching with cardinality $K$ and the objective value is at least as good as the value achieved by $x$ in ($\text{BND}_{\text{sw}}^{\text{SEL}}$).

Conversely, let $M \subseteq E$ be a matching in $G$ with $|M| = K$. For an edge $(v_i, v_j) \in M$, let $p^*, q^*$ denote the maximizing arguments of its respective weight $w((v_i, v_j))$ and then, either define

$$x_t^{(i)} := 1 \text{ for } t = 0, ..., p^* \, , x_t^{(j)} := 1 \text{ for } t = q^*, ..., T \quad \text{(and zero otherwise), or}$$
$$x_t^{(i)} := 1 \text{ for } t = p^*, ..., T \, , x_t^{(j)} := 1 \text{ for } t = 0, ..., q^* \quad \text{(and zero otherwise),}$$

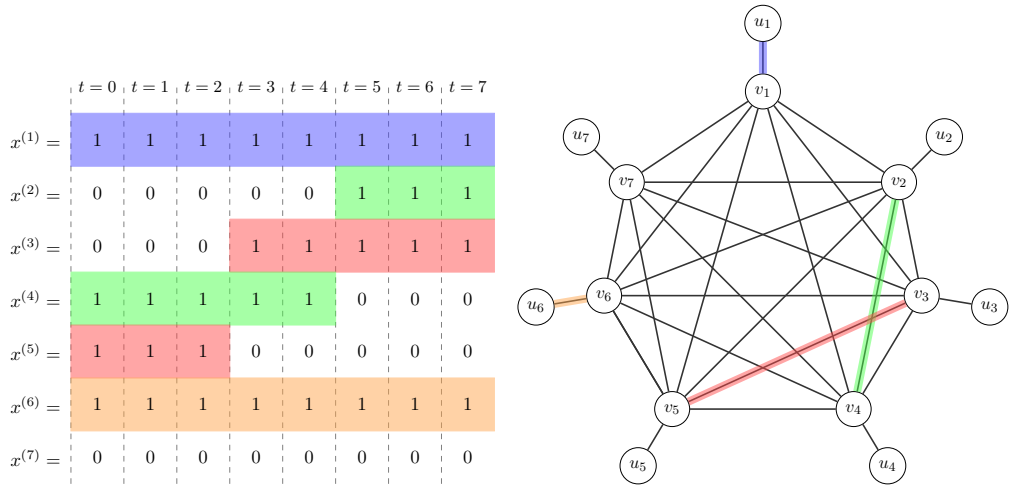
depending on which solution yields the weight of the edge. For any edge $(v_i, u_i) \in M$, let $p^*$ denote the maximizing argument of its weight $w((v_i, u_i))$ and, analogously, define

$$x_t^{(i)} := 1 \text{ for } t = 0, ..., p^* \quad \text{(and zero otherwise), or}$$
$$x_t^{(i)} := 1 \text{ for } t = p^*, ..., T \quad \text{(and zero otherwise).}$$

By definition, $x$ does not exceed the bound of one on the variation for any switch. Since the cardinality of $M$ is $K$, $x$ also satisfies the selection constraint in ($\text{BND}_{\text{sw}}^{\text{SEL}}$) and is therefore feasible. Moreover, $x$ achieves an objective value of $w(M)$ in ($\text{BND}_{\text{sw}}^{\text{SEL}}$). The relation between a feasible solution $x$ and a corresponding matching in $G$ is sketched in Figure 4.6.

□

This concludes the study of ($\text{BND}_{\text{sw}}^{\text{SEL}}$) for the case $S = \mathbb{1}_n$ and also this section about the problem ($\text{BND}_{\text{sw}}^{\text{SEL}}$). Summing up, the type of variation that is considered has a strong impact on the complexity of the problem (BND) if additional selection constraints as in (SEL) are present. For time-wise variation, additional selection constraints do not change the problems complexity compared to the unbounded case in which $X_t = \{0, 1\}^n$ for all $t = 0, ..., T$. For switch-wise variation, however, the unbounded problem is tractable, while the problem ($\text{BND}_{\text{sw}}^{\text{SEL}}$) with additional selection constraints is strongly $\mathcal{NP}$-hard even for rather small values $S$ and $K$.




(a) solution $x$, pairs of switches in $A$ have a common color, switches in $B$ have an individual color

(b) corresponding matching in $G$

Figure 4.6: Sketch of the reduction in the proof of Theorem 4.2.9 with $n = 7$, $T = 7$ and $K = 4$

Table 4.1 below provides an overview of the complexity results obtained in this section.

| | $S = \mathbb{1}_n$ | $S = 2 \cdot \mathbb{1}_n,\ K = \mathbb{1}_{T+1}$ | $n$ constant | $T$ constant |
|---|---|---|---|---|
| $(\text{BND}^{\text{SEL}}_{\text{sw}})$ | $\mathcal{P}$ (Theorem 4.2.8) | strongly $\mathcal{NP}$-hard (Theorem 4.2.5) | $\mathcal{P}$ (Theorem 4.2.6) | $\mathcal{P}$ (Theorem 4.2.7) |

Table 4.1: Summary of complexity results obtained in Section 4.2.2

This leaves only one type of variation left to study, namely total variation.

### 4.2.3 Total Variation

This section is dedicated to the problem (BND) with total variation. In the following, the unbounded problem will be discussed, while the problem with additional selection constraints will be considered thereafter.

Just as the unbounded problem (BND) with switch-wise variation, also the unbounded problem (BND) with total variation can be considered a generalization of the single switch problem studied in Section 3.3. Now, however, this problem does not simply decompose into $n$ single switch problems. Of course, one could enumerate all possibilities to transform the bound $S$ on the total variation into a switch-wise variation bound $(S_1, ..., S_n)$, but this reduction is not polynomial unless $S$ is a fixed

constant. Broadly speaking, a total variation bound is somewhat less restrictive than a switch-wise variation bound.

Approaching the problem from a polyhedral perspective does not help much either, as there is no obvious way in which LP descriptions for the single switch problem can be extended to the general unbounded case with multiple switches. Recall that in an extended LP formulation for the single switch problem, the key element is that the activations and deactivations of the switch are bounded separately by either $\lfloor \frac{S}{2} \rfloor$ or $\lfloor \frac{S-1}{2} \rfloor + 1$, instead of their overall sum. This implies that the same is necessary for the generalization here. However, if there are multiple switches, then it is no longer possible to define a similar non-trivial bound on the number of total activations and deactivations. For example, if $n \geq S$ then even a solution with $S$ total activations is feasible.

Nonetheless, if there is no combinatorial structure, the problem (BND) with total variation is tractable. Indeed, by concatenating the switches $i = 1, ..., n$ in no particular order, one obtains a single switch with $n \cdot (T+1)$ stages. However, different consecutive entries at the stages where one switch transitions into the next must not be included when computing the variation. Adjusting the dynamic programming scheme for the single switch problem from Section 3.3 accordingly yields

**Lemma 4.2.4.** The unbounded problem (BND) with total variation can be solved in polynomial time with a dynamic programming scheme. A straightforward implementation of this scheme leads to a running time of $\mathcal{O}(n \cdot S \cdot T)$.

*Proof.* The dynamic programming scheme devised in the following is a generalization of the one that was devised for the single switch problem. In fact, the set of $n$ switches can be interpreted as a single switch of length $n \cdot (T+1)$ and the only necessary adjustment is a redefinition of the recursion formula for the stages that indicate the beginning of a new switch. Now, for $i = 1, ..., n$, let $\gamma^{(i)}(\bar{t}, s, b)$ denote the optimal value considering only the stages $\{0, 1, ..., i \cdot \bar{t}\}$ of this large single switch, where $s \in \{0, 1, ..., S\}$ is a bound on the term $\sum_{j=1}^{i} \sum_{t=0}^{\bar{t}-1} |x_t^{(i)} - x_{t+1}^{(i)}|$, and it is required that $x_{\bar{t}}^{(i)} = b \in \{0, 1\}$. Now, define the initial values by

$$\gamma^{(1)}(0, s, b) := c_0^{(1)} \cdot b \qquad \forall s \in \{0, 1, ..., S\},\ b \in \{0, 1\}\ .$$

Then, for $i = 1, ..., n$, compute the remaining optimal values with the help of the subsequent recursion formula: For $\bar{t} = 0$ (and $i > 1$), compute

$$\gamma^{(i)}(0, s, b) = b \cdot c^{(i)} + \max\left\{\gamma^{(i-1)}(T, s, b),\ \gamma^{(i-1)}(T, s, 1-b)\right\}$$

for all $s \in \{0, ..., S\}$ and $b \in \{0, 1\}$. For $\bar{t} = 1, ..., T$, compute:

$$\gamma^{(i)}(\bar{t}, s, b) = b \cdot c_{\bar{t}}^{(i)} + \max\begin{cases}\gamma^{(i)}(\bar{t}-1, s, b) \\ \gamma^{(i)}(\bar{t}-1, s-1, 1-b) & \text{if } s \geq 1\end{cases}$$

for all $s \in \{0, ..., S\}$, $b \in \{0, 1\}$.
The optimal value is then given by $\max_{b\in\{0,1\}} \gamma^{(n)}(T, S, b)$ and it can be computed in time $\mathcal{O}(n \cdot S \cdot T)$. An optimal solution can be constructed from the recursion. Note that the recursion formula for $\bar{t} > 0$ is exactly the recursion from the single switch case, hence it suffices to argue the correctness of the other recursive term. The correctness of this formula, however, is obvious: If the switch $i - 1$ transitions into switch $i$, then different consecutive entries do not contribute to the total variation. Consequently, the best solution so that switch $i$ begins with the state $b$ and has a total variation of at most $s$ is obtained by simply extending the best solution that ends at the last stage of switch $i - 1$ with the same bound on the variation. □

As soon as additional selection constraints come into play, however, the previous dynamic programming scheme clearly ceases to work. The remainder of this section thus deals with the following problem

$$(\mathrm{BND}_{\mathrm{tv}}^{\mathrm{SEL}}) \qquad \begin{cases} \max & \displaystyle\sum_{i=1}^{n}\sum_{t=0}^{T} c_t^{(i)} \cdot x_t^{(i)} & \\ \text{s.t.} & \displaystyle\sum_{t=0}^{T-1}\sum_{i=1}^{n} |x_{t+1}^{(i)} - x_t^{(i)}| \leq S & \\ & \displaystyle\sum_{i=1}^{n} x_t^{(i)} = K_t & t = 0, ..., T \\ & x_t \in \{0, 1\}^n & t = 0, ..., T\ . \end{cases}$$

The general complexity of $(\mathrm{BND}_{\mathrm{tv}}^{\mathrm{SEL}})$ will not be settled in this thesis. However, if some of the parameters are not considered part of the input, the problem is tractable.

**Lemma 4.2.5.** The problem $(\mathrm{BND}_{\mathrm{tv}}^{\mathrm{SEL}})$ can be solved with a dynamic programming scheme. A straightforward implementation of this dynamic programming scheme leads to a running time of $\mathcal{O}\left(T \cdot S \cdot (\max_{t=0,...,T} |X_t|)^2\right)$. For fixed selection bounds $K_t$, $t = 0, ..., T$ or a fixed number of switches $n$, the running time is polynomial.

*Proof.* The dynamic programming scheme is devised in the following. Recall the shorthand notation $|x - y| := (|x_1 - y_1|, ..., |x_n - y_n|)^\top$ for two vectors $x$, $y$ of the same dimension. Let the vector $b \in X_{\bar{t}}$ represent the states of the switches at the currently considered stage $\bar{t}$ and introduce an integer $\sigma \in \{0, 1, ..., S\}$ which describes an adjusted upper bound on the total variation. Within the scheme, the optimal

values of the following subproblems for $\bar{t} \in \{0, ..., T\}$ are computed recursively:

$$\begin{aligned} c^*(\bar{t}, \sigma, b) := \max \quad & \sum_{i=1}^{n} \sum_{t=0}^{\bar{t}} c_t^{(i)} \cdot x_t^{(i)} \\ \text{s.t.} \quad & \sum_{t=0}^{\bar{t}-1} \sum_{i=1}^{n} |x_{t+1}^{(i)} - x_t^{(i)}| \leq \sigma \\ & x_t \in X_t \qquad\qquad t = 0, ..., \bar{t} \\ & x_{\bar{t}}^{(i)} = b_i \qquad\qquad i = 1, ..., n \end{aligned}$$

The initial values are defined by

$$c^*(0, \sigma, b) := \begin{cases} \sum_{i=1}^{n} c_0^{(i)} \cdot b_i & \forall b \in X_0,\ \forall \sigma \in \{0, 1, ...., S\} \\ -\infty & \text{otherwise .} \end{cases}$$

Then, for $\bar{t} = 1, ..., T$, the scheme recursively computes

$$c^*(\bar{t}, \sigma, b) := \max_{x \in X_{\bar{t}-1}} c^*(\bar{t} - 1, \sigma - |b - x|, x) + \sum_{i=1}^{n} c_{\bar{t}}^{(i)} \cdot b_i$$

for all $b \in X_{\bar{t}}$ and all $\sigma \in \{0, ..., S\}$. The optimal value of $(\text{BND}_{\text{tv}}^{\text{SEL}})$ is thus given by

$$\max_{b \in X_T} c^*(T, S, b) \ .$$

The correctness of the recursion formula is obvious, since the best solution $x_0, ..., x_{\bar{t}-1}$ that extends the solution $b$ at stage $\bar{t}$ can clearly be determined by enumeration of all solutions for the subproblem up until the stage $\bar{t} - 1$ so that the bound on the variation is not exceeded. It remains to analyze the running time. In each iteration $\bar{t} = 1, ..., T$, the maximum in the recursion formula can be determined by enumeration of the elements in $X_{\bar{t}-1}$ and the scheme computes a total number of $\mathcal{O}(S \cdot |X_{\bar{t}}|)$ such maxima. This yields an overall estimation of the running time by $\mathcal{O}\left(T \cdot S \cdot (\max_{t=0,...,T} |X_t|)^2\right)$. If the selection bounds $K_t$, $t = 0, ..., T$, are not considered part of the input, then the feasible elements in any set $X_t$ can be enumerated in polynomial time, since their number is bounded by $\mathcal{O}(n^{K_{max}})$, where $K_{\max}$ denotes the largest selection bound. This implies the polynomial running time for fixed selection bounds. Since the maximum cardinality $\max_{t=0,...,T} |X_t|$ can trivially be bounded by $2^n$, polynomial running time also follows for a fixed number of switches $n$. □

**Remark 4.2.8.** It is obvious that the previous dynamic programming scheme can again be adapted to work for the special case $(\text{BND}_{0,\text{tv}}^{\text{SEL}})$ by simply defining $K_0 := 0$. Actually, also this dynamic programming scheme can be adapted to work for any other combinatorial structure, as long as the membership in a set $X_t$ for any given element can be verified efficiently. This will resurface in the following chapter.

Analogously, one might consider the case in which the bound $S$ on the total variation is a fixed constant. As already mentioned in the beginning of this section, when speaking of the unbounded problem with total variation, it is possible to enumerate all $\mathcal{O}(n^S)$ possibilities to transform the total bound $S$ into a switch-wise variation bound $(S_1, ..., S_n)$. Although this reduction is polynomial if $S$ is fixed, it is not very helpful, as the problem ($\text{BND}_{\text{sw}}^{\text{SEL}}$) with a selection constraint is $\mathcal{NP}$-hard in general, even for small values of $S$. Analogously, enumerating all possibilities to transform the total bound $S$ into a time-wise variation bound $(S_1, ..., S_T)$ with $\sum_{t=1}^{T} S_i = S$ needs time $\mathcal{O}(T^S)$. Here, ($\text{BND}_{\text{tv}}^{\text{SEL}}$) reduces to solving a polynomial number of instances ($\text{BND}_{\text{tw}}^{\text{SEL}}$) with time-wise variation, where the latter can be solved efficiently.

**Lemma 4.2.6.** If the bound on the total variation $S$ is not part of the input, then ($\text{BND}_{\text{tv}}^{\text{SEL}}$) can be solved in polynomial time.

If the number of stages $T$ is fixed, the same reduction to ($\text{BND}_{\text{tw}}^{\text{SEL}}$) still works out, since the number of possibilities to transform the total bound $S$ into a time-wise variation bound can trivially be bounded by $\mathcal{O}(S^T)$.

**Lemma 4.2.7.** If the number of stages $T$ is not part of the input, then ($\text{BND}_{\text{tv}}^{\text{SEL}}$) can be solved in polynomial time.

To summarize, as long as either the number of switches $n$, the number of stages $T$, the bound $S$, or the selection bounds $K_t$, $t = 0, ..., T$ are fixed, ($\text{BND}_{\text{tv}}^{\text{SEL}}$) is tractable. The problem's complexity is unclear, however, if all of these are part of the input.

Although the complexity of ($\text{BND}_{\text{tv}}^{\text{SEL}}$) is unclear in general, the previous results still allow to make a few observations regarding the complexity.

**Remark 4.2.9.** As the reduction for Lemmas 4.2.6 and 4.2.7 indicates, the problem ($\text{BND}_{\text{tv}}^{\text{SEL}}$) is tractable, as soon the total bound $S$ is partitioned into a time-wise variation bound $(S_1, ..., S_T)$. Consequently, if the problem turns out to be $\mathcal{NP}$-hard in general, then its hardness should follow from the fact that such an optimal partition is hard to find.

