Multi-Stage Selection under Bounded Variation

Christoph Buchheim and Maja Hügging

Department of Mathematics, TU Dortmund University

January 16, 2026

Abstract

We investigate a multi-stage version of the selection problem where the variation between solutions in consecutive stages is either penalized in the objective function or bounded by hard constraints. While the former problem turns out to be tractable, the complexity of the latter problem depends on the type of bounds imposed: When bounding the number of changes of a single item over all stages, the problem turns out to be strongly NP-hard in general, even if we may select only one item per stage and each item may change only twice over all stages. In contrast, when the number of changes at each stage is bounded over all items, the problem can be efficiently solved by reducing it to a minimum-cost flow problem.

1 Introduction

Many combinatorial optimization problems arising in practical applications do not only require to take decisions over a long planning horizon but also to deal with problem parameters changing over time. At the same time, it is often desired that the overall change in the decisions is not too large. The latter may be due to technical reasons, as frequent modifications may require the transport of goods or material from one place to another or lead to the wear and tear of equipment.

In this paper, we will study the well-known Selection problem in a multi-stage setting. The classical, single-stage Selection problem takes as input a number of items n, a number $K \in \{0, \dots, n\}$, and item costs $c_1, \dots, c_n \in \mathbb{R}$; the aim is to select the k cheapest items, i.e., to minimize $c^\top x$ over all $x \in \{0,1\}^n$ with $\sum_{i=1}^n x_i = K$. The Selection problem itself can be solved very efficiently, but its simple structure makes it an ideal candidate for study in combination with additional constraints. This has established the Selection problem as a prominent subject in robust optimization; see, e.g., Chapter 6 in the recent textbook [17] and the references therein.

In the multi-stage setting, we consider a discrete time horizon $\{1, \dots, T\}$, where at each point t in time, a feasible solution $x^{(t)}$ of a given Selection instance has to be chosen. The

objective is a linear function in the incidence vectors $x^{(t)}$ of the chosen items. In addition, we impose an upper bound S on the variation in the sequence $x^{(1)}, \ldots, x^{(T)}$. We call this problem Multi-Stage Selection under Bounded Variation. Such problems arise in production planning, where for each time point $t \in \{1, \ldots, T\}$ the decision maker has to activate exactly K out of n available units, but the change in the selected units may not be too large so as not to decrease the longevity of the technical parts.

1.1 Related literature

In multi-stage optimization, 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 NP-hard, even if the underlying combinatorial problem is tractable. For example, the multi-stage version of Spanning Tree is NP-hard [18] and the multi-stage version of Bipartite Perfect Matching is $n^{1-\varepsilon}$-inapproximable even for $T = 2$, unless $\mathrm{P} = \mathrm{NP}$ [3]. Bampis et al. [5] study the multi-stage version of Knapsack. They present a PTAS and prove that an FPTAS cannot exist, unless $\mathrm{P} = \mathrm{NP}$, even for $T = 2$ 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 T of stages but strongly NP-hard in the general case, even for uniform transition costs or rewards. Chimani et al. investigate a multistage variant of Perfect Matching, where the aim is to minimize the variation, presenting NP-hardness results and approximation algorithms [11], and of Shortest Path, where they evaluate different solution approaches experimentally [10]. More general approaches based on linear programming [2] and on a rounding scheme [1] have also been proposed, leading to new approximation results for several combinatorial multistage problems; another general framework for subgraph problems is given in [12]. Moreover, Bampis et al. [4] investigate an online version of multistage optimization for a large class of optimization problems including Selection. The opposite objective is considered by Kellerhals et al. [20], who aim at diverse solutions rather than similar ones. They present parameterized algorithms for a variety of underlying problems, including Perfect Matching and s-t-Path.

In this paper, we study the multi-stage version of Selection with transition costs and show that this problem can, in fact, be solved efficiently via linear programming; see Section 2. However, our focus is on a more restrictive variant of Multi-Stage Selection: instead of considering transition costs or rewards in the objective function, we introduce hard bounds on the variation, i.e., the change in the sequence of solutions. More precisely, we propose two possible definitions of the variation: we will consider a *time-wise variation* that bounds the symmetric difference of each two consecutive solutions and a *switch-wise variation* that, for each component, bounds the total number of times the respective com-

ponent may change over time. While research concerning switch-wise variation is very rare, hard constraints on the time-wise variation have also been considered in the multi-stage literature. Fluschnik et al. study a variety of optimization problems in this framework. The input is an instance varying over time, given by temporal layers, and the objective is to find a feasible solution for each layer while ensuring that two solutions of consecutive layers do not differ too much in their symmetric difference. Fluschnik et al. show that this problem is NP-hard even under strong restrictions on the number of layers or the allowed number of changes, for 2-SAT [13], 2-Coloring [14], Vertex Cover [15], and s-t-Path [16], while also investigating fixed-parameter tractability. Similar results are obtained for a committee election problem by Bredereck et al. [8].

A related research area are reconfiguration problems, which are based on combinatorial or graph-theoretical problems. Given two solutions X, Y to the underlying problem, the respective reconfiguration problem then asks whether there exists a sequence of feasible solutions $X = X_0, X_1 \ldots, X_T = Y$ that sequentially transforms X into Y under the constraint that solution X_{i+1} can be obtained from X_i by certain reconfiguration operations. As an example, Lokshtanov and Mouawad [22] study Stable Set Reconfiguration. Among others, they consider the variant where each X_i must be a stable set of size at least k in a given graph and only a single vertex per step may be exchanged; the resulting decision problem is NP-complete. This can be interpreted as a very restricted case of our time-wise variation model with at most two changes per time-step and an objective function that is only used to fix the first and last solution. However, the optimization version is already NP-hard for $T = 1$ in case of the Stable Set problem. At the same time, the corresponding reconfiguration problem for Selection is trivial.

Lendl et al. [21] consider a generalization of Multi-Stage Selection under Bounded Variation where feasible solutions are bases of an arbitrary matroid; however, their study is restricted to very few stages. More precisely, they consider $T = 2$ stages and a time-wise variation that is bounded from below or above, or must be matched exactly. The authors 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 NP-hard.

1.2 Problem formulation and contribution

Despite the connections to existing literature discussed above, we are not aware of any results for Multi-Stage Selection (as a special case of Multi-Stage Knapsack) or for Multi-Stage Selection under Bounded Variation. This paper aims to close this gap. We will not only study the less restrictive multi-stage variant of Selection with transition cost, but also show that the definition of the variation has an immense impact on the complexity of the hard-constrained problem variant. For the following, we introduce the shorthand notation $[k] := \{1, \ldots, k\}$ and $[k]_0 := \{0, \ldots, k\}$ for $k \in \mathbb{N}_0$.

To formalize the problem, we consider a discrete time horizon $t \in [T]$. For each stage t, one must choose a feasible solution $x^{(t)}$ from a given feasible set $X^{(t)} \subseteq \{0,1\}^n$. The *variation* is measured by a function $\mathrm{Var}\colon \bigtimes_{t=1}^{T} X^{(t)} \to \mathbb{N}_0^\ell$, where $\mathbb{N}_0$ denotes the natural numbers including zero and ℓ depends on the type of variation, as discussed below. Moreover, we define $\bigtimes_{t=1}^{T} X^{(t)} := X^{(1)} \times \cdots \times X^{(T)}$. We are either given a penalty parameter $\kappa \in \mathbb{R}^\ell$, or an upper bound $S \in \mathbb{R}^\ell$.