Approaches that aim to show the $\mathcal{NP}$-hardness of ($\text{BND}_{\text{tv}}^{\text{SEL}}$) by a reduction from ($\text{BND}_{\text{sw}}^{\text{SEL}}$) usually fail due to the fact that the total variation bound leaves too much freedom compared to a switch-wise variation constraint.

This concludes the section. Its most important findings are briefly summarized in the following: While the penalty problem is tractable both for feasible sets $X_t = \{0, 1\}^n$ for all $t$ and even for feasible sets corresponding to the solutions of a SELECTION PROBLEM, the complexity of the bounded problem (BND) strongly

depends on which type of variation is considered. If there are no additional combinatorial constraints, i.e., $X_t = \{0,1\}^n$ for all $t$, then the problem can be solved efficiently for all three types of variation. For time-wise variation, the resulting problem can be solved by linear programming. For switch-wise variation, the tractability is implied by the tractability of the single switch problem from Section 3.3, as the former problem immediately decomposes into $n$ separate instances of the latter problem. For total variation, a generalization of the efficient dynamic programming scheme from the single switch problem shows the polynomial time solvability.

However, the complexity of the problem versions can change drastically if a selection constraint is imposed on each of the feasible sets $X_t$. In fact, the bounded problem with time-wise variation and a selection constraint can be reduced to linear programming as well as to a min cost flow problem, implying the problem's tractability, so that the additional selection constraints do not lead to an increase in the problem's complexity. In contrast, the problem ($\text{BND}^{\text{SEL}}_{\text{sw}}$) is strongly $\mathcal{NP}$-hard, even if the bound on the (switch-wise) variation is uniformly equal to two and all selection bounds are uniformly equal to one. In case the number of switches, or the number of stages is fixed, the problem can be solved in polynomial time by dynamic programming. In addition, the problem reduces to a maximum cardinality matching problem if the bounds are uniformly equal to one. The complexity of the problem ($\text{BND}^{\text{SEL}}_{\text{tv}}$) in general was not settled, but it can be solved in polynomial time as soon one of the parameters $n$, $S$, $T$, or $K$ is fixed. Table 4.2 below summarizes the main complexity results of this section.

| | (P) | ($\text{BND}^{\text{SEL}}_{\text{tw}}$) | ($\text{BND}^{\text{SEL}}_{\text{sw}}$) | ($\text{BND}^{\text{SEL}}_{\text{tv}}$) |
|---|---|---|---|---|
| $X_t = \{0,1\}^n$ | $\mathcal{P}$ | $\mathcal{P}$ | $\mathcal{P}$ | $\mathcal{P}$ |
| $X_t$ as in (SEL) | $\mathcal{P}$ | $\mathcal{P}$ | strongly $\mathcal{NP}$-hard for $S = 2$, $K = 1$<br>$\mathcal{P}$ for $n$ or $T$ fixed, or for $S = 1$ | ?<br>$\mathcal{P}$ for $n$,$T$,$S$ or $K$ fixed |

Table 4.2: Summary of main results of Section 4.2

The subsequent section explores some generalizations and related problems.

## 4.3 Generalizations and other Remarks

In this section, different generalizations of (BND) are studied. With similar reasoning as in the previous sections, these problems can be proven to be $\mathcal{NP}$-hard. In the first generalization, the problem input additionally includes a family $\mathcal{I}$ of subsets of the set of switches $\{1, ..., n\}$ and now, the selection constraint does not apply to the set of all switches, but instead there is a selection constraint for each of the sets in $\mathcal{I}$. Moreover, the selection constraint at each stage is relaxed to enforce only

an upper bound on the cardinality. It is not difficult to show that this problem is strongly $\mathcal{NP}$-hard in general and unlike before, this now holds for all three types of variation. In fact, this statement can be taken even further, since actually, the problem is strongly $\mathcal{NP}$-hard, even if $\mathcal{I}$ is a laminar set family and there is a bound on the total variation. The complexity of this generalization immediately implies the complexity of a so-called Multi Stage Knapsack Problem subject to Bounded Variation, which was studied further in a recent master's thesis [Mai25] supervised by this author. Finally, this section closes with some remarks on other, loosely related problems.

To begin, consider the following generalized version of (BND): Given a set family $\mathcal{I}$ that lives on the set of switches, and upper bounds $K_t^I$ for all stages $t \in \{0, ..., T\}$ and all sets $I \in \mathcal{I}$, there now is a selection constraint for each stage-set pair. The resulting problem thus reads

$$
\begin{aligned}
(\mathrm{BND}_{\mathcal{I}}^{\mathrm{SEL}}) \qquad \max \quad & \sum_{i=1}^{n} \sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} \\
\text{s.t.} \quad & \mathrm{Var}(x_1, ..., x_T) \leq S \\
& \sum_{i \in I} x_t^{(i)} \leq K_t^{(I)} \qquad I \in \mathcal{I},\ t = 0, ..., T \\
& x_t^{(i)} \in \{0, 1\} \qquad i = 1, ..., n,\ t = 0, ..., T\ ,
\end{aligned}
$$

where $S \in \mathbb{N}_{\geq 0}^r$ is a bound on the variation with appropriate dimension. Observe that the problem (BND) is a special case of $(\mathrm{BND}_{\mathcal{I}}^{\mathrm{SEL}})$ with $\mathcal{I} = \{\{1, ..., n\}\}$. Moreover, it is easy to recognize that the proof of Theorem 4.2.5 can be adapted to show the strong $\mathcal{NP}$-hardness of $(\mathrm{BND}_{\mathrm{SW}}^{\mathrm{SEL}})$ where each feasible set $X_t$ is instead given by

$$
X_t = \left\{ x_t \in \{0, 1\}^n \colon \sum_{i=1}^{n} x_t^{(i)} \leq 1 \right\}\ .
$$

In fact, it suffices to simply omit the introduction of the $T + 1$ dummy switches to adapt the construction to an upper bound in the selection problem. This immediately implies

**Corollary 4.3.1.** The problem $(\mathrm{BND}_{\mathcal{I}}^{\mathrm{SEL}})$ with switch-wise variation is strongly $\mathcal{NP}$-hard, even if $\mathcal{I} := \{\{1, ..., n\}\}$, $K_t^{\{1,...,n\}} := 1$ for all $t$ and $S_i = 2$ for $i = 1, ..., n$.

Actually, the problem is generally hard, independent of the type of variation, as a reduction from Stable Set shows.

**Theorem 4.3.1.** The problem ($\text{BND}_{\mathcal{I}}^{\text{SEL}}$) is strongly $\mathcal{NP}$-hard for switch-wise, time-wise and total variation.

*Proof.* Let an instance $G = (V, E)$, $k \in \mathbb{N}_{\geq 0}$, of Stable Set be given. Now, define an instance of ($\text{BND}_{\mathcal{I}}^{\text{SEL}}$) with $n := |V|$ switches and $|E|$ stages, i.e., $T := |E| - 1$. The set family $\mathcal{I}$ is defined as the edge set of the graph $\mathcal{I} := E = \bigcup_{t=0}^{T} e_t$ where $e_t$ denotes the $t$-th edge in $E$ and the selection bounds are given by

$$K_t^{e_j} := \begin{cases} 1 & \text{for } j = t \\ 2 & \text{for } j \neq t \end{cases} \quad \text{for all } t = 0, ..., T \, .$$

Note that this definition causes all selection bounds other than the one for the edge $e_t$ to become redundant at stage $t$. Depending on the considered type of variation, let $S$ be defined as the zero vector of dimension $r$. This has the effect that all three types of variations actually agree with one another so that the specific type does not matter. In particular, a feasible solution must satisfy $x_0^{(i)} = x_1^{(i)} = ... = x_T^{(i)}$ for each switch $i \in \{1, ..., n\}$. Finally, define the objective by $c_t^{(i)} := 1$ for all $i = 1, ..., n$ and all $t = 0, ..., T$. It is not difficult to see that for any feasible solution of ($\text{BND}_{\mathcal{I}}^{\text{SEL}}$), the set of active switches, i.e., switches that are not entirely zero for all stages, corresponds to a stable set in $G$ (and vice versa). Thus, $G$ contains a stable set of size $k$ or larger if and only if the defined instance of ($\text{BND}_{\mathcal{I}}^{\text{SEL}}$) has an optimal value of at least $k(T + 1)$. □

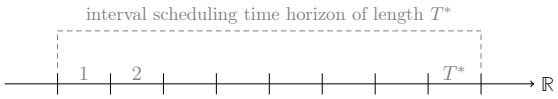
Although the previous theorem and its proof cover all problem versions at the same time, it does not make Corollary 4.3.1 redundant, as the latter shows the intractability of the problem with switch-wise variation for the special case where $\mathcal{I} = \{\{1, ..., n\}\}$ and is thus a stronger statement.

Also for total variation, there is a special case of ($\text{BND}_{\mathcal{I}}^{\text{SEL}}$) that is strongly $\mathcal{NP}$-hard. More precisely, the problem remains $\mathcal{NP}$-hard, even if $\mathcal{I}$ is a laminar set family. A family of subsets $\mathcal{F}$ over some ground set $E$ is a *laminar family*, if for any two sets $F_1, F_2 \in \mathcal{F}$ holds that either one set is a subset of the other, i.e., $F_1 \subseteq F_2$, or the sets have nothing in common, i.e., $F_1 \cap F_2 = \emptyset$. Laminar family sets are computationally easy in the sense that the corresponding linear optimization problem can be solved in polynomial time. In fact, it is not uncommon that $\mathcal{NP}$-hard problems become polynomially solvable for laminar family sets. However, it turns out that even for such a simple set family, the problem ($\text{BND}_{\mathcal{I}}^{\text{SEL}}$) remains strongly $\mathcal{NP}$-hard.

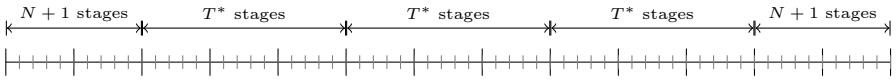
**Theorem 4.3.2.** The problem ($\text{BND}_{\mathcal{I}}^{\text{SEL}}$) with total variation is $\mathcal{NP}$-hard in the strong sense, even if $\mathcal{I}$ is a laminar set family.

*Proof.* In the following, Interval Selection (see Problem 4.2.1) from Section 4.2.2 is polynomially reduced to an instance of problem ($\mathrm{BND}^{\mathrm{SEL}}_{\mathcal{I}}$) with total variation and a laminar set family $\mathcal{I}$. Then, the claim follows due to Theorem 4.2.4.

So, let an instance of Interval Selection as in Theorem 4.2.4 be given, i.e., it consists of $N$ jobs, $m = 3$ machines, as well as intervals $I_{ij}$ with length two and integer endpoints for each job $i = 1, ..., N$ and each machine $j = 1, 2, 3$. Just as in the proof of Theorem 4.2.5, let $T^*$ denote the smallest length of the time horizon on the real line that contains all of these intervals and divide this time horizon into $T^*$ segments of length one, so that the intervals $I_{ij}$ can be interpreted as subsets of $\{1, ..., T^*\}$.

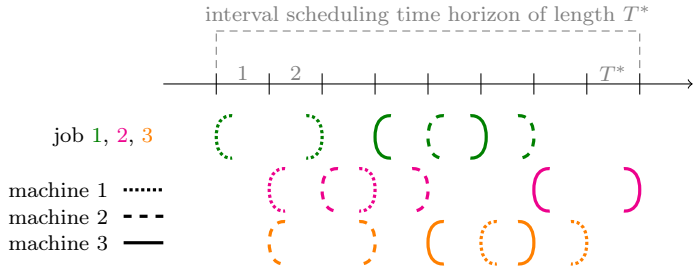



Now, construct an instance of ($\mathrm{BND}^{\mathrm{SEL}}_{\mathcal{I}}$) with $2(N + 1) + 3T^*$ stages, i.e., define $T := 2(N + 1) + 3T^* - 1$. This can be seen as concatenating three copies of the time horizon obtained from Interval Scheduling, where now further $N+1$ stages are added both in the front and in the back. The idea behind this is the same as before: the scheduling decisions on the machines are each modeled in one dedicated copy of the original time horizon. The additional $N+1$ stages in the front and in the back are auxiliary stages that serve two purposes: The first and last stage will serve to model a fixation to zero, while the remaining stages will be used in order to design an instance so that solutions whose objective value surpasses a certain threshold $\gamma$ (which will be specified later) are forced to have a specific structure.

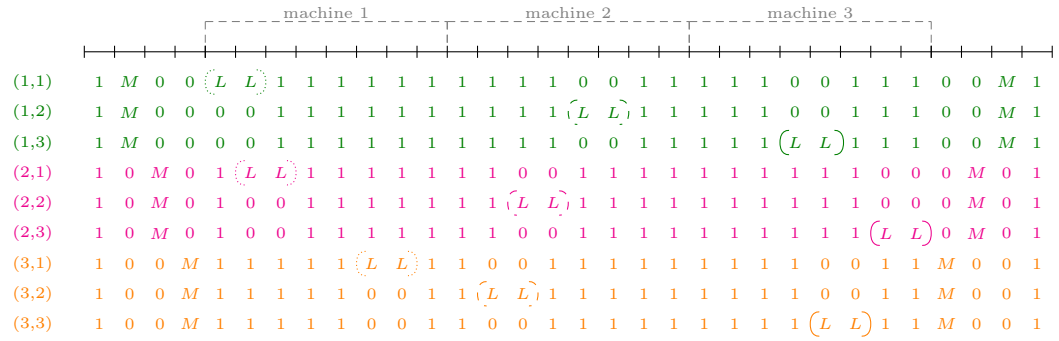



Next, introduce a switch denoted by the tuple $(i, j)$ for each pair $i, j$ of jobs $i = 1, ..., N$ and machines $j = 1, 2, 3$. The bound on the total variation is given by $S := 2n$, where $n$ denotes the number of switches, i.e., $n := 3N$ and thus $S = 6N$. For each switch $(i, j)$, the corresponding objective coefficients are defined by

$$c_t^{(i,j)} = \begin{cases} L & \text{for } t \in \bar{I}_{ij} \\ M & \text{for } t \in \{i, T - i\} \\ 0 & \text{for } t \in \bar{I}_{ij'} \quad \text{with } j' \in \{1, 2, 3\} \setminus \{j\} \\ 0 & \text{for } t \in (\{1, ..., N + 1\} \setminus \{i\}) \cup (\{T - N, ..., T - 1\} \setminus \{T - i\}) \\ 1 & \text{otherwise ,} \end{cases}$$




(a) An instance of Interval Selection with three jobs




(b) Illustration of the objective of the instance of $(\mathrm{BND}^{\mathrm{SEL}}_{\mathcal{I}})$

Figure 4.7: Example for the construction of the instance $(\mathrm{BND}^{\mathrm{SEL}}_{\mathcal{I}})$

where $\bar{I}_{ij} := I_{ij} + (j-1)T^* + N$ is the interval $I_{ij}$ shifted to the $j$-th copy of the Interval Scheduling time horizon and $M >> L$ are 'large enough' positive numbers. More precisely, it suffices to define $L := 3T^* - 3$ and $M := 2L + 3T^* - 3$ which is, in particular, polynomially bounded in the input of $(\mathrm{BND}^{\mathrm{SEL}}_{\mathcal{I}})$. To ease the understanding of the defined objective, an example is provided in Figure 4.7b below.