For the problem variant Multi-Stage Selection with transition costs κ, we consider

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} {c^{(t)}}^\top x^{(t)} + \kappa^\top \mathrm{Var}(x^{(1)}, \ldots, x^{(T)}) \\ \text{s.t.} \quad & x^{(t)} \in X^{(t)} \quad \forall t \in [T] \,, \end{aligned} \tag{PEN}$$

while the problem Multi-Stage Selection under Bounded Variation is defined as

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} {c^{(t)}}^\top x^{(t)} \\ \text{s.t.} \quad & \mathrm{Var}\left(x^{(1)}, \ldots, x^{(T)}\right) \leq S \\ & x^{(t)} \in X^{(t)} \quad \forall t \in [T] \,. \end{aligned} \tag{BND}$$

In both cases, the vector $c^{(t)} \in \mathbb{R}^n$ defines the objective function at stage $t \in [T]$.

For each component $i \in [n]$, we can interpret the binary sequence $(x_i^{(1)}, \ldots, x_i^{(T)})$ as a *switch* that can either be "on" at stage t (if $x_i^{(t)} = 1$) or "off" (if $x_i^{(t)} = 0$). In this interpretation, the problems PEN and BND consist in operating n parallel switches, which we imagine as *horizontal*, that are connected to each other through *vertical constraints* for each stage.

A special case of PEN and BND has been investigated in [9]. It corresponds to the case with only one switch (and therefore without vertical constraints), i.e., where $n = 1$ and $X^{(t)} = \{0,1\}$ for all $t \in [T]$. In this special case, the resulting problem turned out to be polynomially solvable, either with a dynamic programming approach or a much faster *merging*-algorithm. In [9], the variation measure is defined as

$$\mathrm{Var}(x_1^{(1)}, \ldots, x_1^{(T)}) := \sum_{t=1}^{T-1} |x_1^{(t+1)} - x_1^{(t)}| \,.$$

In the single switch setting, this is the only natural definition, and the same is true for PEN. However, since our setting allows for more than one switch, we now have more than one reasonable possibility to measure the variation in BND. In this paper, we will discuss the following two types of variation: If we speak of *switch-wise variation*, we refer to Problem BND with the variation function

$$\mathrm{Var}_{sw}\colon \bigtimes_{t=1}^{T} X^{(t)} \longrightarrow \mathbb{N}_0^n, \quad \mathrm{Var}_{sw}\left(x^{(1)}, \ldots, x^{(T)}\right)_i := \sum_{t=1}^{T-1} |x_i^{(t+1)} - x_i^{(t)}| \quad \forall i \in [n] \,.$$

In other words, each switch i has an individual upper bound S_i on the number of its changes. If we instead speak of *time-wise variation*, we consider the variation function

$$\mathrm{Var}_{tw}\colon \bigtimes_{t=1}^{T} X^{(t)} \longrightarrow \mathbb{N}_0^{T-1}, \quad \mathrm{Var}_{tw}\left(x^{(1)},\dots,x^{(T)}\right)_t := \sum_{i=1}^{n} |x_i^{(t+1)} - x_i^{(t)}| \quad \forall t \in [T-1]$$

upper bounding the changes at each stage t by S_t.

As in [9], we will also consider a variant of BND where all switches start being "off". In this case, we formally include the fictitious time point $t = 0$ along with the constraint

$$x_i^{(0)} = 0 \quad \forall i \in [n] \,.$$

This can be considered a special case of BND on the time horizon $\{0, 1, \dots, T\}$, where the fixing to zero can be achieved by either choosing the objective coefficients $c^{(0)}$ large enough (in case $0 \in X^{(0)}$) or by defining $X^{(0)} := \{0\}$. We will denote the respective problem as (BND_0). This variant can be interpreted as modeling a "start-up variation" for the combinatorial structure that must be built in the first stage $t = 1$.

The motivation of this paper is to study the complexity of Problems PEN and BND for both presented measures of variation. From the results in [9], it easily follows that the case $X^{(t)} = \{0,1\}^n$ for all $t \in [T]$ is tractable for switch-wise variation, while the same result can be shown for time-wise variation by means of a totally unimodular extended formulation. However, it is not obvious how the complexity of the multiple-switch problem is influenced by the introduction of additional constraints on the stage-wise feasible sets, in particular if the latter are *easy* in the sense that linear optimization over each combinatorial set $X^{(t)}$ can be performed in polynomial time. As mentioned above, we will restrict ourselves to a selection type constraint for each point in time:

$$X^{(t)} := \left\{ x \in \{0,1\}^n \colon \sum_{i=1}^{n} x_i = K^{(t)} \right\} \quad \forall t \in [T] \,. \tag{SEL}$$

This is arguably the second-simplest combinatorial structure one can think of, and it is an interesting question whether the combination of such a simple vertical constraint and another simple horizontal constraint (the bound on the variation) remains polynomial-time solvable or not. In fact, we will show that the complexity of the respective problem highly depends on the chosen type of variation measure: while bounding the switch-wise variation renders the problem strongly NP-hard in general, as discussed in Section 3.1, time-wise variation can be handled efficiently; see Section 3.2.

2 Multi-Stage Selection with transition cost

In the absence of vertical constraints, the problem with a hard bound on the variation is significantly more involved than the corresponding problem with penalized variation,

although both versions of the problem are tractable [9]. Motivated by this, we will first study the penalty version PEN with a selection constraint as in SEL. We will show the problem to be polynomial-time solvable by linear programming.

First, note that the two types of variation defined above agree in this setting, since only the total number of switchings is relevant in the penalty version of the problem. In both cases, the problem can be written as

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} {c^{(t)}}^{\top} x^{(t)} && + \kappa \sum_{i=1}^{n} \sum_{t=1}^{T-1} |x_i^{(t+1)} - x_i^{(t)}| \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i^{(t)} = K^{(t)} && \forall t \in [T] \\ & x^{(t)} \in \{0,1\}^n && \forall t \in [T] \end{aligned}$$

with $\kappa > 0$. To show that this problem can be solved efficiently, we first rewrite it as follows: we introduce new variables $u_i^{(t-1,t)}$ and $d_i^{(t-1,t)}$ for all $i \in [n]$ and $t \in \{2, \dots, T\}$ that are forced to take a value of at least one if the switch i is turned "on" or "off" at time t, respectively. These auxiliary variables can be weighted with κ in the objective function. In the following, we consider a more general problem formulation where not only every switching may incur a different penalty, but even switching "on" and "off" can lead to different penalties. Using penalty parameters $\mu_i^{(t-1,t)}, \lambda_i^{(t-1,t)} > 0$, we arrive at

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} {c^{(t)}}^{\top} x_i^{(t)} + \sum_{t=2}^{T} \sum_{i=1}^{n} \left(\mu_i^{(t-1,t)} u_i^{(t-1,t)} + \lambda_i^{(t-1,t)} d_i^{(t-1,t)} \right) \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i^{(t)} = K^{(t)} && \forall t \in [T] && (1) \\ & - x_i^{(t-1)} + x_i^{(t)} - u_i^{(t-1,t)} \leq 0 && \forall t \in \{2, \dots, T\},\ i \in [n] && (2) \\ & + x_i^{(t-1)} - x_i^{(t)} - d_i^{(t-1,t)} \leq 0 && \forall t \in \{2, \dots, T\},\ i \in [n] && (3) \\ & u_i^{(t-1,t)}, d_i^{(t-1,t)} \geq 0 && \forall t \in \{2, \dots, T\},\ i \in [n] \\ & x^{(t)} \in \{0,1\}^n && \forall t \in [T] \end{aligned} \qquad (\text{PEN}^{\text{sel}})$$

The central result of this section will be that the constraint matrix corresponding to the constraints (1), (2), and (3) is totally unimodular, implying that Problem PEN^{sel} can be solved efficiently. For the definition of total unimodularity and a comprehensive discussion of its consequences, we refer the reader to [23, Chapter 19].

Lemma 2.1. *The constraint matrix M corresponding to the constraints* (1)*,* (2)*, and* (3) *is totally unimodular.*

Proof. Let M be the matrix that models the constraints (1), (2), and (3). We will prove by induction over T that, for any subset R of rows in M, there exists a partition $R = R_1 \dot{\cup} R_2$ 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 $i_1 \in [n]$ and $t \in \{2, \ldots, T\}$ and a row r_2 of constraint (3) for some $i_2 \in [n]$ and the same t such that $i_1 \neq i_2$, then r_1 and r_2 belong to different parts of the partition.

We first claim that it suffices to show this for all sets R satisfying the following condition:

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

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.

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

Now, let $\tilde{R}$ refer to the subset of rows in R that correspond to the constraints (1), (2), and (3) for $i \in [n]$ and $t \in [T-1]$. Let $\tilde{M}$ denote the restriction of M to the rows in $\tilde{R}$. Since assumption $(\star)$ holds for R, it also holds for the subset $\tilde{R}$. Then, according to our 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, we extend the partition $\tilde{R}_1 \dot{\cup} \tilde{R}_2$ of $\tilde{R}$ to a partition $R_1 \dot{\cup} R_2$ of R. We define $R_1 := \tilde{R}_1$ and $R_2 := \tilde{R}_2$ and it remains to assign the rows $R \setminus \tilde{R}$, which represent the constraints (1), (2), and (3) for $i \in [n]$ and $t = T$. By our induction hypothesis (ii), we know 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. We 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, we assign all the 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, we 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.

In both cases, the resulting partition $R_1 \dot{\cup} R_2$ satisfies (i), (ii), and $(\star)$: Indeed,$(\star)$ is ensured by assumption, while the fact that (ii) is satisfied follows immediately from the definition of R_1 and R_2. It remains to show that (i) is satisfied, too. For sake of brevity and readability, let $row_{(k),t}$ for all $k \in \{1, 2, 3\}$ and all $t \in [T]$ (or $t \in \{2, \ldots, T\}$, respectively) refer to the row of M that models the constraint (k) for stage t.

In order to show that (i) is satisfied in both cases, we describe a case distinction depending on what rows are contained in R. More precisely, we compute the values of $\sum_{r\in R_1} m_{rj}$ and $\sum_{r\in R_2} m_{rj}$ and show that they differ by at most one.

Observe that it suffices to show that (i) is satisfied for the columns corresponding to variables $x_1^{(t)}, \ldots, x_n^{(t)}$ for $t \in \{T-1, T\}$, since the induction hypothesis ensures that (i) is satisfied for all other columns. To argue that (i) is satisfied in the first case, let j be a column corresponding to a variable $x_i^{(T-1)}$ for some $i \in [n]$.

1. If R contains both $row_{(2),T-1}$ and $row_{(3),T}$, then by definition of R_1, $\sum_{r\in R_1} m_{rj} = 2$. Note that $(\star)$ implies that R neither contains $row_{(3),T-1}$ nor $row_{(2),T}$. Thus, the definition of R_2 yields $\sum_{r\in R_2} m_{rj} = 1$.

2. If R contains $row_{(2),T-1}$ but not $row_{(3),T}$ (or vice versa), then $\sum_{r\in R_1} m_{rj} = 1$ by definition of the partition. Moreover, $(\star)$ implies that R cannot contain $row_{(3),T-1}$ (or $row_{(2),T}$ for the reverse case). Then, the definition of the partition yields either $\sum_{r\in R_2} m_{rj} = 0$ if R contains $row_{(2),T}$ (or $row_{(3),T-1}$ for the reverse case) or $\sum_{r\in R_2} m_{rj} = 1$ if R does *not* contain $row_{(2),T}$ (or $row_{(3),T-1}$ for the reverse case).

3. If R neither contains $row_{(2),T-1}$ nor $row_{(3),T}$, then $\sum_{r\in R_1} m_{rj} = 0$ by definition of R_1. Because of $(\star)$, R neither contains $row_{(3),T-1}$ nor $row_{(2),T}$. By definition of the partition, the sum $\sum_{r\in R_2} m_{rj}$ equals 1 if R neither contains $row_{(3),T-1}$ nor $row_{(2),T}$, it equals 0 if R contains exactly one of the two rows, and, finally, it equals -1 if it contains both rows simultaneously.

Clearly, the difference between the two sums $\sum_{r\in R_1} m_{rj}$ and $\sum_{r\in R_2} m_{rj}$ is an element of $\{-1, 0, 1\}$ in all of the above cases. It thus remains to verify that (i) is satisfied for any column j that corresponds to a variable $x_i^{(T)}$ for some $i \in [n]$, too. To this end, the definition of the partition yields the following:

$$\sum_{r\in R_1} m_{rj} = \begin{cases} 0, & \text{if } R \text{ contains either both } row_{(1),T} \text{ and } row_{(3),T} \text{ or neither one,} \\ 1, & \text{if } R \text{ contains } row_{(1),T} \text{ but not } row_{(3),T}, \\ -1, & \text{if } R \text{ contains } row_{(3),T} \text{ but not } row_{(1),T}, \end{cases}$$

and

$$\sum_{r\in R_2} m_{rj} = \begin{cases} 0, & \text{if } R \text{ does not contain } row_{(2),T}, \\ 1, & \text{if } R \text{ contains } row_{(2),T}. \end{cases}$$

Since assumption $(\star)$ prevents the simultaneous containment of both $row_{(2),T}$ and $row_{(3),T}$ in R, the above shows that the difference $\sum_{r\in R_1} m_{rj} - \sum_{r\in R_2} m_{rj}$ yields an element of $\{-1, 0, 1\}$. Now, we consider the definition of the partition for the second case and show that (i) is satisfied using the same logic. First, for any column j corresponding to a variable $x_i^{(T-1)}$ for some $i \in [n]$, we distinguish the following cases:

1. If R contains both $row_{(2),T-1}$ and $row_{(2),T}$, then $\sum_{r\in R_1} m_{rj} = 0$ by definition of the partition. Moreover, $(\star)$ implies that R contains neither $row_{(3),T-1}$ nor $row_{(3),T}$. By definition of R_2, it follows that $\sum_{r\in R_2} m_{rj} = 0$.

2. If R contains $row_{(2),T-1}$ but not $row_{(2),T}$ (or vice versa), then $(\star)$ yields that R does not contain $row_{(3),T-1}$ (or $row_{(3),T}$ for the reverse case) and the definition of the partition implies $\sum_{r\in R_1} m_{rj} = 1$ (or -1 for the reverse case). Then, the definition of R_2 yields $\sum_{r\in R_2} m_{rj} = 1$ (or -1 for the reverse case) if R contains $row_{(3),T}$ (or $row_{(3),T-1}$ for the reverse case) or zero if $row_{(3),T}$ (or $row_{(3),T-1}$ for the reverse case) is not contained.

3. If R contains neither $row_{(2),T-1}$ nor $row_{(2),T}$, then $\sum_{r\in R_1} m_{rj} = 0$ by definition of the partition. Furthermore, $\sum_{r\in R_2} m_{rj}$ equals -1 if R contains $row_{(3),T-1}$ but not $row_{(3),T}$, it equals -1 if it is the other way around and, finally, it equals zero if neither of these two rows are contained.

Once again, the difference between the two sums $\sum_{r\in R_1} m_{rj}$ and $\sum_{r\in R_2} m_{rj}$ is an element of $\{-1, 0, 1\}$ in all of the above cases. It thus remains to verify that (i) is satisfied for any column j that corresponds to a variable $x_i^{(T)}$ for some $i \in [n]$, too. For any of these remaining columns, the definition of the partition yields

$$\sum_{r\in R_1} m_{rj} = \begin{cases} 1, & \text{if } R \text{ contains } row_{(2),T}, \\ 0, & \text{otherwise} \end{cases}$$

and

$$\sum_{r\in R_2} m_{rj} = \begin{cases} 0, & \text{if } R \text{ contains } row_{(1),T} \text{ and } row_{(3),T}, \\ 1, & \text{if } R \text{ contains } row_{(1),T} \text{ but not } row_{(3),T}, \\ -1, & \text{if } R \text{ contains } row_{(3),T} \text{ but not } row_{(1),T}, \\ 0, & \text{if } R \text{ contains neither } row_{(1),T} \text{ nor } row_{(3),T}. \end{cases}$$

Since $(\star)$ prevents the simultaneous containment of $row_{(2),T}$ and $row_{(3),T}$, the difference between the above two sums is once again an element of $\{-1, 0, 1\}$.