The laminar set family $\mathcal{I}$ on the set of switches is given by

$$\mathcal{I} := \Phi_1 \cup \Phi_2 \cup \ldots \cup \Phi_N \cup \Phi \,,$$

where the following shorthand notations are used

$$\Phi_i := \{(i,1),(i,2),(i,3)\} \;\text{ for } i = 1,\ldots,N \;\text{ and }\; \Phi := \{(i,j) \colon i = 1,\ldots,N,\; j = 1,2,3\} \,,$$

i.e., the family contains the full set of switches and all triples corresponding to the same job. Finally, the selection bounds are defined as

$$K_t^{\Phi_i} := \begin{cases} 2 & \text{for } t \in \{1,\ldots,T-1\} \setminus \left\{\bar{I}_{i1} \cup \bar{I}_{i2} \cup \bar{I}_{i3}\right\} \\ 3 & \text{otherwise} \end{cases} \qquad \text{for } i = 1,\ldots,N$$

and

$$K_t^{\Phi} := \begin{cases} 0 & \text{for } t \in \{0, T\} \\ 2N+1 & \text{otherwise} \end{cases} .$$

In the following, it is shown that the given Interval Scheduling instance is a yes-instance if and only if the defined instance of $(\text{BND}_{\mathcal{I}}^{\text{SEL}})$ has an optimal value of at least $N \cdot \gamma$, where

$$\gamma := 4M + 6L + 6T^* - 8 .$$

So, let there be a feasible schedule for the instance of Interval Scheduling, i.e., a schedule in which each job is scheduled on exactly one machine in a conflict-free way. Now, define a solution for $(\text{BND}_{\mathcal{I}}^{\text{SEL}})$ as follows: Let $j_i \in \{1, 2, 3\}$ denote the machine on which job $i$ is scheduled and define the states of switches $(i, 1), (i, 2)$ and $(i, 3)$ by

$$x_t^{(i,j_i)} := \begin{cases} 1 & \text{for } t \in \bar{I}_{ij_i} \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad x_t^{(i,j)} := \begin{cases} 0 & \text{for } t \in \{0, T\} \\ 1 & \text{otherwise} \end{cases} \quad \text{for } j \neq j_i .$$

This is a feasible solution since each switch $(i, j)$ switches exactly twice, which yields a total variation of $6N$. It is easy to verify that the solution satisfies

$$x_t^{(i,1)} + x_t^{(i,2)} + x_t^{(i,3)} = 2 \;\text{ for all } t \in \{1, ...., T-1\} \setminus \left\{\bar{I}_{i1} \cup \bar{I}_{i2} \cup \bar{I}_{i3}\right\} ,$$

so that the selection constraints on the sets $\Phi_i, i = 1, ..., N$ are fulfilled. Note that these particular constraints only exists at stages where no interval $\bar{I}_{ij}$ of job $i$ is located. Furthermore, since the schedule is conflict-free and the defined solution $x$ is fixed to zero at the first and last stage, the selection constraints for the set $\Phi$ are satisfied as well:

$$\sum_{i=1}^{N} \sum_{j=1}^{3} x_t^{(i,j)} \leq 0 \quad \text{for } t \in \{0, T\} \quad \text{and} \quad \sum_{i=1}^{N} \sum_{j=1}^{3} x_t^{(i,j)} \leq 2N+1 \text{ otherwise} .$$

By definition of $x$, each group of switches $(i, 1)$,$(i, 2)$,$(i, 3)$ for $i = 1, ..., N$ contributes exactly $\gamma$ to the objective value. Hence, the objective value of this solution is exactly $N \cdot \gamma$.

Conversely, let a feasible solution $x$ of $(\text{BND}_{\mathcal{I}}^{\text{SEL}})$ be given that achieves an objective value of $N \cdot \gamma$ or better. First, note that because of the fixation to zero in the first and last stage, any switch that is not inactive over all stages must have an even total variation of at least two on its own. Due to the selection constraints for the set $\Phi_i$ at stages $t \in \{1, ..., T-1\} \setminus \left\{\bar{I}_{i1} \cup \bar{I}_{i2} \cup \bar{I}_{i3}\right\}$, it is impossible to collect more than $4M$ with each group $(i, 1), (1, 2), (i, 3)$ of switches. But, since the solution $x$ achieves an objective value of at least $N \cdot \gamma > 4NM$ and $M >> L$, exactly two of those three switches must be active at the respective stage $t = i$ and

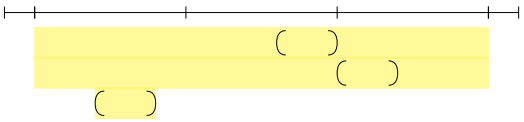
two must also be active at $t = T - i$. In fact, the latter switches must be the same. More precisely, the solution has the following structure:

($\diamond$) Exactly two switches $(i, j_1), (i, j_2)$ are active only at stages $t = i, ...., T - i$ and the third switch $(i, j_3)$ is active at stages $t \in \bar{I}_{ij_3}$ only.

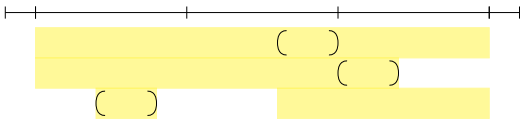
Here, it is $j_1 \neq j_2 \neq j_3 \in \{1, 2, 3\}$ and the (partial) solution on these three switches with structure ($\diamond$) achieves an objective value of $\gamma$ with a total variation of 6. See Figure 4.8a for an illustration of the structure ($\diamond$).

Recall that the intervals $\bar{I}_{i1}$, $\bar{I}_{i2}$, $\bar{I}_{i3}$ are each located within one dedicated copy of the original time horizon from Interval Selection. Therefore, it is impossible for more than two active switches to overlap at a stage $t$ that is not in $\bar{I}_{i1} \cup \bar{I}_{i2} \cup \bar{I}_{i3}$, as this would violate the selection constraint for the set $\Phi_i$ (see Figure 4.8d). Any (partial) solution that does not have the structure ($\diamond$) either achieves the same objective value, but has a variation of at least 8, whereas the solution in ($\diamond$) has a variation of only 6. Or, the solution also has a variation of 6 (or less), but then, it achieves an objective value less than $\gamma$. See Figures 4.8b and 4.8c for an illustration.

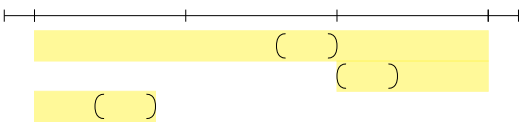
Consequently, the fact that solution $x$ achieves an objective value of at least $N \cdot \gamma$ implies that any group $(i, 1), (1, 2), (i, 3)$ of switches must have the structure ($\diamond$).



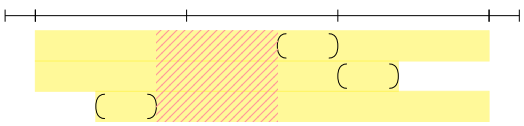
(a) A solution with structure ($\diamond$), objective value $\gamma$ and total variation of 6



(b) A solution with objective value $\gamma$ and total variation of 8



(c) A solution with total variation of 6 but an objective value less than $\gamma$



(d) Such a solution violates the selection constraints for $\Phi_i$

Figure 4.8: Possible structures other than ($\diamond$), the color yellow marks the states at which the switches are active

With this knowledge, a feasible schedule for the instance of Interval Scheduling can now be defined as follows: For each $i = 1, ..., N$, schedule job $i$ on machine $j$ if the switch $(i, j)$ is active at stages $\bar{I}_{ij}$ only. This way each job is obviously scheduled exactly once. The fact that the scheduled jobs do not cause any conflicts on the machines follows from the remaining selection constraints for the set $\Phi$ at the

stages $t = 2, ..., T-1$:

$$\sum_{i=1}^{N} x_t^{(i,1)} + x_t^{(i,2)} + x_t^{(i,3)} \leq 2N + 1 \qquad \text{for } t = 1, ..., T-1 \ .$$

Now, consider a machine $j \in \{1, 2, 3\}$ and let $I_j \subseteq \{1, ..., N\}$ refer to all jobs that have been scheduled on machine $j$. Since the solution $x$ has the structure ($\diamond$), the above constraint constraint implies

$$\sum_{i \in I_j} x_t^{(i,j)} \leq 1 \qquad \text{for } t = 1, ..., T-1 \text{ and } j = 1, 2, 3 \ .$$

In other words, the intervals scheduled on the same machine are conflict-free. Consequently, the instance of Interval Scheduling is a yes-instance. This concludes the proof. □

Another possible generalization of (BND) is to consider a different type of combinatorial constraint for the feasible sets $X_t$. A natural generalization of a selection constraint is a knapsack constraint. However, this clearly results in an at least $\mathcal{NP}$-hard problem due to the weak $\mathcal{NP}$-hardness of the binary knapsack problem. Nevertheless, an analogous reduction from Stable Set as in the proof of Theorem 4.3.1 actually shows that presence of the bound on the variation leads to a further increase in complexity.

**Lemma 4.3.1.** The following problem

$$\text{(BND-KP)} \qquad \begin{cases} \max & \displaystyle\sum_{i=1}^{n} \sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} & & \\ \text{s.t.} & \text{Var}\,(x_0, ..., x_T) \leq S & & \\ & \displaystyle\sum_{i=1}^{n} a_t^{(i)} x_t^{(i)} \leq b_t & & t = 0, ..., T \\ & x_t^{(i)} \in \{0, 1\} & i = 1, ..., n, & t = 0, ..., T \end{cases}$$

is strongly $\mathcal{NP}$-hard for all three types of variation.

Note that not only the profits, but also the weights of the items and the knapsack capacity change over time. The $\mathcal{NP}$-hardness proof and a more detailed study, including dynamic programming schemes which solve the problem in pseudo-polynomial time if either the number $n$ or the number $T$ is bounded, can be found in a recent master thesis [Mai25].

Finally, this section closes with a third shift in perspective and briefly studies the problem (BND) for a fourth type of variation, defined as follows:

**Definition 4.3.1.** Define the variation function $\mathrm{Var}_\infty : \times_{t=1}^{T} X^{(t)} \to \mathbb{R}_{\geq 0}$ by

$$\mathrm{Var}_\infty(x) := \sum_{t=1}^{T} ||x_{t-1} - x_t||_\infty := \sum_{t=1}^{T} \max_{i=1,\dots,n} |x_{t-1}^{(i)} - x_t^{(i)}| \,.$$

Note that this type of variation $\mathrm{Var}_\infty(x)$ actually agrees with total variation and switch-wise variation if there is only one switch. For multiple switches, however, measuring the variation in this way is less precise than before, since a bound on it only restricts the total number of times that two consecutive solutions $x_t$ and $x_{t+1}$ are not identical, but it does not bound the number of different consecutive entries (unless $n = 1$). The resulting version of (BND) reads

$$(\mathrm{BND}_\infty) \qquad \begin{cases} \max & \displaystyle\sum_{i=1}^{n}\sum_{t=0}^{T} c_t^{(i)} x_t^{(i)} & \\ \text{s.t.} & \displaystyle\sum_{t=1}^{T} ||x_{t-1} - x_t||_\infty \leq S & \\ & \displaystyle\sum_{i=1}^{n} x_t^{(i)} \leq K_t & t = 0, ..., T \\ & x_t \in \{0,1\}^n & t = 0, ..., T \end{cases}$$

**Lemma 4.3.2.** The problem $(\mathrm{BND}_\infty)$ can be solved efficiently with a dynamic programming scheme. A straightforward implementation of this dynamic programming scheme leads to a running time of $\mathcal{O}(T^3 \cdot S \cdot n \log n)$.

The mentioned scheme is devised in the following. It recursively computes the optimal values $c^*(t_2, s)$ of problem $(\mathrm{BND}_\infty)$ restricted to time points $\{0, ..., t_2\}$ such that the variation $\mathrm{Var}_\infty$ does not exceed $s \in \{0, ..., S\}$, i.e.,

$$\begin{cases} \max & \displaystyle\sum_{i=1}^{n}\sum_{t=0}^{t_2} c_t^{(i)} x_t^{(i)} & \\ \text{s.t.} & \displaystyle\sum_{t=1}^{t_2} ||x_{t-1} - x_t||_\infty \leq s & \\ & \displaystyle\sum_{i=1}^{n} x_t^{(i)} \leq K_t & t = 0, ..., t_2 \\ & x_t \in \{0,1\}^n & t = 0, ..., t_2 \end{cases}$$

for all $t_2 = 1, ..., T$ and $s \in \{0, ..., S\}$. The initial values are defined by

$$c^*(0, s) := \max_{x \in X_0} \sum_{i=1}^{n} c_0^{(i)} x^{(i)} \qquad \text{for } s = 0, ..., S \ .$$

The idea behind the recursive computation of the optimal values of the subproblems is described in the following: Consider the optimal value $c^*(t_2, s)$ for some $t_2 \in \{1, ..., T\}$. An optimal solution restricted to stages $\{0, ..., t_2\}$ and with a bound of $s$ on the variation $\mathrm{Var}_\infty$ can be obtained by considering all possibilities for the preceding stage $t_1 < t_2$ where the previous change occurred. Either there is no such stage $t_1$ at all, which means that the solution must be constant over all stages $\{0, ..., t_2\}$, hence the optimal value is given by $\bar{c}(0, t_2)$. Otherwise, there are two possibilities where the stage $t_1$ may be located: If $t_1$ is the stage right before $t_2$, i.e., $t_1 = t_2 - 1$, then one may simply extend an optimal solution restricted to stages $\{0, ..., t_2 - 1\}$ with a bound of $s - 1$ on the variation by an optimal solution for the classical selection problem restricted to stage $t_2$. If the stage $t_1$ is not right before $t_2$, then the sequence of solutions from stage $t_1 + 1$ until $t_2$ must be constant, due to the definition of $t_1$. Consequently, an optimal solution is obtained by extending an optimal solution restricted to stages $\{0, ..., t_1\}$ with a bound of $s - 1$ on the variation with the best constant sequence for the stages $t_1 + 1, ...., t_2$. More formally, the scheme recursively computes

$$c^*(t_2, s) = \max \begin{cases} \bar{c}(0, t_2) \ , \\ \max_{(t_1, t_2)} c^*(t_1, s-1) + \bar{c}(t_1 + 1, t_2) \quad \text{if } s \geq 1 \end{cases}$$

for all $t_2 \in \{1, ...., T\}$. In the recursion formula above, the pair $(t_1, t_2) \in \{0, ..., T\}^2$ denotes a pair of indices with $t_1 < t_2$, and the values $\bar{c}(t_1 + 1, t_2)$ give the best value that can be achieved with a feasible constant sequence $x_{t_1+1} = ... = x_{t_2}$. More formally, these values are defined by

$$\bar{c}(t_1 + 1, t_2) := \max \left\{ \sum_{i=1}^{n} \left( \sum_{t=t_1+1}^{t_2} c_t^{(i)} \right) x^{(i)} \ : x \in \{0, 1\}^n, \ \sum_{i=1}^{n} x^{(i)} \leq \min_{t_1 \leq t \leq t_2} K_t \right\} \ .$$

Note that for $t_1 = t_2 - 1$, the value $\bar{c}(t_2, t_2)$ is given by

$$\max\{c_{t_2}^\top x \colon x \in \{0, 1\}^n, \mathbb{1}_n^\top x \leq K_{t_2}\} \ .$$

Then, the optimum value of $(\mathrm{BND}_\infty)$ is given by $c^*(T, S)$ and an optimal solution can be constructed from the recursion.

Concerning the running time of the proposed dynamic programming scheme, the values $\bar{c}(t_1, t_2)$ can clearly be computed in time $\mathcal{O}(n \log n)$ as the associated subproblem reduces to a classical selection problem. Overall, there is a quadratic number $\mathcal{O}(T^2)$ of such values $\bar{c}(t_1, t_2)$. Hence, the computation of the (outer) maximum in the recursive formula needs time $\mathcal{O}(T^2 \cdot n \log n)$ and since these need to be computed for all $t$ and all $s$, the total running time can be estimated by $\mathcal{O}(T^3 \cdot S \cdot n \log n)$.

**Remark 4.3.1.** The previous dynamic programming scheme still runs in polynomial time for any other combinatorial feasible set $X_t$, for which the computation of the values $\bar{c}(t_1, t_2)$ remains tractable. More precisely, if the problem

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} \left( \sum_{t=t_1}^{t_2} c_t^{(i)} \right) x^{(i)} \\ \text{s.t.} \quad & x \in \bigcap_{t=t_1}^{t_2} X_t \end{aligned}$$

is tractable for any pair $t_1 \leq t_2 \in \{0, ..., T\}^2$, then the dynamic programming scheme yields an optimal solution in polynomial time.

This concludes the discussion of possible generalizations. The next chapter will investigate the bounded problem (BND) in a more abstract form. More precisely, it will take an oracle-based approach which assumes the existence of a linear optimization problem for the underlying combinatorial problem and it will study the oracle complexity of the problem (BND), i.e., the number of oracle calls needed to solve it.

# Chapter 5

# Oracle-based Linear Optimization subject to Bounded Variation

The variety of its possible applications is the reason for which it is desirable to study the problem (BND) in its most abstract form, i.e., for arbitrary feasible sets. For a high level of abstraction, this chapter therefore relies on no particular properties of the feasible sets $X_t$. Instead, it is assumed that the feasible set $X_t$ for each stage is given via a *linear optimization oracle* $\Upsilon_t$. The latter can be considered an abstract algorithm, or a black box, which for a respective set $X_t$ and any objective $c \in \mathbb{R}^n$ returns an optimal solution of the linear problem

$$(\mathrm{LO}_t) \qquad \begin{cases} \max & c^\top x \\ \text{s.t.} & x \in X_t \end{cases}$$

in a single operation. Thus, the oracle $\Upsilon_t$ for the problem $(\mathrm{LO}_t)$ runs in constant time, independent of the complexity class to which $(\mathrm{LO}_t)$ belongs. It can be seen as a theoretical device serving as a placeholder for an efficient $(\mathrm{LO}_t)$-algorithm. Note that the linear optimization over a set $X_t$ is equivalent to the optimization over the convex hull $\mathrm{conv}(X_t)$, which is a polytope. Thus, the polynomial equivalency between linear optimization and separation holds (Theorem 2.1.4). As a consequence, every statement in this chapter that assumes a linear optimization oracle for $X_t$ will also hold with a separation oracle for the respective convex hull instead.

The aim of this chapter is to investigate the complexity of (BND) in the setting where each feasible set $X_t$ is given by an $(\mathrm{LO}_t)$-oracle $\Upsilon_t$, with occasional reference to the penalty problem (P) whenever relevant implications arise. In simpler terms, it will try to answer the question about the relative complexity of the problem (BND) compared to the complexity of the respective underlying combinatorial problems. This essentially boils down to whether an oracle-polynomial algorithm for (BND) with respect to the $(\mathrm{LO}_t)$-oracles $\Upsilon_t$ exists. Recall from the preliminaries in Chapter 2 that an oracle-polynomial algorithm is an algorithm which uses a polynomial number of subroutines corresponding to oracle calls.

As mentioned earlier, an ($\mathrm{LO}_t$)-oracle $\Upsilon_t$ for some set $X_t$ can be seen as a placeholder for an efficient ($\mathrm{LO}_t$)-algorithm. Thus, the existence of an oracle-polynomial algorithm for (BND) implies the tractability of the problem (BND) in the classical sense under the condition that every problem ($\mathrm{LO}_t$) that is solved via an oracle call is in fact tractable.

**Remark 5.0.1.** Suppose that there exists an oracle-polynomial algorithm $\mathcal{A}$ for the problem (BND) with either switch-wise, time-wise or total variation. For each $t$, let $C_t \subseteq \mathbb{R}^n$ denote the set of objectives for which $\mathcal{A}$ calls the ($\mathrm{LO}_t$)-oracle $\Upsilon_t$.

If the problem ($\mathrm{LO}_t$) for each stage $t$ is tractable when restricted to objectives in $C_t$, then the respective problem (BND) can be solved in polynomial time.

According to the last remark, an oracle-polynomial algorithm for (BND) combined with the tractability of the problems ($\mathrm{LO}_t$) when restricted to the objectives in $C_t$ suffices to derive the tractability of (BND). In the special case in which the tractability of the problem ($\mathrm{LO}_t$) is ensured for *every* possible objective $c \in \mathbb{R}^n$, the corresponding set $X_t$ will be called *simple*. For example, the set of all feasible solutions of a Selection problem is simple, since an optimal solution for any objective can be obtained by the simple greedy algorithm described in the preliminaries.

Soon, it will become clear that the search for an oracle-polynomial algorithm for the problem (BND) is futile, unless at least one of the problem parameters is a fixed constant. This also naturally opens up a path into to the field of fixed-parameter tractability. Recall that a problem is called fixed-parameter tractable if it admits an $FPT$-algorithm, i.e., an algorithm which solves any instance $(x, k)$ of the problem with fixed parameter $k$ in time bounded by $\mathcal{O}(f(k) \cdot size(x)^c)$, where $f$ is a computable function and $c$ is a constant. In particular, an $FPT$-algorithm runs in polynomial time, if $k$ is fixed. However, note that an algorithm which runs in polynomial time if $k$ is fixed is not necessarily an $FPT$-algorithm. Further recall that the running time of an $FPT$-algorithm is sometimes also stated as $\mathcal{O}^*(f(k))$, i.e., ignoring the polynomial part and just focusing on the part involving the fixed parameter.

This concludes the preliminaries. The remainder of this chapter is structured as follows: In the first section, the problem (BND) is studied for the case in which the number of switches $n$ is considered a fixed parameter. Conveniently, for switch-wise and total variation, the dynamic programming schemes from the previous chapter can be adapted without too much trouble to fit into the setting of this chapter. As a result, they yield oracle-polynomial algorithms for the problem (BND). For time-wise variation, a dynamic programming scheme is developed in this chapter. Also this yields an oracle-polynomial algorithm for (BND). Afterwards, the problem (BND) is studied under the assumption that either the bound on the variation or the number of stages are considered fixed parameters, instead of $n$. Beginning with a complexity-theoretical approach in Section 5.2, the non-existence of an oracle-

polynomial optimization algorithm for (BND) with switch-wise variation is deduced from its $\mathcal{NP}$-hardness if $\mathcal{P} \neq \mathcal{NP}$ is assumed. In the third and final Section 5.3, the answer to the main question of this chapter will be given. Indeed, it turns out that even if both $S$ and $T$ are fixed parameters, the feasibility of (BND) cannot be decided using only a polynomial number of ($\mathrm{LO}_t$)-oracle calls, independent of the type of variation. Due to this result, the remainder of Section 5.3 will be dedicated to the presentation of a positive result regarding the oracle-polynomial solvability of (BND). To be precise, assuming the existence of a different linear optimization oracle $\Psi$ for the so-called Optimal Constant Extension Problem (OCE), an oracle-algorithm for (BND) with time-wise or total variation can be devised, which will become oracle-polynomial for constant $S$ and $T$ with respect to the oracle $\Psi$. Finally, some tractable cases of (OCE) will be presented which will, in turn, imply the tractability of the corresponding problem (BND) with either time-wise or total variation.

## 5.1 Oracle-Polynomial Solution for Constant $n$

In this section, the problem (BND) is studied for all three types of variation regarding the existence of an oracle-polynomial algorithm under the assumption that each set $X_t$ is given via an ($\mathrm{LO}_t$)-oracle $\Upsilon_t$ and that the number of switches $n$ is constant. Although the latter assumption implies that the size of each feasible set $|X_t|$ is bounded by a constant, namely $2^n$, this does not render the problem (BND) trivial. In fact, while it is possible to enumerate all binary elements in $\{0,1\}^n$ efficiently for constant $n$, it still remains to verify their membership regarding a feasible set $X_t$.

Note that an ($\mathrm{LO}_t$)-oracle $\Upsilon_t$ for the set $X_t$ can be used to verify the membership of any binary vector regarding $X_t$, i.e., it can be used as a membership oracle.

**Remark 5.1.1.** Let $X_t \subseteq \{0,1\}^n$ be given via an ($\mathrm{LO}_t$)-oracle $\Upsilon_t$, $x \in \{0,1\}^n$ and define $c_x$ via

$$(c_x)_i := \begin{cases} 1 & \text{if } x_i = 1 \\ -1 & \text{if } x_i = 0 \end{cases} \qquad i = 1, ..., n \ .$$

Then $x \in X_t$ if and only if the oracle $\Upsilon_t$ returns the optimal value $\mathbb{1}_n^\top x$ when it is called for the objective $c_x$. Thus, a linear optimization oracle includes a membership oracle by default.

Now, observe that the dynamic programming scheme which was devised for (BND) with switch-wise variation (Theorem 4.2.6) can actually be adapted to work for combinatorial sets $X_t$ that are given via ($\mathrm{LO}_t$)-oracles $\Upsilon_t$. Indeed, it suffices to replace the enumeration of all elements in the set $X_t$ with the enumeration of all binary elements in $\{0,1\}^n$ and to verify of their membership in $X_t$ with respective oracle subroutines.

This results in an oracle algorithm, and since the dynamic programming scheme from Section 4.2.2 runs in $\mathcal{O}\left((\max_{t=0,\dots,T}|X_t|)^2 \cdot (T+1)^{n+1}\right)$ time, this adapted dynamic programming scheme with oracle subroutines runs in time $\mathcal{O}\left(2^{2n} \cdot (T+1)^{n+1}\right)$.

**Lemma 5.1.1.** The problem (BND) with switch-wise variation where each set $X_t$ is given via an ($\mathrm{LO}_t$)-oracle $\Upsilon_t$ can be solved by an oracle algorithm which runs in time $\mathcal{O}\left(2^{2n} \cdot (T+1)^{n+1}\right)$. If $n$ is constant, the algorithm runs in oracle-polynomial time. This also holds true if $X_t$ for $t \in \{0,..,T\}$ is given via only a membership oracle.

Note that the above lemma implies that (BND) with switch-wise variation and fixed parameter $n$ is slice-wise polynomial, i.e., in *XP*, but it does *not* imply that it is (oracle-)fixed-parameter tractable.

Furthermore, recall that the penalty problem can be reduced to the bounded problem by enumeration of all possible bounds. This allows for the following observation.

**Remark 5.1.2.** For any penalty parameter $\lambda \in \mathbb{R}^n_{>0}$, the penalty problem (P) with switch-wise variation can be reduced to the problem (BND) by enumeration of all $T^n$ possible switch-wise bounds $S \in \mathbb{N}^n$. For a constant $n$, the reduction is polynomial. Hence, under the assumptions of the previous lemma, the corresponding penalized problem can be solved oracle-polynomially, too.

Recall Remark 5.0.1: if each ($\mathrm{LO}_t$)-oracle call in an oracle-polynomial algorithm can be substituted by a respective efficient ($\mathrm{LO}_t$)-optimization algorithm, the result is a polynomial time algorithm. For simple sets $X_t$, the existence of an efficient ($\mathrm{LO}_t$)-algorithm is ensured for any possible objective, therefore an immediate consequence of the latter lemma is:

**Corollary 5.1.1.** If $n$ is constant and each set $X_t$, $t = 0,...,T$, is simple, the problem (BND) with switch-wise variation can be solved in polynomial time.

This concludes the study of the problem (BND) with switch-wise variation. Next, the problem (BND) with total variation is studied regarding the same question. Recall that in Section 4.2.3, a dynamic programming scheme with running time $\mathcal{O}\left(T \cdot S \cdot (\max_{t=0,\dots,T}|X_t|)^2\right)$ for (BND) with total variation was devised. With analogous argumentation, this scheme can be turned into an oracle algorithm for (BND) where each $X_t$ is given via an ($\mathrm{LO}_t$)-oracle $\Upsilon_t$.

**Lemma 5.1.2.** The problem (BND) with total variation where each set $X_t$ is given via an $(\mathrm{LO}_t)$-oracle $\Upsilon_t$ can be solved by an oracle algorithm which runs in time $\mathcal{O}\left(T \cdot S \cdot 2^{2n}\right)$. If $n$ is fixed, the algorithm is oracle-polynomial. In particular, the problem is (oracle-)fixed parameter tractable for the parameter $n$ with $\mathcal{O}^*(2^{2n})$. This also holds true if $X_t$ for $t \in \{0, .., T\}$ is given only via a membership oracle.

Once more, recall that the penalty problem can be reduced to the bounded problem by enumeration of all possible bounds.

**Remark 5.1.3.** For any penalty parameter $\lambda \in \mathbb{R}_{>0}$, the penalty problem (P) with total variation can be reduced to the problem (BND) by enumeration of all $Tn$ possible total bounds $S \in \mathbb{N}_{\geq 0}$. Since the reduction is polynomial, the latter lemma implies that under the assumptions of the previous lemma, the corresponding penalized problem can be solved oracle-polynomially, too.

Once again, the immediate consequence of Lemma 5.1.2 is

**Corollary 5.1.2.** If $n$ is constant and each set $X_t$, $t = 0, ..., T$, is simple, then the problem (BND) with total variation can be solved in polynomial time.

An example for a simple set $X_t$ is a set which is given as a list of its elements. Clearly, any linear objective can be optimized over this set by enumeration of its elements which is, in this case, polynomial in the size of the input. As a result, it can be solved by the dynamic programming scheme for total variation which computes the recursive terms by enumeration of elements in $X_t$, i.e., in time $\mathcal{O}\left(T \cdot S \cdot \left(\max_{t=0,...,T} |X_t|\right)^2\right)$, which then is polynomial in the input. Most importantly, here the running time is polynomial without the assumption that $n$ is fixed.

**Corollary 5.1.3.** If each set $X_t$ for $t = 0, ..., T$ is given as a list of its elements, then (BND) with total variation can be solved in polynomial time.

**Remark 5.1.4.** Note that even if each feasible set is given as a list of its elements, the previous dynamic programming scheme for the problem (BND) with switch-wise variation still has a running time that is exponential in $n$ and hence, is only polynomial under the assumption that $n$ is constant. Indeed, unless $\mathcal{P} = \mathcal{NP}$, a result analogous to the latter corollary for (BND) with switch-wise variation is not possible due to its $\mathcal{NP}$-hardness even for polynomial size feasible sets (Theorem 4.2.5).

It now remains to study the third and last problem variant, namely (BND) with time-wise variation. Unlike for switch-wise and total variation, a dynamic programming scheme for time-wise variation was not devised in the previous chapter. This will be taken care of now.

**Lemma 5.1.3.** The problem (BND) with time-wise variation where each set $X_t$ is given via an ($\mathrm{LO}_t$)-oracle $\Upsilon_t$ can be solved by an oracle algorithm which runs in time $\mathcal{O}\left(2^{2n} \cdot T\right)$. If $n$ is fixed, the algorithm is oracle-polynomial. In particular, the problem is (oracle-)fixed parameter tractable for the parameter $n$ with $\mathcal{O}^*(2^{2n})$. This also holds true if $X_t$ for $t \in \{0, .., T\}$ is given only via a membership oracle.

*Proof.* In the following, an oracle algorithm is devised. It is a dynamic programming scheme using subroutines of ($\mathrm{LO}_t$)-oracle calls.

Let $c^*(\bar{t}, b)$ be the optimal value of (BND) with time-wise variation restricted to the stages $0, ..., \bar{t}$ with $x_{\bar{t}} = b$, so that the time-wise variation does not exceed the partial bound $(S_1, ..., S_{\bar{t}}) \in \mathbb{N}_{\geq 0}^{\bar{t}}$. Recall the short notation

$$|x - y| := (|x_1 - y_1|, ..., |x_n - y_n|)^\top$$

for two vectors $x$, $y$ of the same dimension. Within the scheme, the optimal values of the following subproblems for $\bar{t} \in \{0, ..., T\}$ are computed:

$$\begin{aligned} c^*(\bar{t}, b) = \max \quad & \sum_{i=1}^{n} \sum_{t=0}^{\bar{t}} c_t^{(i)} \cdot x_t^{(i)} & \\ \text{s.t.} \quad & \sum_{i=1}^{n} |x_{t-1}^{(i)} - x_t^{(i)}| \leq S_t & t = 1, ..., \bar{t} \\ & x_t \in X_t & t = 0, ..., \bar{t} \\ & x_{\bar{t}}^{(i)} = b_i & i = 1, ..., n \end{aligned}$$

For $\bar{t} \geq 1$, the scheme recursively computes the optimal values $c^*(\bar{t}, b)$ with the help of the following formula:

$$c^*(\bar{t}, b) := \max_{\substack{x \in X_{\bar{t}-1}, \\ |x-b| \leq S_{\bar{t}}}} c^*(\bar{t} - 1, x) + \sum_{i=1}^{n} c_{\bar{t}}^{(i)} \cdot b_i \quad \text{for all } b \in X_{\bar{t}} \,.$$

The optimal value of (BND) is thus given by $\max_{b \in X_T} c^*(T, b)$.

Indeed, the correctness of the recursion formula is easy to see, as an optimal solution for the subproblem restricted to stages $0, ..., \bar{t}$ can be obtained in the following way: Due to the fixation to some $b$ at stage $\bar{t}$, any optimal solution $(x_0^*, ..., x_{\bar{t}}^*)$ with $x_{\bar{t}}^* = b$ of the subproblem for $c^*(\bar{t}, b)$ must not exceed the variation bound $S_{\bar{t}}$

between stages $\bar{t}-1$ and $\bar{t}$, i.e., $|x^*_{\bar{t}-1} - b| \leq S_{\bar{t}}$. Furthermore, there cannot exist any $x \in X_{\bar{t}-1}$ with $|x-b| \leq S_{\bar{t}}$ that achieves an optimal value $c^*(\bar{t}-1, x)$ larger than $c^*(\bar{t}-1, x^*_{\bar{t}-1})$, as this would clearly contradict the optimality of $(x^*_0, ..., x^*_{\bar{t}})$. Hence, $x^*_{\bar{t}-1}$ is a maximizer of $c^*(\bar{t}-1, x)$ subject to $x \in X_{\bar{t}-1}, |x-b| \leq S_{\bar{t}}$.

Finally, the running time of the dynamic programming scheme is analyzed. If each set $X_t$ is given via an ($\mathrm{LO}_t$)-oracle $\Upsilon_t$, then the recursive term $c^*(\bar{t}, b)$ for some pair $\bar{t}$ and $b$ can again be computed by enumeration of all $x \in \{0,1\}^n$ and verifying their membership in $X_t$ by calling $\Upsilon_t$ for $c_x$. Thus, the overall running time can be estimated by $\mathcal{O}\left(2^{2n} \cdot T\right)$. □

As before, the latter lemma allows to draw a similar conclusion for the corresponding penalized problem.

**Remark 5.1.5.** For any penalty parameter $\lambda \in \mathbb{R}^T_{>0}$, the penalty problem (P) with time-wise variation can be reduced to the problem (BND) with time-wise variation by enumeration of all $n^T$ possible time-wise bounds $S \in \mathbb{N}^T_{\geq 0}$. For a constant $T$, the reduction is polynomial and hence, under the assumptions of the previous lemma and the additional assumption that $T$ is constant, the penalized problem can be solved oracle-polynomially, too.

Once again, Lemma 5.1.3 implies

**Corollary 5.1.4.** If $n$ is constant and each set $X_t$, $t = 0, ..., T$, is simple, then (BND) with time-wise variation can be solved in polynomial time.

As in the case of total variation, (BND) with time-wise variation can be solved efficiently if each set $X_t$, $t = 0, ..., T$, is given as a list of its elements, since then the dynamic programming scheme can be implemented to run in time $\mathcal{O}\left(T \cdot \left(\max_{t=0,...,T} |X_t|\right)^2\right)$. Again, in this special case the running time is polynomial in the input and does not depend on $n$ being fixed.

**Corollary 5.1.5.** If each set $X_t$, $t = 0, ..., T$, is given as a list of its elements, then (BND) with time-wise variation can be solved in polynomial time.