This concludes our proof, since condition (i) implies total unimodularity by the Ghouila-Houri criterion [6]. □

Since the right-hand side of PEN$^{\text{sel}}$ is integer and all constraints in PEN$^{\text{sel}}$ other than (1) to (3) impose only upper or lower bounds, we may relax the binarity constraint for x. We obtain the following result as an immediate consequence:

Theorem 2.2. *Problem* PEN$^{\text{sel}}$ *can be solved in polynomial time using linear programming.*

To conclude this section, we mention that the tractability of variants of the selection problem is often shown by total unimodularity arguments similar to the one used above; see [19] for a similar result.

3 Multi-Stage Selection under Bounded Variation

The integer linear program PEN^{sel} can easily be adapted to the case where variation is bounded rather than penalized. Indeed, in the time-wise case, we could add the constraints

$$\sum_{i=1}^{n} \left(u_i^{(t-1,t)} + d_i^{(t-1,t)} \right) \leq S_{t-1} \quad \forall t \in \{2, \ldots, T\} \, .$$

However, in this model, total unimodularity is lost. While this does not necessarily mean that integrality is lost as well, the following tiny example shows that we cannot omit integrality constraints any more: Let $n = 2$, $T = 2$, $S = 1$, and $K = 1$. Define $c_i^{(t)} = -1$ if $i = t$ and $c_i^{(t)} = 0$ otherwise. It is easily verified that all optimal integer solutions have value -1 then, while setting $x_1^{(1)} = x_1^{(2)} = \frac{1}{2}$, $x_2^{(1)} = 0$, and $x_2^{(2)} = 1$ is feasible for the LP relaxation and yields a better objective value of $-\frac{3}{2}$.

For switch-wise variation, the LP relaxation of the corresponding integer program can also have fractional optimal solutions, even in the case of a single switch (where the selection constraint is redundant); see [9, Ex. 4.2].

As we will see in the next section, the penalty version of our problem is in fact strictly easier than the version with a bound on the switch-wise variation, unless P = NP, while the problem version with a bound on the time-wise variation remains polynomial-time solvable, as shown in Section 3.2.

3.1 Switch-wise variation

This section is dedicated to Problem BND with *switch-wise variation* and a selection constraint as in SEL. The resulting problem then reads

$$\begin{aligned}
\min \quad & \sum_{t=1}^{T} {c^{(t)}}^{\top} x^{(t)} & & \\
\text{s.t.} \quad & \sum_{t=1}^{T-1} |x_i^{(t+1)} - x_i^{(t)}| \leq S_i & \forall i \in [n] & \qquad (\text{BND}_{\text{sw}}^{\text{sel}}) \\
& \sum_{i=1}^{n} x_i^{(t)} = K^{(t)} & \forall t \in [T] & \\
& x^{(t)} \in \{0,1\}^n & \forall t \in [T] &
\end{aligned}$$

We refer to the variant with fixation to zero at time $t = 0$ as $\text{BND}_{0,\text{sw}}^{\text{sel}}$. First, we state a simple observation:

Remark. *Without the vertical constraint, Problem* $\text{BND}_{\text{sw}}^{\text{sel}}$ *decomposes into* n *disjoint single-switch problems as in [9] and is therefore polynomially solvable.*

However, in the presence of selection constraints, we have a stark contrast to the previous remark, as we will prove that the inclusion of such constraints renders Problem $\text{BND}_{\text{sw}}^{\text{sel}}$ strongly NP-hard, even if $K^{(t)} = 1$ for all $t \in [T]$ and $S_i = 2$ for all $i \in [n]$, i.e., if at any time at most one switch may be turned on and every switch may change at most twice. We show this statement by a reduction from the following decision problem discussed in [7]:

Problem 3.1 (Interval Selection)**.** *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 must be selected. To schedule several jobs, no two selected intervals on the same machine must intersect.*

Question: Is it 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 [7] prove the following result:

Theorem 3.2. Interval Selection *is 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 [7], 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 large 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 statement above as follows:

Theorem 3.3. Interval Selection *is NP-complete, even if the number of machines is fixed to three, all intervals have integer endpoints, and each interval has length two.*

With the help of this result, we are now able to prove NP-hardness of $\text{BND}_{\text{sw}}^{\text{sel}}$.

Theorem 3.4. *Problem* $\text{BND}_{\text{sw}}^{\text{sel}}$ *is strongly NP-hard, even if* $K^{(t)} = 1$ *for all* $t \in [T]$ *and* $S_i = 2$ *for all* $i \in [n]$*.*

Proof. Let an instance of Interval Selection as in Theorem 3.3 be given, consisting of n jobs, $m = 3$ machines, and intervals I_{ij} having length two and integer endpoints for each job $i \in [n]$ and each machine $j = 1, 2, 3$. Let T^* denote the smallest length of the time horizon on the real line containing all these intervals. We divide this time horizon into T^* segments of length one and point out 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 $[T^*]$.

We now define an instance of Problem $\text{BND}_{\text{sw}}^{\text{sel}}$. 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 stage 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.

We define a switch i for each job $i \in [n]$ as well as T dummy switches $i \in \{n+1, ..., n+T\}$ and restrict their variation to two, i.e., $S_i := 2$ for $i \in [n+T]$. For $i \in [n]$, the objective coefficients $c_i^{(t)}$ will serve to model the intervals I_{ij}. More precisely, for $i \in [n]$ we define

$$c_i^{(t)} = \begin{cases} -1, & \text{if } t \in \bigcup_{j=1,2,3} \bar{I}_{ij} \\ 3 & \text{otherwise,} \end{cases}$$

where $\bar{I}_{ij} := I_{ij} + (j-1)(T^*+1) + 1$ is the interval I_{ij} shifted to the jth block. For the dummy switches, all objective coefficients are defined as zero, i.e., $c_i^{(t)} = 0$ for all $t \in [T]$ and $i = n+1, ..., n+T$. Finally, we set the selection bounds to one, i.e., $K^{(t)} := 1$ for all $t \in [T]$.

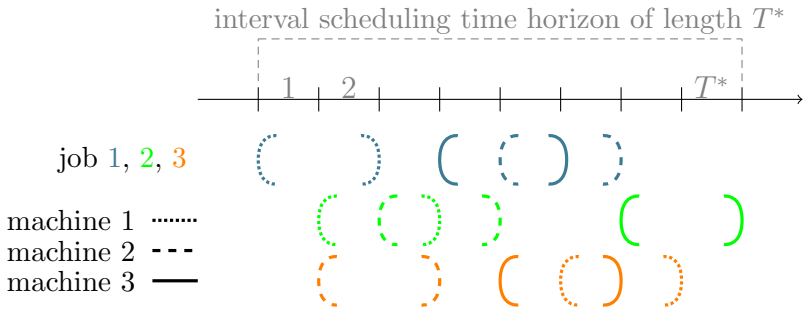
Observe that the dummy switches ensure that the instance of Problem $\text{BND}_{\text{sw}}^{\text{sel}}$ has an optimal value of at most zero. Indeed, the solution in which the first n switches are "off" over all periods and each dummy switch $n+t$ for $t \in [T]$ is only "on" at stage t is feasible with objective value zero. Moreover, any partial solution on the first n switches that satisfies the variation constraints but leaves a slack in the selection constraint at some stage t can uniquely be extended without increasing the objective value by defining dummy switch $n+t$ to be "on" at stage t only. As a result, the definition of $c_i^{(t)}$, $i = 1 \in [n]$, together with $S_i = 2$ implies that in an optimum solution of $\text{BND}_{\text{sw}}^{\text{sel}}$ each switch can only be active within at most one of these three intervals $\bar{I}_{ij}$, $j = 1, 2, 3$. Consequently, we derive a lower bound of $-2n$ for the optimum value of Problem $\text{BND}_{\text{sw}}^{\text{sel}}$.