To summarize the results of this first section, under the assumption that $n$ is fixed, the problem (BND) is oracle-polynomially solvable if the sets $X_t$ are given via ($\mathrm{LO}_t$)-oracle $\Upsilon_t$ (or alternaltively, a membership oracle). This holds regardless of the type of variation and follows from the combination of an oracle-based membership verification and a respective dynamic programming scheme. Regarding the

relative complexity of (BND) with respect to the complexity of the underlying problems ($\mathrm{LO}_t$), this means that the former is equal to the latter as long as $n$ is a constant. Table 5.1 below provides an overview of the results obtained in this section.

| assumptions \variation | total/ time-wise | switch-wise |
|---|---|---|
| each $X_t$ given via ($\mathrm{LO}_t$)-oracle and $n \in \mathcal{O}(1)$ | oracle-polynomial and even (oracle-)$FPT$ in $n$ | oracle-polynomial |
| each $X_t$ simple and $n \in \mathcal{O}(1)$ | polynomial | |
| each $X_t$ given as list | polynomial | polynomial **only if** $n \in \mathcal{O}(1)$ |

Table 5.1: Summary of results in Section 5.1

It will turn out that the existence of an oracle-polynomial algorithm for (BND) strongly depends on what part of the input is considered a constant. The next section studies the problem (BND) from a complexity-theoretical viewpoint where either the bound $S$ on the respective variation or the number of stages $T$ is constant. Contrary to this section, a polynomial number of ($\mathrm{LO}_t$)-oracle calls will not suffice to solve the problem (BND) with switch-wise variation, and the existence of a respective oracle-polynomial optimization algorithm is ruled out, unless $\mathcal{P} = \mathcal{NP}$.

## 5.2 Complexity-Theoretical Results for Constant $S$ or $T$

In this section, the problem (BND) is studied from a complexity-theoretical point of view regarding the existence of oracle-polynomial algorithms under the assumption that each set $X_t$ is given via an ($\mathrm{LO}_t$)-oracle $\Upsilon_t$, and that the bound on the variation $S$ or the number of stages $T$ is constant.

For the problem (BND) with switch-wise variation there is an immediate consequence obtained from the complexity of ($\mathrm{BND}^{\mathrm{SEL}}_{\mathrm{SW}}$) (see Theorem 4.2.5) and the fact that the feasible set corresponding to a Selection problem is simple.

**Lemma 5.2.1.** Unless $\mathcal{P} = \mathcal{NP}$, the problem (BND) with switch-wise variation cannot be solved using a polynomial number of ($\mathrm{LO}_t$)-oracle calls, even if the feasible set does not change, i.e., $X := X_t$ for all $t$ and $S = 2 \cdot \mathbb{1}_n$.

*Proof.* The claim is proven by contradiction. Suppose that there exists an oracle-polynomial algorithm $\mathcal{A}$ for the problem (BND) with switch-wise variation. By definition, $\mathcal{A}$ uses no more than polynomially many calls to the ($\mathrm{LO}_t$)-oracles. Clearly, $\mathcal{A}$ can be used to efficiently solve the special case of (BND) in which $S = 2 \cdot \mathbb{1}_n$ and $X := X_t := \{x \in \{0,1\}^n \colon \sum_{i=1}^{n} x^{(i)} \leq 1\}$ for all $t$. Note that $X$ is simple as it corresponds to the feasible solutions of a Selection problem. As explained in Remark 5.0.1, any call of an ($\mathrm{LO}_t$)-oracle can be substituted by an efficient algorithm

for the Selection problem (e.g., the greedy algorithm). This results in a polynomial time algorithm for the special case. The existence of the oracle-polynomial algorithm $\mathcal{A}$ therefore contradicts Theorem 4.2.5, unless $\mathcal{P} = \mathcal{NP}$. □

Next, the problem (BND) with time-wise variation is studied. Although the special case with a selection constraint, which was studied in Section 4.2.1 of the previous Chapter 4, is tractable (see Theorem 4.2.2), the problem is generally $\mathcal{NP}$-hard. The $\mathcal{NP}$-hardness will be proven with a reduction from the following decision problem:

**Problem 5.2.1.** Vertex Cover Reconfiguration ([LM17])

**Input:** Bipartite graph $G = (V, E)$, number $k \in \mathbb{N}$,
two vertex covers $R, Q \subseteq V$, $|R|, |Q| \leq k$
**Question:** Is there a *reconfiguration sequence*, i.e., a sequence of vertex covers
$\alpha_1 = R$, $\alpha_2$, ..., $\alpha_N = Q$ with $|\alpha_i| \leq k$, $i = 1, ..., N$,
and $|\alpha_i \Delta \alpha_{i+1}| = 1$ for all $i = 1, ..., N-1$ ?

The complexity of Vertex Cover Reconfiguration has already been established in the literature by exploiting its equivalency to the so-called *cops-and-robber game* and the $\mathcal{NP}$-hardness of computing the treewidth of cobipartite graphs.

**Theorem 5.2.1.** ([LM17], Theorem 3) Vertex Cover Reconfiguration is $\mathcal{NP}$-complete.

A reduction from Vertex Cover Reconfiguration shows

**Theorem 5.2.2.** The problem (BND) with time-wise variation is $\mathcal{NP}$-hard in the strong sense, even if $S = \mathbb{1}_T$, and the feasible set does not change over time, i.e., $X := X_t$ for all $t$, and even if the underlying problem $\max_{x \in X} c_t^\top x$ is tractable for each stage $t = 0, ..., T$.

*Proof.* Let $(G = (V, E), R, Q, k)$ be an instance of Vertex Cover Reconfiguration. Now, define an instance of (BND) with time-wise variation and $n := |V|$ switches. Each feasible set $X_t \subseteq \{0, 1\}^n$ is given by the set $X$ which consists of all characteristic vectors $\chi_C \in \{0, 1\}^n$ that correspond to a vertex cover $C$ of at most size $k$ in the bipartite graph $G$. Define the objective function for (BND) by

$$c_0^{(i)} := \begin{cases} 1 & \text{for } i \in R \\ -1 & \text{else} \end{cases}, \quad c_T^{(i)} := \begin{cases} 1 & \text{for } i \in Q \\ -1 & \text{else} \end{cases},$$

and $c_t := \mathbb{0}_n$ for $0 < t < T$. Due to this definition of $c_t$, all underlying problems $\max_{x \in X} c_t^\top x$ are tractable. Indeed, for $t = 0$ and $t = T$, the corresponding problem is trivially solved by the characteristic vector of the respective vertex cover $R$ and $Q$. For all other stages $0 < t < T$, the problem $\max_{x \in X} \mathbb{0}_n^\top x$ reduces to the decision version of vertex cover in a bipartite graph, which asks whether there exists a vertex cover in $G$ of at most size $k$. Recall that the decision version of Vertex Cover in a bipartite graph can be solved polynomially. In fact, here it is trivial since $R$ (or $Q$) serves as a certificate.

Further, all bounds on the time-wise variation are defined to be one, i.e., $S := \mathbb{1}_T$. Observe that the symmetric difference $|\alpha_t \Delta \alpha_{t+1}|$ of two vertex covers $\alpha_t$ and $\alpha_{t+1}$ is equal to the time-wise variation of their characteristic vectors.
In order for this construction to be polynomial in size, it remains to show that it is possible to define the number of stages $T$ so that this is polynomially bounded and that the reduction from Vertex Cover Reconfiguration works out. Here, this is ensured, since the authors of [LM17] show the membership of Vertex Cover Reconfiguration in $\mathcal{NP}$. More precisely, they prove that for a yes-instance of Vertex Cover Reconfiguration, the number of reconfiguration steps needed in order to obtain $Q$ from $R$ is at most $\mathcal{O}(|V|^4)$. Therefore, defining the number $T$ to be this upper bound yields a polynomial construction.
It is not difficult to see that the instance of Vertex Cover Reconfiguration is a yes-instance if and only if the optimal value of (BND) is $|R| + |Q|$. Indeed, if $(G = (V, E), R, Q, k)$ is a yes instance, then let $\alpha_1$, $\alpha_2$,...,$\alpha_N$ be a corresponding reconfiguration sequence with $N \leq T$, where the latter may be assumed due to the definition of $T$. By setting $x_{t-1} := \chi_{\alpha_t}$ for all $t = 1, ..., N$ and $x_t := \chi_{\alpha_N}$ for $t \geq N$ one clearly yields a feasible solution for the instance of (BND) that reaches the upper bound $|R| + |Q|$ on the objective value, since $\alpha_1 = R$ and $\alpha_N = Q$.
Conversely, any optimal solution of (BND) with an objective value of $|R| + |Q|$ must satisfy that $x_0$ is the characteristic vector of $R$ and $x_T$ is the characteristic vector of $Q$, due to the definition of the objective function. The bound on the time-wise variation and the definition of $X$ then ensure that $x_0$, $x_1$, ..., $x_T$ is a sequence of characteristic vectors of vertex covers in $G$ of sizes at most $k$, where the symmetric difference of two consecutive vectors is at most one. Since the symmetric difference of a reconfiguration sequence in Vertex Cover Reconfiguration has to be exactly one, the removal of sequences of duplicates yields a certificate for the instance of Vertex Cover Reconfiguration, which is therefore a yes-instance. □

**Remark 5.2.1.** The hardness result in Theorem 5.2.2 does not imply that under the assumption $\mathcal{P} \neq \mathcal{NP}$ there cannot exist an oracle-polynomial algorithm for the problem (BND) with time-wise variation. This is due to the fact that although

the underlying problem as defined in the proof is tractable for any objective $c \in \{c_0, .., c_T\}$, it is not known to be polynomially solvable for an arbitrary objective $c$.

Actually, another result regarding a parameterized variant of Vertex Cover Reconfiguration has been proven in the literature. Here, the problem is parameterized in $\ell \in \mathbb{N}_{\geq 0}$ and the question is whether there exists a reconfiguration sequence that transforms $R$ into $Q$ with $N \leq \ell$, i.e., with at most $\ell$ reconfiguration steps. The authors of [MNR14] have proven this problem to be $W[1]$-hard.

**Lemma 5.2.2.** ([MNR14], Theorem 4.0.2 ) Vertex Cover Reconfiguration is $W[1]$-hard when parameterized in $\ell$.

Since Vertex Cover Reconfiguration parameterized in $\ell$ is fixed-parameter reducible to the problem (BND) parameterized in $T$, the previous result implies

**Theorem 5.2.3.** The problem (BND) with time-wise variation parameterized in $T$ is $W[1]$-hard, even if $S = \mathbb{1}_T$, $X := X_t$ for all $t$, and the underlying problem $\max_{x \in X} c_t^\top x$ is tractable for each stage.

*Proof.* Let $(G = (V, E), R, Q, k)$ be an instance of Vertex Cover Reconfiguration parameterized in $\ell$. The definition of the instance of (BND) is analogous to the proof of Theorem 5.2.2, the only exception is that here, it is $T := \ell - 1$. It is obvious that this is an $FPT$-time reduction and that the (parameterized) instance of Vertex Cover Reconfiguration is a yes-instance, if and only if the optimal value of (BND) is $|R|+|Q|$. Thus follows the $W[1]$-hardness of (BND) with time-wise variation. □

Observe that Theorem 5.2.3 claims the $W[1]$-hardness, but not the $\mathcal{NP}$-hardness of (BND) with time-wise variation parameterized in $T$. In fact, the first reduction from Vertex Cover Reconfiguration clearly does not work if $T$ is constant. Thus, the complexity of (BND) with time-wise variation and fixed $T$ is still unclear. For the reason given in Remark 5.2.1, the $W[1]$-hardness of (BND) parameterized in $T$ from Theorem 5.2.3 does not rule out the existence of an (oracle-)$FPT$ algorithm.

This concludes the complexity-theoretical study of (BND). Table 5.2 below briefly summarizes the findings of this section.

| assumption \variation | total | time-wise | switch-wise |
|---|---|---|---|
| $S \in \mathcal{O}(1)$ and $X := X_t$ for all $t$ | ? | strongly $\mathcal{NP}$-hard, even if all underlying problems tractable | strongly $\mathcal{NP}$-hard, even if $X$ simple no oracle-poly algo unless $\mathcal{P} \neq \mathcal{NP}$ |
| $T \in \mathcal{O}(1)$ and $X := X_t$ for all $t$ | ? | $W[1]$-hard | ? |

Table 5.2: Summary of results obtained in Section 5.2

Although Theorem 5.2.2 does not suffice to show it, it is a fact that there cannot exist an oracle-polynomial algorithm for (BND) with time-wise variation, even if both $S$ and $T$ are constant. The proof will be given in the next section. It will turn out that using only a polynomial number of calls to $(\mathrm{LO}_t)$-oracles, already the feasibility of problem (BND) cannot be decided regardless of the type of variation, and even if both $S$ and $T$ are constant. More importantly, this result will not rely on the assumption $\mathcal{P} \neq \mathcal{NP}$.

## 5.3 Information-Based Results for Constant $S$ and $T$

As the title suggests, this section pursues more of an information oriented approach to the problem (BND). It is still driven by the question whether there exist oracle-polynomial algorithms for the variants of problem (BND) under the assumption that each set $X_t$ is given via an $(\mathrm{LO}_t)$-oracle $\Upsilon_t$. Here, however, it is assumed that the bound on the variation $S$ and the number of stages $T$ is constant. While this question was only partially answered so far, this section will give the definitive answer *no*. Thus, based on $(\mathrm{LO}_t)$-oracles, the relative complexity of (BND) increases significantly compared to the complexity of the underlying problem. Indeed, it will turn out that not only is it impossible to determine an optimal solution for (BND) (regardless of the type of variation) using polynomially many $(\mathrm{LO}_t)$-oracle calls, but even its feasibility cannot be decided oracle-polynomially. Most importantly, due to the switch to an information related view represented by the next theorem, this result holds independent of the assumption $\mathcal{P} \neq \mathcal{NP}$.

**Theorem 5.3.1.** ([Buc20]) Assume that a set $X \subseteq \{0,1\}^n$ is accessible only via a linear optimization oracle. Then, using a polynomial number of oracle calls, it cannot be decided whether $X$ contains any vector with $\mathbb{1}_n^\top x = \frac{n}{2}$.

Note that the above theorem does not depend on $\mathcal{P} \neq \mathcal{NP}$ whatsoever. Now, the existence of an oracle-polynomial algorithm for the problem (BND) with time-wise variation or total can be shown to contradict Theorem 5.3.1, even if $T$ and $S$ are constant and the feasible set does not change.

**Theorem 5.3.2.** Consider the problem (BND) with either switch-wise, time-wise or total variation and assume that each feasible set $X_t$ is given via an ($\mathrm{LO}_t$)-oracle $\Upsilon_t$. Then, a polynomial number of calls to the ($\mathrm{LO}_t$)-oracles $\Upsilon_t$ does not suffice to solve (BND), even if $T = 1$, $S$ is bounded by $= \mathbb{1}_r$, and even if the feasible set does not change, i.e., $X := X_t$ for all $t$.

*Proof.* This proof is structured as follows: First, a particular instance of (BND) is defined and its optimal value is analyzed. Then, the existence of an oracle-polynomial algorithm for (BND) is proven to contradict Theorem 5.3.1.
Let $\bar{X}_1 \subseteq \{0,1\}^n$ be an arbitrary set given by an ($\mathrm{LO}_t$)-oracle $\bar{\Upsilon}_1$ and define the set

$$\bar{X}_0 := \left\{ x \in \{0,1\}^n \colon \mathbb{1}_n^\top x = \tfrac{n}{2} \right\} .$$

Note that the set $\bar{X}_0$ is simple as it corresponds to the feasible set of a SELECTION PROBLEM with equality constraint. For any linear objective $c \in \mathbb{R}^n$, this problem can be solved by sorting the components by decreasing objective coefficients and selecting the $\frac{n}{2}$ first with respect to this sorting. Recall that for $T = 1$, time-wise and total variation are equivalent. Thus, define an instance $\mathcal{I}_1$ of (BND) for these two types of variation with $T = 1$ and $S = 1 \in \mathbb{N}$ by

$$\begin{aligned} \max \quad & \begin{pmatrix} \mathbb{0}_n \\ -1 \end{pmatrix}^\top x_0 + \begin{pmatrix} \mathbb{0}_n \\ 1 \end{pmatrix}^\top x_1 \\ \text{s.t.} \quad & \sum_{i=1}^{n+1} |x_0^{(i)} - x_1^{(i)}| \leq 1 \\ & x_0, x_1 \in X , \end{aligned}$$

where the feasible set does not change and is given by

$$X := \left(\bar{X}_0 \times \{0\}\right) \cup \left(\bar{X}_1 \times \{1\}\right) \subseteq \{0,1\}^{n+1} .$$

Note that the last entry of $x \in X$ indicates the membership of $\bar{x} := (x_1, ..., x_n)^\top$ in $\bar{X}_0$ or $\bar{X}_1$, i.e.,

$$\begin{pmatrix} \bar{x} \\ 0 \end{pmatrix} \in X \;\Leftrightarrow\; \bar{x} \in \bar{X}_0 \qquad \text{and} \qquad \begin{pmatrix} \bar{x} \\ 1 \end{pmatrix} \in X \;\Leftrightarrow\; \bar{x} \in \bar{X}_1 .$$

For switch-wise variation, define an analogous instance $\mathcal{I}_2$, where the only difference is the definition $S := (\mathbb{0}_n^\top, 1)^\top \in \mathbb{N}_{\geq 0}^{n+1}$.
Both for the instance $\mathcal{I}_1$ and for the instance $\mathcal{I}_2$, the optimal value of (BND) equals 1 if and only if $\bar{X}_0 \cap \bar{X}_1 \neq \emptyset$. Indeed, if there exists some $\bar{x} \in \bar{X}_0 \cap \bar{X}_1$, then obviously, the solution

$$x_0 := \begin{pmatrix} \bar{x} \\ 0 \end{pmatrix}, \; x_1 := \begin{pmatrix} \bar{x} \\ 1 \end{pmatrix}$$

is feasible and its objective value equals the trivial upper bound 1. Thus, the solution is optimal.

Conversely, if the optimal value of $\mathcal{I}_1$ (or $\mathcal{I}_2$, respectively,) is one, then it must hold that $x_0^{(n+1)} = 0$ and $x_1^{(n+1)} = 1$ as otherwise, the objective value no greater than zero. Since the variation constraint is already tight due to the $n+1$-st switch, the remaining components must be constant, i.e., $\bar{x}_i := x_0^{(i)} = x_1^{(i)}$ for all $i = 1, ..., n$. By definition of $X$, this on the one hand implies $\bar{x} \in \bar{X}_0$ by $x_0^{(n+1)} = 0$ and, on the other hand, $\bar{x} \in \bar{X}_1$ by $x_1^{(n+1)} = 1$. Consequently, $\bar{x} \in \bar{X}_0 \cap \bar{X}_1$.

Suppose that the claim is not true, i.e., that there exists an oracle-polynomial algorithm $\mathcal{A}$ which solves (BND) given an oracle $\Upsilon$ for the feasible set $X$. Note that by assumption, there is only an (LO$_t$)-oracle $\bar{\Upsilon}_1$ for the set $\bar{X}_1$ available, but no oracle $\Upsilon$ for $X$. Thus, neither the instance $\mathcal{I}_1$ nor $\mathcal{I}_2$ can be solved by calling the algorithm $\mathcal{A}$ due to the lack of an oracle for $X$. Nevertheless, using the oracle $\bar{\Upsilon}_1$ and the fact that $\bar{X}_0$ is simple, it is still possible to efficiently optimize any given linear objective $c \in \mathbb{R}^{n+1}$ over the set $X$ with the following subroutine:

1. Call the (LO$_t$)-oracle $\bar{\Upsilon}_1$ for the objective $\bar{c} := (c_1, ...., c_n)$, and obtain a corresponding optimal solution $x^*$.
2. Solve the problem $\max_{x \in \bar{X}_0} \bar{c}^\top x$ in polynomial time (since $\bar{X}_0$ is simple), and obtain corresponding optimal solution $\hat{x}$.
3. Return the better of the two solutions $\binom{x^*}{1}$ and $\binom{\hat{x}}{0}$ regarding the objective $c$.

By substituting the call of the (LO$_t$)-oracle $\Upsilon$ with the above subroutine, the oracle-polynomial algorithm $\mathcal{A}$ based on $\Upsilon$ becomes an oracle-polynomial algorithm $\bar{\mathcal{A}}$ based on the (LO$_t$)-oracle $\bar{\Upsilon}_1$. Thus, the instance $\mathcal{I}_1$ (and $\mathcal{I}_2$, respectively,) can be solved by calling $\bar{\mathcal{A}}$, and its optimal value indicates whether $\bar{X}_0 \cap \bar{X}_1 \neq \emptyset$. In particular, this implies that using a polynomial number of $\bar{\Upsilon}_1$-oracle calls it can be decided whether there exists some $x \in \bar{X}_1$ with $\mathbb{1}_n^\top x = n/2$, which contradicts Theorem 5.3.1. □

Actually, the former construction and argument strategy can also be applied to show an analogous statement for the penalized problem.

**Corollary 5.3.1.** Consider the problem (P) and assume that each feasible set $X_t$ is given via an (LO$_t$)-oracle $\Upsilon_t$. Then, a polynomial number of calls to the (LO$_t$)-oracles $\Upsilon_t$ does not suffice to solve (P), even if $T = 1$, and even if the feasible set does not change, i.e., $X := X_t$ for all $t$.

*Proof.* The construction and argument is analogous to the previous proof, except that the definition of the bound on the respective variation is omitted and replaced by an appropriate definition of a penalty parameter $\lambda \in \mathbb{R}^r_{>0}$, depending on the type of variation.

For switch-wise variation, define $\lambda \in \mathbb{R}^n_{>0}$ via $\lambda_i := n+1$ for $i = 1, ..., n$ and $\lambda_{n+1} := \varepsilon \in (0,1)$. For time-wise variation (or equivalently, total variation), define $\lambda \in \mathbb{R}_{>0}$ by $\lambda := \varepsilon \in (1/2, 1)$.

Then, it is easy to see that the optimal value of the penalty problem is positive if and only if there exists some $\bar{x} \in \bar{X}_0 \cap \bar{X}_1$. The rest of the argument is analogous to the previous proof. □

While in the setting of the former theorem, a polynomial number of $(\mathrm{LO}_t)$-oracle calls does not suffice to obtain an optimal solution for the problem (BND), its feasibility can be decided oracle-polynomially. Indeed, if $X := X_t$ for all $t$, then deciding the feasibility of (BND) by only using an $(\mathrm{LO}_t)$-oracle $\Upsilon$ for the set $X$ actually reduces to deciding whether $X$ is nonempty, as any constant solution $x_t := x$ for all $t$ with $x \in X$ is feasible, regardless of the type of variation and its bound. In fact, one call of the oracle $\Upsilon$ with an arbitrary objective $c$ suffices to decide whether $X$ is nonempty, depending on whether the optimal value is finite or $-\infty$.
However, this immediately changes if the assumption $X := X_t$ for all $t$ is dropped. In this case, the feasibility of (BND) cannot be decided with a polynomial number of oracle calls, even if both $S$ and $T$ are constant.

**Theorem 5.3.3.** Consider the problem (BND) with either switch-wise, time-wise or total variation and assume that each feasible set $X_t$ is given via an $(\mathrm{LO}_t)$-oracle $\Upsilon_t$. Then, a polynomial number of calls to the $(\mathrm{LO}_t)$-oracles $\Upsilon_t$ does not suffice to decide whether a given instance of (BND) is feasible, even if $S = \mathbb{0}_r$ and $T = 1$.

*Proof.* Suppose that the claim is not true, i.e., that there exists an algorithm $\mathcal{A}$ which decides the feasibility of any given instance of (BND) with $T = 1$ and $S = \mathbb{0}_r$ by using only a polynomial number of $(\mathrm{LO}_t)$-oracle calls for the feasible sets $X_t$. Here, $S$ is defined as the $r$-dimensional zero vector whose dimension depends on the type of variation. In particular, note that $S = \mathbb{0}_r$ results in all types of variation being equivalent, more precisely, only constant solutions $x_t := x$ with $x \in X_t$ for all $t$ are feasible. Now, consider an instance $\mathcal{I}$ of (BND) with $T = 1$, $S = \mathbb{0}_r$, and the feasible sets

$$X_0 := \left\{ x \in \{0,1\}^n \colon \mathbb{1}_n^\top x = \tfrac{n}{2} \right\} \quad \text{and} \quad X_1$$

where $X_1 \subseteq \{0,1\}^n$ is given via a linear optimization oracle $\Upsilon_1$.

Note that $X_0$ is not given via an oracle so that the algorithm $\mathcal{A}$ *cannot* be called to decide the feasibility of $\mathcal{I}$. Nevertheless, since $X_0$ is simple, there exists an efficient optimization algorithm which solves the linear optimization problem over $X_0$ for any objective. By replacing any call of the oracle $\Upsilon_0$ for the set $X_0$ in $\mathcal{A}$ with an efficient optimization algorithm, $\mathcal{A}$ turns into an algorithm $\bar{\mathcal{A}}$ which is oracle-polynomial regarding the oracle $\Upsilon_1$ for $X_1$.

The resulting algorithm $\bar{\mathcal{A}}$ can thus be called to decide the feasibility of the instance $\mathcal{I}$ by using only a polynomial number of $\Upsilon_1$-calls.
Now since $S = \mathbb{0}_r$, any feasible solution $x_0, x_1$ of the instance $\mathcal{I}$ must satisfy $x_0 = x_1 \in X_0 \cap X_1$. Therefore, $\mathcal{I}$ is feasible if and only if there exists some $x \in X$ with $\mathbb{1}_n^\top x = n/2$. Thus, the claim follows with Theorem 5.3.1. □

Note that the idea of the previous proof does not depend on a specific type of variation but rather on the fact that it is possible to enforce that a solution of (BND) must be constant over time. Thus, Theorem 5.3.3 actually holds for any type of variation measure where this is possible. Further note that the feasibility of the penalty problem (P) can always be decided oracle-polynomially, as it reduces to deciding whether each feasible set $X_t$ is nonempty. The latter can obviously be decided by calling the respective ($\text{LO}_t$)-oracle $\Upsilon_t$ for each $t = 0, ..., T$ with an arbitrary objective.

With the former theorem, the main question driving this section has been answered: given an instance of (BND) where each set is given by an ($\text{LO}_t$)-oracle, there does not exist an algorithm solving (BND) using only a polynomial number of ($\text{LO}_t$)-oracle calls, regardless of the type of variation and even in the quite restricted setting that $S$ and $T$ are both constant. Table 5.3 below summarizes all current results regarding ($\text{LO}_t$)-oracle algorithms obtained in this chapter.

| assumption \variation | total /time-wise | switch-wise |
|---|---|---|
| $n \in \mathcal{O}(1)$ | ($\text{LO}_t$)-oracle-polynomial and even (($\text{LO}_t$)-oracle-)$FPT$ in $n$ | oracle-polynomial |
| $T, S \in \mathcal{O}(1)$ and $X := X_t$ for all $t$ | no polynomial time ($\text{LO}_t$)-oracle optimization algorithm | |
| $T, S \in \mathcal{O}(1)$ | no polynomial time ($\text{LO}_t$)-oracle algorithm to decide feasibility | |

Table 5.3: Results regarding oracle-polynomial solution of (BND) assuming ($\text{LO}_t$)-oracles for $X_t$

As a result of this section, it is clear that if the aim is to find an efficient oracle-based algorithm for (BND), then another approach is required. Indeed, the next subsection will not consider an ($\text{LO}_t$)-oracle, but a differnt, more powerful oracle and present a corresponding oracle-algorithm for (BND) with time-wise and total variation that runs in polynomial time if $S, T \in \mathcal{O}(1)$.

### 5.3.1 An Oracle-Algorithm for Constant $S$ and $T$

This subsection will devise an oracle-algorithm for (BND) with time-wise and total variation. In case $S$ and $T$ are constant, this algorithm will turn out to be efficient, i.e., require at most polynomially many oracle calls. The main difference to the preceding sections is that instead of assuming each set $X_t$ to be given via an ($\text{LO}_t$)-oracle $\Upsilon_t$,

it is now assumed that there exists a linear optimization oracle $\Psi$ for the so-called Optimal Constant Extension Problem (OCE). This problem can be described as follows:

Let there be a discrete time horizon $\{0, ..., T\}$ along with corresponding feasible sets $X_t \subseteq \{0,1\}^n$ for each $t$ and a fixed partial solution $\tau = (\tau_0, ..., \tau_T)$ in $\{0,1\}^{J \times (T+1)}$, where $J \subseteq \{1, ..., n\}$. Note that without loss of generality, one may assume $J = \{n - |J| + 1, ..., n\}$ by a permutation of the superscripts, if necessary. Then, the task of (OCE) is to find a constant solution $y = y_t$ for all $t$ in $\{0,1\}^{n-J}$ that forms a feasible solution $\binom{y}{\tau_t} \in X_t$ at any stage and maximizes the objective $C^\top y$, for some given $C \in \mathbb{R}^{n-J}$. More formally, the problem is defined by

**Problem 5.3.1.** Optimal Constant Extension Problem

**Input:** Feasible sets $X_t \subseteq \{0,1\}^n$, $t = 0, ..., T$, a partition $I \cup J = \{1, ..., n\}$, a partial solution $(\tau_0, ..., \tau_T) \in \{0,1\}^{J \times (T+1)}$ and an objective $C \in \mathbb{R}^I$

**Task:** Find $y \in \{0,1\}^I$ with $\binom{y}{\tau_t} \in X_t$ for all $t$ so that $y$ maximizes $C^\top y$

$$\text{(OCE)} \qquad \begin{cases} \max & \displaystyle\sum_{i \in I} C^{(i)} \cdot y^{(i)} & \\ \text{s.t.} & (y, \tau_t) \in X_t & t = 0, ..., T \\ & y^{(i)} \in \{0,1\} & i \in I\ . \end{cases}$$

The assumption that there exists a linear optimization oracle $\Psi$ for (OCE) is stronger than the previous assumption that there exists a linear optimization oracle $\Upsilon_t$ for ($\text{LO}_t$), since an oracle $\Psi$ is more powerful than an oracle $\Upsilon_t$. Indeed, by definition, a single call to $\Psi$ yields an optimal solution to (OCE), while a polynomial number of calls to the ($\text{LO}_t$)-oracles $\Upsilon_t$ does not suffice to even decide the feasibility of (OCE), even if $X := X_t$ for all $t$.

**Lemma 5.3.1.** Consider an instance of (OCE) and let each set $X_t$ be given via an ($\text{LO}_t$)-oracle $\Upsilon_t$. Using a polynomial number of calls to the oracles $\Upsilon_t$, it cannot be decided whether the instance of (OCE) is feasible, even if $T = 1$, $|I| = n - 1$, and even if the feasible set does not change, i.e., $X := X_0 = X_1$.

*Proof.* Define $\bar{n} := n - 1$ and construct an instance $\mathcal{I}$ of (OCE) with $I = \{1, ..., \bar{n}\}$, $J = \{n\}$, and $T = 1$, $C = \mathbb{0}_I$ as well as the partial solution $\tau_0^{(n)} := 0$ and $\tau_1^{(n)} := 1$, i.e.,

$$\begin{aligned} \max \quad & \mathbb{0}_{\bar{n}}^\top y \\ \text{s.t.} \quad & \binom{y}{0}, \binom{y}{1} \in X\ , \end{aligned}$$

where $X$ is defined as in the proof of Theorem 5.3.2, i.e.,

$$X := \big(\bar{X}_0 \times \{0\}\big) \cup \big(\bar{X}_1 \times \{1\}\big) \subseteq \{0,1\}^n \ .$$

Let $\bar{X}_0 := \{x \in \{0,1\}^{\bar{n}} \colon \mathbb{1}^\top x = {}^{\bar{n}}\!/_2\}$ and let $\bar{X}_1 \subseteq \{0,1\}^{\bar{n}}$ be given via an $(\mathrm{LO}_t)$-oracle $\bar{\Upsilon}_1$. This instance $\mathcal{I}$ is feasible if and only if $\bar{X}_0 \cap \bar{X}_1 \neq \emptyset$. Now, suppose that the claim is not true and let $\mathcal{A}$ be an oracle-polynomial algorithm solving $\mathcal{I}$ using an $(\mathrm{LO}_t)$-oracle $\Upsilon$ for the set $X$. With analogous reasoning as in the proof of Theorem 5.3.2, $\mathcal{A}$ can be transformed into an oracle-polynomial algorithm $\bar{\mathcal{A}}$ based on the $(\mathrm{LO}_t)$-oracle $\bar{\Upsilon}_1$. Since the feasibility of $\mathcal{I}$ is equivalent to the question whether there exists some $x \in \bar{X}_1$ with $\mathbb{1}_{\bar{n}}^\top x = {}^{\bar{n}}\!/_2$, the existence of $\mathcal{A}$ contradicts Theorem 5.3.1. □

Although the problem (OCE) does not contain a variation constraint, it is still related to the problem (BND). In fact, it can be derived by studying the problem (BND) with time-wise or total variation as described in the following:

Let $x$ be an optimal solution for (BND) where the bound on the variation $S \in \mathbb{N}^r_{\geq 0}$ is fixed. Then, a fixed bound $S$ on the switch-wise variation will generally not lead to many switches being constant over the entire time horizon, however, for time-wise or total variation this is eventually the case.

Thus, the set of switches $\{1,...,n\}$ can be partitioned into two disjoint subsets $I$ and $J$, where the former set contains only switches that remain constant over the entire time horizon, i.e.,

$$i \in I \quad \Leftrightarrow \quad x_t^{(i)} = x_{t+1}^{(i)} \quad \text{for all } t = 0,...,T-1 \ ,$$

while all other switches are contained in the other set $J$. Let $\sigma := |J|$ denote the cardinality of $J$. The states of all $n - |J| = n - \sigma$ switches in $I$ can thus be represented by a corresponding vector $y \in \{0,1\}^{n-\sigma}$ with $y^{(i)} := x_t^{(i)}$ for all $t$. Let the restriction $\tau \in \{0,1\}^{|J| \times (T+1)}$ of the solution $x$ to the subset $J$ give the states of the switches in $J$, i.e.,

$$\tau_t^{(i)} := x_t^{(i)} \qquad \text{for all } t = 0,...,T \quad \text{and } i \in J \ .$$

Using this notation, $x$ can be written as $(y,\tau)$. Since $x_t$ has to be an element of $X_t$ for each stage $t$, this translates to $(y,\tau_t) \in X_t$ for all $t$. Observe that the constant switches in $I$ do not contribute to the variation of the solution at all. Consequently, the partial solution $\tau$ must satisfy the variation constraint $\mathrm{Var}(\tau_0,...,\tau_T) \leq S$.