We now claim that the original instance of Interval Scheduling is a yes-instance if and only if the optimum value of the constructed instance of Problem $\text{BND}_{\text{sw}}^{\text{sel}}$ is exactly $-2n$, i.e., it reaches this lower bound. The construction and the idea behind the definition of the objective c is illustrated in Figure 1, where the dummy switches are omitted.

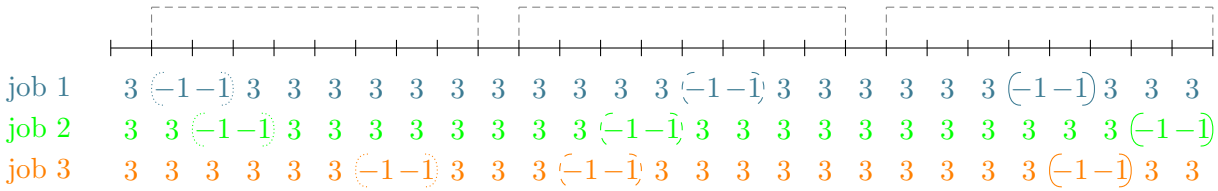
Given a yes-instance of Interval Scheduling, let I_{ij_i} be the selected interval for job i. We then set set $x_i^{(t)} = 1$ for $i \in [n]$ if and only if $t \in \bar{I}_{ij_i}$ and extend this partial solution by defining the dummy switch $n+t$ to only be "on" at stage t if the selection constraint at this stage is not already satisfied by the partial solution over the first n switches. Otherwise, we define the dummy switch $n+t$ as "off" over all stages. By definition of the objective, this extension does not change the objective value.

This solution for Problem $\text{BND}_{\text{sw}}^{\text{sel}}$ is feasible, since each switch switches at most twice. 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 point in time. The objective value is $-2n$ and thus agrees with the lower bound.

Now, assume that this lower bound is achieved. Then, as argued above, each of the first n switches must contribute exactly -2 to the objective value, therefore each switch i is "on" for exactly two consecutive stages, namely those in $\bar{I}_{ij_i}$ for some $j_i \in \{1, 2, 3\}$. We then



(a) An instance of Interval Selection with three jobs.



(b) Illustration of the objective of the instance of $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$.

Figure 1: Illustration of the reduction in the proof of Theorem 3.4.

define a feasible solution for Interval Scheduling by scheduling job i on the respective machine j_i. Since the selection constraint is satisfied at each point in time, no two intervals corresponding to jobs scheduled on the same machine can intersect. Hence, the instance of Interval Scheduling is a yes-instance. □

The construction in the proof of Theorem 3.4 can be used to also work for $K^{(0)} = 0$ and $K^{(t)} = 1$, $t > 0$. In other words, it can be adapted to also show the strong NP-hardness of the special case $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$ in which all switches are fixed to zero in the first stage.

We point out that in the context of Theorem 3.4, the size of each feasible set $X^{(t)}$ is given by n. Theorem 3.4 thus shows that we cannot expect polynomial-time algorithms for $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ even for small values of S and for polynomially large sets $X^{(t)}$. We will now investigate the cases of a small number of switches n or a small number of stages T. For this, we will devise a dynamic programming scheme; see [8, Theorem 5] for a similar approach.

In the following description, we introduce the vector $\sigma \in \times_{i=1}^{n}[S_i]_0$, where S_i is the i-th component of $S \in \mathbb{N}_0^n$ that gives the upper bounds on the variation of switch i. The vector σ represents an adjusted upper bound on the variation for all switches. Moreover, we use a vector $b \in \{0,1\}^n$ that represents the states of the switches in the stage $\bar{t}$ currently considered. We use the shorthand notation $|x-y| := (|x_1-y_1|,\dots,|x_n-y_n|)^\top$ for two vectors x, y of the same dimension. Now, in our dynamic programming scheme, we recursively

compute the optimum values of the following subproblems:

$$\begin{aligned} c^*(\bar{t},\sigma,b) := \min \quad & \sum_{t=1}^{\bar{t}} {c^{(t)}}^\top x^{(t)} \\ \text{s.t.} \quad & \sum_{t=1}^{\bar{t}-1} |x_i^{(t+1)} - x_i^{(t)}| \le \sigma_i \quad \forall i \in [n] \\ & x^{(t)} \in X^{(t)} \qquad\qquad\qquad \forall t \in [\bar{t}] \\ & x^{(\bar{t})} = b \end{aligned}$$

We define the initial values as follows:

$$c^*(1,\sigma,b) := \begin{cases} {c^{(1)}}^\top b & \text{if } \sigma \in \bigtimes_{i=1}^n [S_i]_0,\ b \in X^{(1)} \\ \infty & \text{otherwise.} \end{cases}$$

Then, for $\bar{t} = 2,\dots,T$, we recursively compute

$$c^*(\bar{t},\sigma,b) := \min_{\substack{x \in X^{(\bar{t}-1)} \\ |x-b| \le \sigma}} c^*(\bar{t}-1, \sigma - |x-b|, x) + \sum_{i=1}^n c_i^{(\bar{t})} b_i$$

for all $\sigma \in \bigtimes_{i=1}^n \{0,1,\dots,S_i\}$ and $b \in X^{(\bar{t})}$, where $|x-b| \le \sigma$ is meant componentwise. The optimum value of Problem $\mathrm{BND}_{\mathrm{sw}}^{\mathrm{sel}}$ is then given by

$$\min_{x \in X^{(T)}} c^*(T,S,x)\ .$$

We now analyze the running time of this algorithm. For fixed $\bar{t}$ and σ and for polynomial-sized sets $X^{(t)}$, we can 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 $X^{(\bar{t})}$. However, we have to compute up to $|\bigtimes_{i=1}^n [S_i]_0|$ such values. In summary, we can roughly estimate the running time of the scheme by

$$\mathcal{O}\Big(\big(\max_{t\in[T]} |X^{(t)}| \big)^2 \cdot \big(\max_{i\in[n]} S_i + 1 \big)^n \cdot T \Big)\ .$$

Since $S_i \le T-1$ without loss of generality, we can also bound the running time by

$$\mathcal{O}\Big(\big(\max_{t\in[T]} |X^{(t)}| \big)^2 \cdot T^{n+1} \Big)\ .$$

In particular, we obtain the following result:

Theorem 3.5. *If n is constant, then Problem* $\mathrm{BND}_{\mathrm{sw}}^{\mathrm{sel}}$ *is polynomially solvable.*

We emphasize another advantage of this dynamic programming scheme:

Remark. *The dynamic programming scheme presented above works independently of the combinatorial constraints given by the sets $X^{(t)}$. Moreover, it can also be adapted to the special case $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$. For this, it suffices to substitute the definition of the initial values for $\bar{t} = 0$ in the dynamic programming scheme by*

$$\begin{aligned} c^*(0, \sigma, 0_n) &:= 0 && \text{for all } \sigma \in \bigtimes_{i=1}^{n} [S_i]_0, \text{ and} \\ c^*(0, \sigma, x) &:= \infty && \text{for all } \sigma \in \bigtimes_{i=1}^{n} [S_i]_0,\ x \in X^{(0)} \setminus \{0_n\}\ . \end{aligned}$$

The remaining values can be computed using the same recursion formula for $t \in [T]$.