Thus, the optimality of the solution $x = (y,\tau)$ implies that the vector $y \in \{0,1\}^{n-\sigma}$, which corresponds to the constant switches, must be an optimal solution of the following instance of (OCE) with the fixed partial solution $\tau$

$$\text{(P)} \qquad \begin{cases} \mathrm{val}_{J,\tau} := \max & \displaystyle\sum_{i \notin J} C^{(i)} \cdot \mathbf{y^{(i)}} \\ \text{s.t.} & (\mathbf{y},\tau_t) \in X_t \qquad t = 0,...,T \\ & \mathbf{y} \in \{0,1\}^{n-|J|} \ . \end{cases}$$

where $C^{(i)} := \sum_{t=0}^{T} c_t^{(i)}$ for all $i \notin J$, i.e., for all $i \in I$. Keep in mind that only $y$ is a variable here (indicated by the use of bold font) and not $\tau$, which is fixed.

The idea of the oracle-algorithm is to enumerate all possible sets $J$ and all possibilities for $\tau$ that do not exceed the bound on the variation, to compute the corresponding optimal value $\text{val}_{J,\tau}$ with a call to the oracle $\Psi$, and then return the maximal objective value of a feasible solution computed throughout this enumeration. The basic outline of the algorithm can be found as pseudocode in Algorithm 3.

**Algorithm 3:** oracle-algorithm for (BND)

**Data:** instance of (BND) with variation $\text{Var} \in \{\text{Var}_{tw}, \text{Var}_{tv}\}$ and bound $S \in \mathbb{N}_{\geq 0}^{r}$, a linear optimization oracle $\Psi$ for (OCE)
**Result:** optimal solution $x$ and optimal value $OPT$

```
OPT := ∞;
if Var = Var_tw then
    define σ := max_{t=0,...,T} S_t · T
if Var = Var_tv then
    define σ := S
for each J ⊆ {1, ..., n} with |J| = σ do
    for each τ ∈ {0,1}^{|J|×(T+1)} do
        if Var(τ_0, ..., τ_T) ≤ S then
            compute val_{J,τ} by calling Ψ for (P)
            Val_{J,τ} := Σ_{i∈J} Σ_{t=0}^{T} c_t^{(i)} · τ_t^{(i)} + val_{J,τ}
        if Val_{J,τ} ≥ OPT then
            OPT := Val_{J,τ}
            x_t := (y, τ_t) for all t = 0, ..., T
return optimal value OPT and optimal solution x
```

Next, Algorithm 3 is proven to be an oracle-algorithm for (BND) with time-wise or total variation regarding a linear optimization oracle $\Psi$ for (OCE).

**Lemma 5.3.2.** Given a linear optimization oracle $\Psi$ for (OCE), Algorithm 3 computes an optimal solution for (BND) with time-wise or total variation in time $\mathcal{O}(2^{\sigma \cdot (T+1)} \cdot n^{\sigma})$, where $\sigma := \max_{t=0,\dots,T} S_t \cdot T$ for time-wise variation and $\sigma := S$ for total variation.
In particular, if both $S$ and $T$ are constant, (BND) with time-wise or total variation can be solved oracle-polynomially with respect to the oracle $\Psi$. Moreover, the running time shows that (BND) is in $XP$ for fixed $S$ and $T$.

*Proof.* Depending on whether the variation is measured time-wise or totally, the maximum possible number of switches which are *not* constant over all stages is given by $\sigma := \max_{t=0,...,T} S_t \cdot T$ for time-wise variation, whereas it is given by $\sigma := S$ for total variation (lines 3 and 5).

Now, any subset $J \subseteq \{1, ..., n\}$ consisting of $\sigma$ switches (line 6) and all possible states $\tau \in \{0,1\}^{\sigma \times (T+1)}$ of those switches are enumerated (lines 6 and 7). In order to check whether the currently considered $\tau$ satisfies bound on the time-wise variation (or total variation, respectively) it suffices to check whether the constraints

$$\mathrm{Var}_{tw}(\tau_0, ..., \tau_T) \leq S \quad \Leftrightarrow \quad \sum_{i \in I} |\tau_t^{(i)} - \tau_{t-1}^{(i)}| \leq S_t \qquad \text{for all } t = 1, ..., T$$

are satisfied for time-wise variation, or whether the constraints

$$\mathrm{Var}_{tv}(\tau_0, ..., \tau_T) \leq S \quad \Leftrightarrow \quad \sum_{i \in I} \sum_{t=1,...,T} |\tau_t^{(i)} - \tau_{t-1}^{(i)}| \leq S$$

are satisfied for total variation (line 8).

If the currently considered partial solution $\tau$ does not exceed the respective variation bound, the corresponding problem (P) is solved via a call to the oracle $\Psi$. Let $y^*$ be an optimal solution of (P) (line 9) returned by $\Psi$, then by construction the solution $(y^*, \tau)$ does not exceed the bound on the respective variation and the fact that $(y^*, \tau_t)$ is an element of $X_t$ for each stage $t$ follows immediately from the definition of (P). The objective value of the feasible solution $(y^*, \tau)$ is obtained by summing up the optimum value $\mathrm{val}_{I,\tau}$ and the value that $\tau$ contributes (line 10). Thus, the largest value that is achieved throughout the course of the entire enumeration yields the optimum value as well as an optimal solution to (BND).

The overall running time of the algorithm is given by $\mathcal{O}(2^{\sigma \cdot (T+1)} \cdot n^\sigma)$, since for each subset $J$ of cardinality $\sigma$, there are $2^{\sigma(T+1)}$ possible states of $\tau$, and the number of possible $\sigma$-cardinality subsets $J$ is bounded by $\mathcal{O}(n^\sigma)$.

The second claim follows immediately: If $S, T \in \mathcal{O}(1)$, then also $\sigma$ is constant. The total number of times Algorithm 3 calls the oracle $\Psi$ is given by $\mathcal{O}(2^{\sigma \cdot (T+1)} \cdot n^\sigma)$ which is hence polynomial. □

In the next and final subsection, the complexity of (OCE) is studied. Although the problem is shown to be $\mathcal{NP}$-hard in general, there is quite a number of tractable cases depending on the underlying feasible set. Due to the former Lemma 5.3.2, the tractability of (OCE) will imply the tractability of the corresponding problem (BND) for fixed $S$ and $T$, as the oracle $\Psi$ in Algorithm 3 can be replaced with an efficient optimization algorithm.

### 5.3.2 Some Tractable Examples of (OCE) and (BND)

This subsection is dedicated to the complexity of (OCE). Leaving the former oracle-setting, this section considers examples of feasible sets $X_t$ and settles the complexity

of the corresponding problem (OCE). The tractability of (OCE) in special cases combined with Lemma 5.3.2 will then yield the tractability of the corresponding problem (BND) with time-wise or total variation under the constraint $S, T \in \mathcal{O}(1)$. It is not difficult to see that (OCE) is generally $\mathcal{NP}$-hard, even when the underlying problem is simple, by a reduction from Matroid Intersection:

**Lemma 5.3.3.** The problem (OCE) is $\mathcal{NP}$-hard, even if $T = 2$ and for $t = 0, 1, 2$, each feasible set $X_t$ corresponds to the set of incidence vectors of independent sets of a matroid $\mathcal{M}_t = (E, \mathcal{E}_t)$, all of which are defined over the same finite ground set $E$.

*Proof.* Recall from the preliminaries that a matroid is a simple set as shown, for example, by the greedy algorithm, whereas (weighted) Matroid Intersection is $\mathcal{NP}$-hard. Let $\mathcal{M}_0 = (E, \mathcal{F}_0)$, $\mathcal{M}_1 = (E, \mathcal{F}_1)$ and $\mathcal{M}_2 = (E, \mathcal{F}_2)$ be the three matroids defined in the reduction showing the $\mathcal{NP}$-hardness of Matroid Intersection outlined in the preliminaries.

Then, define an instance of (OCE) by $n := |E|$ and $I := n$, $J := \emptyset$. Note that by this definition, there is no partial solution $\tau$. Further define the objective via $C := \mathbb{1}_n$, $T := 2$ and define each feasible set $X_t$, $t = 0, 1, 2$, as the set of incidence vectors corresponding to the independent sets in $\mathcal{F}_t$. Thus, any feasible solution of (OCE) corresponds to an independent set $F \in \mathcal{F}_0 \cap \mathcal{F}_1 \cap \mathcal{F}_2$ and the optimal value corresponds to the cardinality of a largest independent set contained in this intersection.

All three matroids have a common base if and only if the optimal value of (OCE) is given by $n - 1$. Thus follows the claim. □

Recall from the preliminaries that there exists a polynomial time greedy algorithm which computes an optimal solution for the maximization of any linear objective over the independent sets of a matroid. Thus, the previous proof implies that (OCE) is $\mathcal{NP}$-hard, even if $J = \emptyset$ and each feasible set $X_t$ is simple. The complexity of the problem is thus neither due to the complexity of the feasible set nor caused by the fixed partial solution, but instead arises from the task of finding an element in the intersection of the feasible sets.

Nevertheless, there are quite a few cases where (OCE) is polynomially, or at least pseudo-polynomially solvable. Some examples will be described in the following.

**Lemma 5.3.4.** Let $I \cup J$ be a partition of $\{1, ..., n\}$ and let there be $a_t \in \mathbb{N}_{\geq 0}^n$, $b_t \in \mathbb{N}_{\geq 0}$ for $t = 0, ..., T$. Then, the following the problem

$$\text{(OCE-KP)} \quad \begin{cases} \max & \displaystyle\sum_{i \in I} C^{(i)} \cdot y^{(i)} \\ \text{s.t.} & (y, \tau_t) \in \{x \in \{0,1\}^n \colon \displaystyle\sum_{i=1}^{n} a_t^{(i)} x^{(i)} \leq b_t\} \qquad t = 0, ..., T \\ & y^{(i)} \in \{0,1\} \qquad i \in I \,, \end{cases}$$

with objective vector $C \in \mathbb{R}^I$ and fixed partial solution $\tau \in \{0,1\}^{J \times (T+1)}$ polynomially reduces to a Binary Knapsack Problem on the partition set $I$. This statement also holds true for the version of the Binary Knapsack Problem with an equality constraint $\sum_{i=1}^{n} a_t^{(i)} x^{(i)} = b_t$ for each $t$.

*Proof.* Let an instance of (OCE-KP) be given. Without loss of generality, one may assume $\sum_{j \in J} a_t^{(j)} \tau_t^{(j)} \leq b_t$ for all $t = 0, ..., T$ as otherwise, the instance is clearly infeasible. Since an item $i \in I$ must either be packed in every stage or in none of them, its weight may be defined by $\bar{a}^{(i)} := \sum_{t=0}^{T} a_t^{(i)}$. Since the constant solution $y$ combined with the fixed solution $\tau$ may not exceed the corresponding knapsack capacity at each stage, the total weight of the items in $I$ that are packed is bounded from above by $\bar{b} := \min_{t=0,...,T} \left( b_t - \sum_{j \in J} a_t^{(j)} \tau_t^{(j)} \right)$. Thus, (OCE-KP) clearly reduces to a binary knapsack problem on the partition set $I$ with weights $\bar{a}$, knapsack capacity $\bar{b}$, and profits $C$. It is easy to see that this reduction works for both an inequality and an equality constraint on the weights $a$. □

Recall that there exists a pseudo-polynomial dynamic programming scheme for the Binary Knapsack Problem, and that the Selection Problem, as a special case where all weights are equal to one, can be solved in polynomial time. Consequently, any call of the oracle $\Psi$ for the problem (OCE-KP) in Algorithm 3 can be replaced by a corresponding (pseudo-)polynomial optimization algorithm.

**Corollary 5.3.2.** If $S, T \in \mathcal{O}(1)$, then (BND) with time-wise or total variation and feasible sets $X_t$, $t = 0, ..., T$, given by a knapsack constraint as defined in Lemma 5.3.4 can be solved in pseudo-polynomial time. In particular, the problem can be solved in polynomial time for a selection constraint, i.e., if $a_t = \mathbb{1}_n$ for all $t = 0, ..., T$.

Recall that the second part of the above corollary is actually not a new result: the tractability of (BND) with a selection constraint and bounded time-wise variation was already established in Theorem 4.2.2 where both $S$ and $T$ are actually part of the input. Similarly, for the corresponding problem with total variation, stronger

statements about the tractability were already given in the previous chapter (see Lemmas 4.2.6 and 4.2.7). There, it sufficed that either $S$ or $T$ is fixed, but not necessarily both.

In the following, the problem (OCE) will be proven to be polynomially reducible to the underlying problem for two classes of optimization problems,namely Covering problems and Packing problems. In fact, the structure of a Covering or Packing problem in combination with the fixed partial solution allows a reduction to a smaller problem of the same type.

**Lemma 5.3.5.** Consider (OCE) where each feasible set is given by the set $X$ containing all characteristic vectors of feasible solutions to a Packing Problem. Then, (OCE) polynomially reduces to solving a smaller Packing Problem instance.

*Proof.* Let an instance of (OCE) as defined in the lemma be given, i.e., the task of (OCE) is to find a subset $V^I \subseteq \{v_i \colon i \in I\}$ so that at each stage $t$, the union $V^I \cup V_t^J$ satisfies the packing constraint $|(V^I \cup V_t^J) \cap S_k| \leq 1$ for all $k \in \{1, ..., m\}$. Here, the latter set $V_t^J \subseteq \{v_i \colon i \in J\}$ consists of elements that are *enforced* ($\tau_t^{(i)} = 1$) by the fixation $\tau_t$ at stage $t \in \{0, ..., T\}$. The remaining elements $v_i$ with $i \in J$ are *forbidden* ($\tau_t^{(i)} = 0$) by the fixation. Without loss of generality, one may assume that there exists no stage $t$ so that some set $S_k$, $k \in \{1, ..., m\}$, contains two enforced elements from $V_t^J$. Indeed, in this case the instance is clearly infeasible.

If there is an element $v_i$ with $i \in I$ such that some set $S_k$, $k \in \{1, ..., m\}$, contains $v_i$ as well as an element from $V_t^J$ that is enforced at some stage $t$, then this element $v_i$ may deleted from $\{v_i \colon i \in I\}$. This is due to the fact that the inclusion of any such element $v_i$ in the set $V^I$ would violate the corresponding packing constraint for $S_k$ at stage $t$. Let $W \subseteq \{v_i \colon i \in I\}$ collect all elements which remain after this deletion. At last, delete all sets $S_k$, $k \in \{1, ..., m\}$, which contain an enforced element and, in the remaining sets, remove all elements which either correspond to a deleted element in $\{v_i \colon i \in I\} \setminus W$ or a forbidden element.

The task of (OCE) then consists in finding a (maximum weight) subset $V^I \subseteq W$ of the non-deleted elements in $W$ so that the packing constraint $|S_k \cap V^I| \leq 1$ is satisfied for all remaining sets $S_k$. Obviously, this is a Packing Problem again. □

A similar result can be proven for Covering Problems.

**Lemma 5.3.6.** Consider (OCE) where each feasible set is given by the set $X$ containing all characteristic vectors of feasible solutions to a Covering Problem. Then, (OCE) polynomially reduces to solving a smaller instance of the same problem type.

*Proof.* The argument is symmetric to the previous proof and therefore only briefly outlined. Here, the task of (OCE) is to find a subset $V^I \subseteq \{v_i \colon i \in I\}$ so that at each

stage $t$, the union $V^I \cup V_t^J$ satisfies the covering constraint $|(V^I \cup V_t^J) \cap S_k| \geq 1$ for all $k \in \{1, ...m\}$. W.l.o.g., there exists no stage $t$ so that the corresponding fixation $\tau_t$ forbids all elements in some set $S_k$, $k \in \{1, ..., m\}$, as in this case, the instance is clearly infeasible.

Any element $v_i$ with $i \in I$ such that some set $S_k$, $k \in \{1, ..., m\}$, contains $v_i$ and otherwise only forbidden elements from $V_t^J$ for some stage $t$ may deleted from $\{v_i : i \in I\}$. The inclusion of any such $v_i$ in the set $V^I$ is necessary to satisfy the covering constraint for $S_k$ at stage $t$. Let $W \subseteq \{v_i : i \in I\}$ collect all elements which remain after this deletion and delete all sets $S_k$ which contain an element in $W$. In the remaining sets, remove all elements which either correspond to a deleted element or a forbidden element. The task of (OCE) thus reduces to a (minimum weight) Packing Problem on the ground set $W$ with the remaining sets $S_k$. □

Many combinatorial problems belong to the previous two classes of problems. For example, (weighted) Vertex Cover, (weighted) Stable Set or (weighted) Maximum Matching. In particular, some packing or covering problems that are generally $\mathcal{NP}$-hard (e.g., Vertex Cover or Stable Set) are known to become tractable in a number special cases, e.g., bipartite graphs, graphs with bounded treewidth, trees, cycles or perfect graphs. This thesis will not look for any further combinatorial structures for which the tractability of (OCE) can be shown. This, however, does not mean that the possibility for a reduction of (OCE) to a smaller problem instance of the same type is limited to the previously discussed problems only. For example, by appropriately adjusting the node demands and supplies, an instance of (OCE) where the feasible set corresponds a (binary) minimum-cost flow problem appears to be reducible to a smaller min cost flow instance in a similar way. The tractability of (OCE) for particular combinatorial structures then again yields the tractability of the corresponding problem (BND) with time-wise or total variation if $S, T \in \mathcal{O}(1)$.