Next, we describe a second dynamic programming scheme that leads to polynomial-time solvability if the length of the time horizon T is fixed. Our algorithm is inspired by the approach used in [5] for the multi-stage knapsack problem for a fixed number of stages.

For a single switch $i \in [n]$, let $\mathcal{S}_i \subseteq \{0,1\}^T$ be the set of its feasible switching patterns, i.e., $(x_i^{(1)}, \ldots, x_i^{(T)}) \in \mathcal{S}_i$ if and only if $\sum_{i=1}^{T-1} |x_i^{(t+1)} - x_i^{(t)}| \leq S_i$. Note that this set can be obtained by enumeration in constant time, if T is fixed.

Now for values $k^{(t)} \in [K^{(t)}]_0$, $t \in [T]$, and $j \in [n]$ define

$$\begin{aligned} c^*(k^{(1)}, \ldots, k^{(T)}, j) = \quad \min \quad & \sum_{i=1}^{j} \sum_{t=1}^{T} c_i^{(t)} x_i^{(t)} \\ \text{s.t.} \quad & \sum_{t=1}^{T-1} |x_i^{(t+1)} - x_i^{(t)}| \leq S_i && \forall i \in [j] \\ & \sum_{i=1}^{j} x_i^{(t)} = k^{(t)} && \forall t \in [T] \\ & x_i^{(t)} \in \{0,1\} && \forall i \in [j],\ t \in [T] \end{aligned}$$

In the above problem, we consider only the first j switches and at each stage t, there is a selection constraint with selection bound $k^{(t)}$. We define initial values for $j = 1$ and for all $\left(k^{(1)}, \ldots, k^{(T)}\right) \in \bigtimes_{t=1}^{T} [K^{(t)}]_0$ through

$$c^*(k^{(1)}, \ldots, k^{(T)}, 1) := \min \left\{ \sum_{t=1}^{T} c_1^{(t)} x_1^{(t)} \colon (x_1^{(1)}, \ldots, x_1^{(T)}) \in \mathcal{S}_1,\ x_1^{(t)} = k^{(t)}\ \forall t \in [T] \right\}.$$

Now, for $j = 2, \ldots, n$, we compute the optimal values with the help of the following recursion formula for all $(k^{(1)}, \ldots, k^{(T)}) \in \bigtimes_{t=1}^{T} [K^{(t)}]_0$:

$$c^*(k^{(1)}, \ldots, k^{(T)}, j) = \min_{(x_j^{(1)}, \ldots, x_j^{(T)}) \in \mathcal{S}_j} c^*(k^{(1)} - x_j^{(1)}, \ldots, k^{(T)} - x_j^{(T)}, j-1) + \sum_{t=1}^{T} c_j^{(t)} x_j^{(t)}$$

The optimal value of Problem $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ is then given by $c^*(K^{(1)}, \ldots, K^{(T)}, n)$.

Theorem 3.6. *If T is constant, then both Problem* $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ *and Problem* $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$ *can be solved in polynomial time.*

Proof. We analyze the running time of the dynamic programming scheme described above. The minimum in the recursion formula can be computed by enumerating all feasible switching patterns, their number is bounded by $\max_{i\in[n]} |\mathcal{S}_i| \leq 2^T$. However, this minimum must be computed for all $(k^{(1)}, \ldots, k^{(T)}) \in \bigtimes_{t=1}^{T} [K^{(t)}]_0$, whose number can be bounded by $(\max_{t\in[T]} K^{(t)} + 1)^T$. In total, we can estimate the running time by

$$\mathcal{O}\Big(n\big(\max_{i\in[n]} |\mathcal{S}_i|\big)\big(\max_{t\in[T]} K^{(t)} + 1\big)^T\Big) .$$

Since $K^{(t)} \leq n$ for all t, this shows the claim for $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$.

For $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$, it suffices to replace the definition of $\mathcal{S}_i$ so that $(x_i^{(1)}, \ldots, x_i^{(T)}) \in \mathcal{S}_i$ if and only if $\sum_{t=0}^{T-1} |x_i^{(t+1)} - x_i^{(t)}| \leq S_i$ and $x_i^{(0)} = 0$, and to include the constraint $x_i^{(0)} = 0$ in all subproblems. The estimation of the required running time is still valid. □

As we have seen above, both variants $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ and $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$ are NP-hard in the strong sense, even for $S_i = 2$ for all $i \in [n]$ and $K^{(t)} = 1$ for all $t \in [T]$. This raises the question whether the complexity of the respective problems changes if $S_i = 1$ for all $i \in [n]$. We first consider an easy special case, namely the Problem $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$ with $S_i = 1$ for all $i \in [n]$ and $K^{(t)} = K$ for all $t \in [T]$. Indeed, the combination of uniform selection bounds together with a switch-wise variation of one for each switch immediately makes the problem $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$ collapse to a classical selection problem over $\{1, ..., n\}$ with $C_i := \sum_{t=1}^{T} c_i^{(t)},\ i \in [n]$. Now, consider $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$ with $S_i = 1$ for all $i \in [n]$ with non-uniform selection bounds. Since each switch is required to be inactive in the beginning, i.e., $x_i^{(0)} = 0$ for all $i \in [n]$, 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 "on" until the last stage. The best possible value that any switch i can achieve is thus given by:

$$\mathrm{val}(i) := \max_{\bar{t}=0,\ldots,T} \left\{ \sum_{t=\bar{t}}^{T} c_i^{(t)},\ 0 \right\} .$$

However, this observation does not help, since a switch i that is selected in an optimal solution may not achieve its individual optimum value $\mathrm{val}(i)$, as the example below shows.

Example 3.7. *Consider the instance of* $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$ *with $n = 2$, $T = 2$, and*

$$c = \begin{pmatrix} 0 & -3 & 0 \\ 0 & -1 & -1 \end{pmatrix} ,$$

where row $i = 1, 2$ and column $t = 0, 1, 2$ contains $c_i^{(t)}$. Moreover, let $S_1 = S_2 = 1$ and set $K^{(1)} = 1$, $K^{(2)} = 2$. We obtain the following (unique) optimal solution:

$$x^* = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

However, we have $val(2) = -2$, achieved for $(x_2^{(0)}, x_2^{(1)}, x_2^{(2)}) = (0, 1, 1)$.

Nevertheless, we show that $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$ can still be solved in this situation, although the solution is a little more involved.

Theorem 3.8. *Problem $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$ with $S_i = 1$ for all $i \in [n]$ polynomially reduces to a minimum-cost flow problem and, hence, can be solved efficiently.*

Proof. Given an instance of $\mathrm{BND}^{\mathrm{sel}}_{0,\mathrm{sw}}$ with $S_i = 1$ for all $i \in [n]$, we construct a network as follows: We define a source α and a sink ω and introduce nodes $\bar{S}_i$ for $i \in [n]$ and nodes $s_i^{(t)}$ for $i \in [n]$ and $t \in [T]_0$. Each node $\bar{S}_i$ is connected with the source a through arc $(\alpha, \bar{S}_i)$ and with each node $s_i^{(t)}$ for $t \in [T]_0$ through arc $(\bar{S}_i, s_i^{(t)})$. A positive flow on the arc $(\alpha, \bar{S}_i)$ will correspond to switch i being switched "on" at some point in time; a positive flow on the arc $(\bar{S}_i, s_i^{(t)})$ will correspond to the switch i being activated in stage t.

For each $t \in [T]_0$, we add a node $v^{(t)}$ and introduce arcs $(s_i^{(t)}, v^{(t)})$ for $i \in [n]$. We connect the nodes $v^{(t)}$ to each other via arcs $(v^{(t)}, v^{(t+1)})$, $t \in [T-1]_0$. Finally, the vertex $v^{(T)}$ is connected to the sink ω through arc $(v^{(T)}, \omega)$. To sum up, we obtain the set of vertices

$$V := \{\bar{S}_i \colon i \in [n]\} \cup \{s_i^{(t)} \colon i \in [n],\ t \in [T]_0\} \cup \{v^{(t)} \colon t \in [T]_0\}$$

and the set of arcs

$$\begin{aligned} A :=& \{(\alpha, \bar{S}_i) \colon i \in [n]\} \cup \{(\bar{S}_i, s_i^{(t)}), (s_i^{(t)}, v^{(t)}) \colon i \in [n],\ t \in [T]_0\} \\ & \cup \{(v^{(t)}, v^{(t+1)}) \colon t \in [T-1]_0\} \cup \{(v^{(T)}, \omega)\} \end{aligned}$$

The costs c of the arcs are defined as zero except for

$$c(\bar{S}_i, s_i^{(t)}) := \sum_{j=t}^{T} c_i^{(j)} \qquad \forall i \in [n],\ t \in [T]_0 \,.$$

The capacities u of all arcs are defined to be one, except for

$$u(v^{(t)}, v^{(t+1)}) := K^{(t)}\ \forall t \in [T-1]_0, \quad u(v^{(T)}, \omega) := K^{(T)}\,;$$

while the lower capacity bound is defined as $K^{(t)}$ for the arcs $(v^{(t)}, v^{(t+1)})$ and zero for all other arcs. Finally, we define the supply $b_\alpha = K^{(T)}$ and the demand $b_\omega = -K^{(T)}$, while all other nodes in the network have zero demand. See Figure 2 for an illustration.

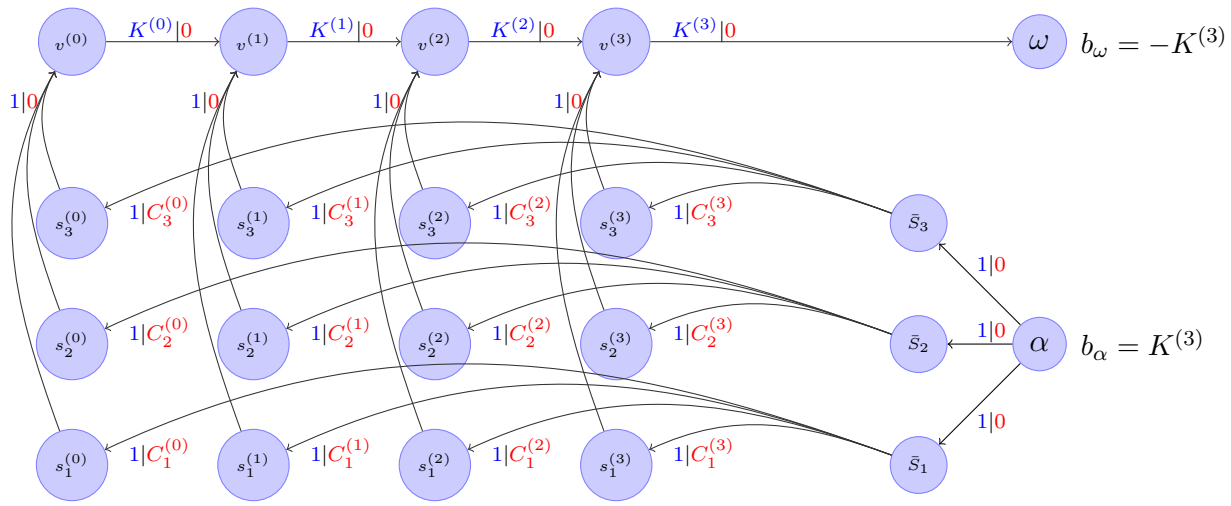


Figure 2: Sketch of the network constructed in the proof of Theorem 3.8, for $n = 3$ and $T = 3$. All edges are labeled by capacity and cost, where $C_i^{(t)} := \sum_{j=t}^{T} c_i^{(j)}$ $\forall t$.

Due to the definition of the capacities in our network, any feasible integer flow f must satisfy the following constraints:

$$f_{(\alpha,\bar{S}_i)} = f_{(\bar{S}_i,s_i^{(t)})} = f_{(s_i^{(t)},v^{(t)})} \in \{0,1\} \qquad \forall i \in [n], t \in [T]$$

$$f_{(v^{(t)},v^{(t+1)})} = \sum_{j=1}^{t}\sum_{i=1}^{n} f_{(s_i^{(j)},v^{(j)})} = K^{(t)} \qquad \forall t \in [T]$$

$$f_{(v^{(T)},\omega)} = \sum_{j=1}^{T}\sum_{i=1}^{n} f_{(s_i^{(j)},v^{(j)})} = \sum_{i=1}^{n} f_{(a,\bar{S}_i)} = K^{(T)}$$

Therefore, a feasible solution x of $\text{BND}^{\text{sel}}_{0,\text{sw}}$ can be transformed into a feasible integer flow f and vice versa using the following equivalence:

$$f_{(\bar{S}_i,s_i^{(t)})} = 1 \quad \Longleftrightarrow \quad x_i^{(t-1)} = 0,\ x_i^{(t)} = 1\ .$$

By definition of the costs c, both solutions yield the same objective value. Since all capacities u are integer, there always exists an optimal integer flow f^*, hence Problem $\text{BND}^{\text{sel}}_{0,\text{sw}}$ polynomially reduces to the constructed minimum-cost flow problem. □

The next question that arises is whether the fixing to zero at the beginning of the time horizon is the reason for which Problem $\text{BND}^{\text{sel}}_{0,\text{sw}}$ can be solved efficiently. In fact, for Problem $\text{BND}^{\text{sel}}_{\text{sw}}$ the reduction to a minimum-cost flow problem does no longer work, as now each switch can also be "on" at the beginning and then be switched off. However, the

problem is still tractable. For the proof of this claim, we need the following variant of the matching problem:

Definition 3.9. *Let $G = (V, E)$ be an undirected graph with edge weights $w : E \to \mathbb{R}$. The* Cardinality-constrained minimum weight matching problem (CMWM) *is given as*

$$\begin{aligned} \min \quad & \sum_{e \in M} w(e) \\ \text{s.t.} \quad & M \subseteq E \text{ is a matching in } G \\ & |M| = k \ . \end{aligned} \tag{CMWM}$$

Here, a matching in $G = (V, E)$ is a subset $M \subseteq E$ such that each vertex of G is adjacent to at most one edge in M. It follows from [24, Cor. 18.10a] that CMWM can be solved in polynomial time using linear programming.

As will be shown soon, $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ with $S_i = 1$ for all $i \in [n]$ polynomially reduces to CMWM, which implies the tractability of $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$. Since this reduction gets less intricate for instances of $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ with a uniform selection bound, the reduction is delayed until after the next lemma.

Lemma 3.10. *The problem $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ polynomially reduces to an instance of $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ with uniform selection bounds. As a result, the selection bounds in $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ may be assumed to be uniform without loss of generality.*

Proof. The proof incrementally reduces a general instance of $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ to the case of a uniform K. As long as there exists an index $\ell \in [T-1]$ with $K^{(\ell)} \neq K^{(\ell+1)}$, we define a new instance of $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ by adding another switch $n+1$. The remainder of the construction depends on the sign of $K^{(\ell)} - K^{(\ell+1)}$. For both cases, let C be large, e.g., $C := \sum_{i=1}^{n} \sum_{t=1}^{T} |c_i^{(t)}| + 1$.