Note that the Lemmas 5.3.4, 5.3.6, and 5.3.5 do not contradict Proposition 5.3.1. Even though the problem (OCE) was reduced to the solution of a single instance of the underlying problem, the reductions were tailored to the specific type of the feasible set in each case. This does not imply that the feasibility of (OCE) can be decided given an ($\mathrm{LO}_t$)-oracle $\Upsilon_t$ for each feasible set $X_t$, since such a strategy must be universal and cannot depend of the feasible sets.

This concludes the chapter. Its most important result is that the question whether there exists an oracle-polynomial algorithm for the problem (BND) based on ($\mathrm{LO}_t$)-oracles for the sets $\Upsilon_t$ strongly depends on what part of the input is considered a constant. As was shown in Section 5.1, if the number of switches $n$ is considered a constant, then a polynomial number of ($\mathrm{LO}_t$)-oracle calls suffices to solve the problem (BND) regardless of the type of variation. In this case, the relative complexity of (BND) therefore does not increase compared to the

complexity of the underlying problems ($\mathrm{LO}_t$). However, this changes immediately if either the bound on the variation $S$, or the number of stages $T$ is considered a constant. The complexity-theoretical approach taken in Section 5.2 proved that at least for the problem (BND) with switch-wise variation, a polynomial number of ($\mathrm{LO}_t$)-oracle calls does not suffice to find an optimal solution if $S$ is constant, unless $P = \mathcal{NP}$. The remaining results in this section concerning the complexity of (BND) for time-wise variation did, however, not allow to draw an analogous conclusion for this variation type. Nevertheless, the answer to the main question of this chapter was finally given in Section 5.3. In fact, a polynomial number of ($\mathrm{LO}_t$)-oracle calls does not suffice to solve the problem (BND), more precisely, it does not even suffice to decide its feasibility. This holds true for any type of variation and even if both $S$ and $T$ are constant. Most importantly, it holds independent of the complexity-theoretical assumption $\mathcal{P} \neq \mathcal{NP}$, unlike the preceding result from Section 5.1. As was shown just before, assuming the existence of a different, more powerful linear optimization oracle $\Psi$ for the Optimal Constant Extension Problem (OCE) yields an oracle-algorithm for (BND) with time-wise or total variation. For constant $S$ and $T$, this algorithm is even oracle-polynomial with respect to the oracle $\Psi$. Although (OCE) is $\mathcal{NP}$-hard in general, there exist a number of cases for which it actually turns out to be tractable. An interesting future research question might be to study problem (OCE) further so as to better understand its possibilities and limitations in terms of tractability.

# Chapter 6

# Conclusion and Outlook

This thesis investigated a multi-stage optimization problem where a linear function was optimized over a sequence of solutions of an underlying problem that changes over time, where additionally, the variation of the sequence of the solutions was restricted. More precisely, it was either penalized in the objective function or, alternatively, contained in the set of constraints where a hard upper bound on it was imposed. In the latter case, the objective value was not directly affected. The bounded problem version turned out to be more challenging and hence, the focus of this thesis lay on this version for the three different types of variation called *time-wise*, *switch-wise* and *total variation*.

A detailed study of the special case in which the underlying problem has dimension one, and hence, no combinatorial structure, was presented in Chapter 3. In this case, all considered problem versions turned out to be tractable. That chapter actually went way beyond only the question of complexity, in fact, it was mostly focused on the investigation of the corresponding polyhedral structure. For all problem versions, a compact extended formulation was given that was proven to be integral when continuously relaxed. Furthermore, a perfect formulation in the original space was presented for each problem version. With time-wise variation being the only exception, all these perfect formulations required an exponential number of linear inequalities. However, the respective separation problem was proven to be solvable in polynomial time. Based on the idea to merge consecutive entries in a particular way, the chapter also devised an efficient combinatorial optimization algorithm for the bounded problem with total or switch-wise variation, which turned out to significantly outperform a previous naive dynamic programming scheme. Since the study conducted in Chapter 3 was very extensive, there is no obvious starting point for future research in this particular case. However, something that might leave some room for improvement could be the length of the correctness-proof for the merging algorithm. While the idea to use LP-duality arguments is an interesting change in strategy, and the proof is two-fold in the sense that it additionally implies the cor-

rectness of the proposed perfect formulation, it is very sophisticated. Future research might try to provide a more compact proof of correctness since the correctness of the perfect formulation can instead also be proven by application of Fourier-Motzkin elimination to the extended model.

The main discovery made in the subsequent Chapter 4 was that the complexity of the bounded problem strongly depends on the type of variation that is considered. The chapter was entirely restricted to one of the simplest underlying problems to consider, the Selection Problem. It was established that the penalized problem is tractable (regardless of the type of variation) by means of linear programming, and that the bounded problem with time-wise variation is tractable by means of a polynomial reduction to a min cost flow problem. In contrast, the corresponding multi-stage problem with a bound on the switch-wise variation already turned out to be strongly $\mathcal{NP}$-hard, even in a quite restricted setting. However, some tractable special cases for the latter problem were presented, such as the cases in which either the number of switches or the number of stages is fixed, or alternatively, the selection bound in all underlying problems is uniformly one. In future research, one could try to look for further tractable special cases of this problem version in order to better understand where the boundary between tractability and intractability runs. Another possible future research topic might consist of approximation algorithms, which were not in the scope of this thesis. The obvious open question of this chapter is the one about the general complexity of the bounded problem with total variation. Although this problem version was proven to be tractable for either a fixed number of stages or a fixed number of switches, its complexity remained unknown if none of the problem parameters is assumed to be fixed. Another possible generalization of the setting considered in Chapter 4 would be to consider other fairly simple underlying problems and see if the tractability of the time-wise case holds up or, alternatively, whether the complexity of the corresponding bounded problem with total variation can be determined.

The final chapter went somewhat in the direction in which the former question aims, but remained on a much more abstract level, pursuing an oracle-based approach. It conducted an information-theoretical study of the bounded problem version and obtained as its main result that, relative to the underlying combinatorial problem, the bounded problem is much more complex regardless of the variation. Assuming that the number of switches is a fixed parameter, the bounded problem was shown to oracle-polynomially solvable.. However, if instead the number of stages and the bound on the variation are fixed, none of the variants could be solved oracle-polynomially and, in fact, a polynomial number of oracle calls to the underlying problem did not even suffice to decide the feasibility of the problem. However, changing the latter assumptions to the degree that there is a (more powerful) linear optimization oracle for the closely related Optimal Constant Extension Problem, allowed to devise an oracle-algorithm which then runs in polynomial time. More-

over, the Optimal Constant Extension Problem was proven to be tractable for some classical combinatorial problems. Since the discussed problems are either always tractable or otherwise known to be tractable for quite some special cases, the corresponding oracle-algorithm can be turned into a polynomial algorithm in these cases. Future research might try to show the tractability of Optimal Constant Extension for further combinatorial structures or, regarding the oracle-approach, it might consider an oracle for different related problems in the hope of finding another oracle-approach in this new setting, too.

Finally, on a larger scale, future research might extend the multi-stage optimization problem with penalized or bounded variation towards the direction of robust or stochastic optimization to better model the fact that, usually, parameters of future instances are subject to uncertainty.

# Bibliography


[AH74] Alfred V. Aho and John E. Hopcroft. *The Design and Analysis of Computer Algorithms.* Addison-Wesley Longman Publishing Co., Inc., USA, 1st edition, 1974. (cited on page 6.)

[AMO93] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows: theory, algorithms, and applications.* Prentice-Hall, Inc., USA, 1993. (cited on pages 20 and 21.)

[BBDSR24] Enrico Bettiol, Christoph Buchheim, Marianna De Santis, and Francesco Rinaldi. An oracle-based framework for robust combinatorial optimization. *Journal of Global Optimization*, 88(1):27–51, 2024. (cited on page 22.)

[BDMW13] Kateřina Böhmová, Yann Disser, Matúš Mihalák, and Peter Widmayer. Interval selection with machine-dependent intervals. In *Algorithms and Data Structures*, pages 170–181, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. (cited on pages 108 and 109.)

[BELP18] Evridipis Bampis, Bruno Escoffiera, Michael Lampis, and Vangelis Th. Paschos. Multistage matchings. In *16th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2018)*, pages 7:1–7:13, 2018. (cited on page 16.)

[BET22] Evripidis Bampis, Bruno Escoffier, and Alexandre Teiller. Multistage knapsack. *Journal of Computer and System Sciences*, 126:106–118, 2022. (cited on pages 16 and 113.)

[BFR18] Pascale Bendotti, Pierre Fouilhoux, and Cecile Rottner. The min-up/min-down unit commitment polytope. *Journal of Combinatorial Optimization*, 36(3):1024–1058, 2018. (cited on page 8.)

[BGM22a] Christoph Buchheim, Alexandra Grütering, and Christian Meyer. Parabolic optimal control problems with combinatorial switching constraints – Part I: Convex relaxations. Technical Report 2203.07121 [math.OC], arXiv, 2022. (cited on page 9.)

[BGM22b] Christoph Buchheim, Alexandra Grütering, and Christian Meyer. Parabolic optimal control problems with combinatorial switching constraints – Part II: Outer approximation algorithm. Technical Report 2204.07008 [math.OC], arXiv, 2022. (cited on page 10.)

[BH22] Christoph Buchheim and Maja Hügging. Bounded variation in binary sequences. In *Proceedings of the 7th International Symposium on Combinatorial Optimization (ISCO 2022)*, pages 64–75. Springer, 2022. (cited on pages 10 and 11.)

[BH23] Christoph Buchheim and Maja Hügging. The polytope of binary sequences with bounded variation. *Discrete Optimization*, 48:100776, 2023. (cited on pages 10 and 11.)

[BH25] Christoph Buchhein and Maja Hügging. Multi-stage selection under bounded variation. Optimization Online, 3 2025. (cited on page 17.)

[Boy96] E. Andrew Boyd. On the complexity of a cutting plane algorithm for solving combinatorial linear programs. *SIAM Journal on Discrete Mathematics*, 9(3):365–376, 1996. (cited on page 22.)

[BT97] Dimitris Bertsimas and John Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1st edition, 1997. (cited on page 13.)

[Buc20] Christoph Buchheim. A note on the nonexistence of oracle-polynomial algorithms for robust combinatorial optimization. *Discrete Applied Mathematics*, 285:591–593, 2020. (cited on pages 23 and 150.)

[CCZ10] Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. Extended formulations in combinatorial optimization. *4OR*, 8(1):1–48, 2010. (cited on page 11.)

[CFK+15] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshantov, Daniel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer Cham, 1 edition, 2015. (cited on page 25.)

[DF13] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Springer Publishing Company, Incorporated, 2013. (cited on pages 25 and 26.)

[DKKRA16] Pelin Damci-Kurt, Simge Küçükyavuz, Deepak Rajan, and Alper Atamturk. A polyhedral study of production ramping. *Mathematical Programming*, 158:175–205, 07 2016. (cited on page 9.)

[Edm65a] Jack Edmonds. Maximum matching and a polyhedron with (0, 1)-vertices. *Journal of Research of the National Bureau of Standards B*, 69:125–130, 1965. (cited on pages 20 and 119.)

[Edm65b] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965. (cited on page 20.)

[Edm71] Jack Edmonds. Matroids and the greedy algorithm. *Mathematical Programming*, 1(1):127–136, 1971. (cited on page 18.)

[FNRZ22] Till Fluschnik, Rolf Niedermeier, Valentin Rohm, and Philipp Zschoche. Multistage vertex cover. *Theory of Computing Systems*, 66(2):454–483, 2022. (cited on page 16.)

[Fou26] Joseph Fourier. Solution d'une question particulière du calcul des in égalités. *Nouvean Bulletin des Sciences par la Société Philomatique de Paris*, 1826. (cited on page 13.)

[GH62] Alain Ghouila-Houri. Caractérisation des graphes non orientés dont on peut orienter les arĕtes de manière à obtenir le graphe d'une relation d'ordre. *C. R. Acad. Sci. Paris*, 254:1370–1371, 1962. (cited on page 12.)

[GJ90] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., USA, 1990. (cited on page 6.)

[GLS93] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer Berlin, Heidelberg, second corrected edition, 1993. (cited on pages 15, 22, and 24.)

[GTW14] Anupam Gupta, Kunal Talwar, and Udi Wieder. Changing bases: Multistage optimization for matroids and matchings. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming*, pages 563–575. Springer Berlin Heidelberg, 2014. (cited on page 16.)

[Hal22] Jonas P. Haldimann. Vergleich zweier Algorithmen zur Optimierung binärer Sequenzen mit beschränkter Variation. Bachelor thesis at TU Dortmund University, 2022. (cited on page 76.)

[HK56] Alan J. Hoffman and Joseph B. Kruskal. Integral boundary points of convex polyhedra. In *Linear inequalities and related systems*, Ann. of Math. Stud., no. 38, pages 223–246. Princeton Univ. Press, Princeton, NJ, 1956. (cited on page 11.)

[KV02] Bernhard Korte and Jens Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer-Verlag Berlin Heidelberg New York, New York, NY, 2 edition, 2002. (cited on pages 11, 18, and 20.)

[Law75] Eugene L. Lawler. Matroid intersection algorithms. *Math. Program.*, 9(1):31–56, 1975. (cited on page 19.)

[LLM04] Jon Lee, Janny Leung, and Francois Margot. Min-up/min-down polytopes. *Discrete Optimization*, 1(1):77–85, 2004. (cited on page 8.)

[LM17] Daniel Lokshtanov and Amer E. Mouawad. The complexity of independent set reconfiguration on bipartite graphs. *CoRR*, abs/1707.02638, 2017. (cited on pages 16, 147, and 148.)

[LPT22] Stefan Lendl, Britta Peis, and Veerle Timmermans. Matroid bases with cardinality constraints on the intersection. *Mathematical Programming*, 194:661–684, 2022. (cited on page 16.)

[LSW15] Yin Tat Lee, Aaron Sidford, and Sam Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *FOCS '15: Proceedings of the 2015 IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1049–1065, USA, 2015. IEEE Computer Society. (cited on page 22.)

[Mai25] Malte Maistrell. Multistage Knapsack Problem subject to Bounded Variation. Master thesis at TU Dortmund University, 2025. (cited on pages 128 and 134.)

[MNR14] Amer E. Mouawad, Naomi Nishimura, and Venkatesh Raman. Vertex cover reconfiguration and beyond. In Hee-Kap Ahn and Chan-Su Shin, editors, *Algorithms and Computation*, pages 452–463, Cham, 2014. Springer International Publishing. (cited on page 149.)

[MS14] Rammohan Mallipeddi and Ponnuthurai Suganthan. Unit commitment – a survey and comparison of conventional and nature inspired algorithms. *International Journal of Bio-Inspired Computation*, 6:71–90, 2014. (cited on page 8.)

[Nem95] Arkadi Nemirovski. Information-based complexity of convex programming, 1995. (cited on page 22.)

[NW88] George Nemhauser and Laurence Wolsey. *Integer and Combinatorial Optimization*. Wiley-Interscience series in discrete mathematics and optimization. John Wiley and Sons, Ltd, 1988. (cited on page 11.)

[NY83] Arkadi Nemirovski and Dawid Yudin. Problem complexity and method efficiency in optimization. *Wiley-Interscience series in discrete mathematics*, 1983. (cited on page 21.)

[PG16] Kai Pan and Yongpei Guan. A polyhedral study of the integrated minimum-up/-down time and ramping polytope. Technical Report 1604.02184, arXiv: Optimization and Control, 2016. (cited on page 9.)

[PG17] Kai Pan and Yongpei Guan. Convex hulls for the unit commitment polytope. Technical Report 1701.08943, arXiv: Optimization and Control, 2017. (cited on page 9.)

[Rad57] Richard Rado. Note on independence functions. *Proceedings of the London Mathematical Society*, s3-7(1):300–320, 1957. (cited on page 18.)

[RT05] Deepak Rajan and Samer Takriti. Minimum up/down polytopes of the unit commitment problem with start-up costs. Technical Report RC23628, IBM Research Report, 2005. (cited on page 8.)

[Sch86] Alexander Schrijver. *Theory of Linear and Integer programming*. Wiley-Interscience Series in Discrete Mathematics and Optimization, 1986. (cited on page 14.)

[SZ21] Sebastian Sager and Clemens Zeile. On mixed-integer optimal control with constrained total variation of the integer control. *Computational Optimization and Applications*, 78(2):575–623, 2021. (cited on page 9.)

[Wel76] Dominic Welsh. *Matroid Theory*. London; New York : Academic Press Inc., dover edition, 1976. (cited on page 19.)