If $K^{(\ell+1)} - K^{(\ell)} > 0$, we define $c_{n+1}^{(t)} := -C$ for $t \leq \ell$ and $c_{n+1}^{(t)} := C$ for $t > \ell$. Moreover, we define new selection bounds $\tilde{K}^{(t)} := K^{(t)} + 1$ for $t \leq \ell$ and $\tilde{K}^{(t)} := K^{(t)}$ for $t > \ell$. The construction guarantees that any optimal solution for the new instance satisfies $x_{n+1}^{(t)} = 1$ if and only if $t \leq \ell$, so that for the original n switches we obtain our original selection bounds. This shows that the two instances are equivalent, up to a constant $-\ell C$ in the objective value. For $K^{(\ell+1)} - K^{(\ell)} < 0$, the construction is symmetric; we then define $c_{n+1}^{(t)} := C$ for $t \leq \ell$ and $c_{n+1}^{(t)} := -C$ for $t > \ell$, set $\tilde{K}^{(t)} := K^{(t)}$ for $t \leq \ell$ and $\tilde{K}^{(t)} := K^{(t)} + 1$ for $t > \ell$, and argue analogously.

In both cases, the number $\sum_{t=1}^{T-1} |K^{(t+1)} - K^{(t)}|$ has decreased by one in our construction. This implies that after at most n^2 steps, we obtain a uniform selection bound. □

Theorem 3.11. *Problem $\mathrm{BND}^{\mathrm{sel}}_{\mathrm{sw}}$ with $S_i = 1$ for all $i \in [n]$ can be solved in polynomial time.*

Proof. We reduce the problem to CMWM. Due to Lemma 3.10, it suffices to prove that an instance of $\mathrm{BND}_{\mathrm{sw}}^{\mathrm{sel}}$ with $S_i = 1$ for all $i \in [n]$ and $K := K^{(t)}$ for all $t \in [T]$ can be polynomially reduced to an instance of CMWM. Thus, given such an instance of $\mathrm{BND}_{\mathrm{sw}}^{\mathrm{sel}}$, we define a graph $G = (V, E)$ by

$$V := \{v_1, \dots, v_n\} \cup \{u_1, \dots, u_n\} ,$$
$$E := \{(v_i, v_j) \colon i, j \in [n], i \neq j\} \cup \{(v_i, u_i) \colon i \in [n]\} .$$

Furthermore, we define weights $w : E \to \mathbb{R}$ for the edges as follows:

$$w(v_i, v_j) := \min_{p \in [T-1]} \Big\{ \sum_{t=1}^{p} c_i^{(t)} + \sum_{t=p+1}^{T} c_j^{(t)}, \sum_{t=1}^{p} c_j^{(t)} + \sum_{t=p+1}^{T} c_i^{(t)} \Big\}$$
$$w(v_i, u_i) := \sum_{t=1}^{T} c_i^{(t)} .$$

Finally, we define $k := K$. The weight $w(v_i, v_j)$ is the optimal value that can be obtained by the two switches i and j if they do not overlap, while $w(v_i, u_i)$ is the value that can be obtained by the switch i alone being "on" over all stages.

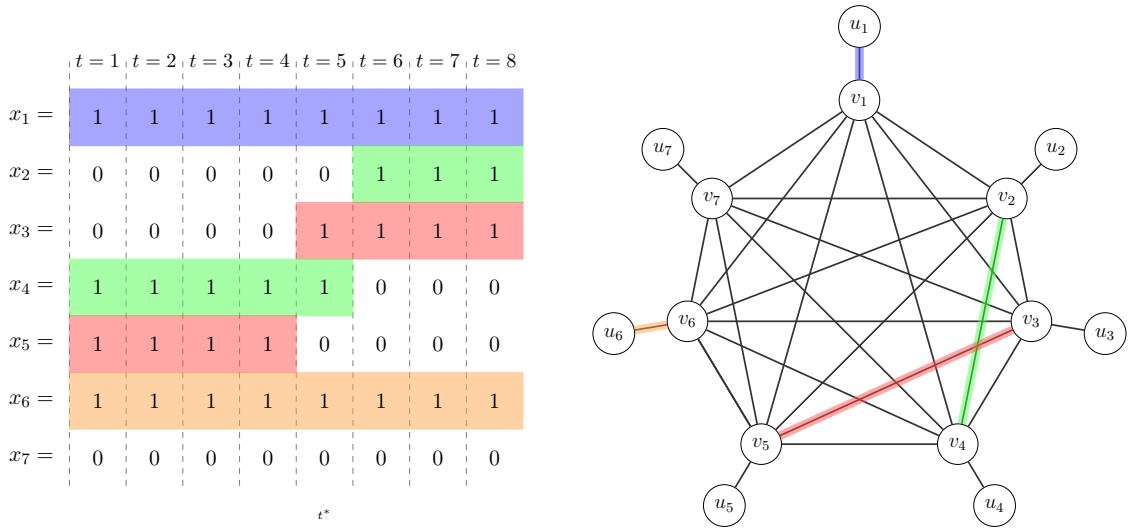
Let x be a feasible solution of $\mathrm{BND}_{\mathrm{sw}}^{\mathrm{sel}}$. Since $S_i = 1$ for all $i \in [n]$, there must exist pairs of non-overlapping switches $D_2 \subseteq [n]^2$ and other switches $D_1 \subseteq [n]$ such that $|D_1| + |D_2| = K$ and all remaining switches are zero throughout the time horizon; here we assume $K \leq n$. In Figure 3a, the pairs in D_2 are illustrated by a common color, while the switches in D_1 have their own color; the number of colors is $K = 4$. We now construct a solution M of CMWM as follows: for $(i, j) \in D_2$, we add edge (v_i, v_j) to M, while for $i \in D_1$, we add edge (v_i, u_i); see Figure 3b. This is a matching in G with $|M| = k$ and its total weight is at most the objective value of x in $\mathrm{BND}_{\mathrm{sw}}^{\mathrm{sel}}$, by definition of w.

Conversely, every solution M of the constructed instance of (CMWM) yields a set of pairs D_2 and a set of switches D_1 with $|D_1| + |D_2| = K$. By choosing optimal non-overlapping switching patterns for each pair in D_2 and choosing all switches in D_1 to be "on" over all stages, we obtain a solution for $\mathrm{BND}_{\mathrm{sw}}^{\mathrm{sel}}$ with objective value $w(M)$. □

We finally show that the uniform selection bound can be relaxed without losing tractability.

Theorem 3.12. *The Problem* $\mathrm{BND}_{\mathrm{sw}}^{\mathrm{sel}}$ *with* $S_i = 1$ *for all* $i \in [n]$ *can be solved in polynomial time.*

We summarize the complexity results we have obtained so far in Table 1.



(a) Solution x and colored sets D_1, D_2. (b) Corresponding matching in G.

Figure 3: Sketch of the reduction of Theorem 3.11 with $n = 7$, $T = 8$, $K = 4$.

$S = 2$, $K = 1$	$S = 1$	n constant	T constant
strongly NP-hard (Theorem 3.4)	polynomial (Theorem 3.12)	polynomial (Theorem 3.5)	polynomial (Theorem 3.6)

Table 1: Summary of complexity results for $\text{BND}^{\text{sel}}_{\text{sw}}$ and $\text{BND}^{\text{sel}}_{0,\text{sw}}$.

3.2 Time-wise variation

Next, we investigate a second type of variation, namely the so-called *time-wise variation*. The resulting problem reads as follows:

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} {c^{(t)}}^{\top} x^{(t)} \\ \text{s.t.} \quad & \sum_{i=1}^{n} |x_i^{(t+1)} - x_i^{(t)}| \leq S_t && \forall t \in [T-1] \\ & \sum_{i=1}^{n} x_i^{(t)} = K^{(t)} && \forall t \in [T] \\ & x^{(t)} \in \{0,1\}^n && \forall t \in [T] \end{aligned} \tag{$\text{BND}^{\text{sel}}_{\text{tw}}$}$$

As before, let $\text{BND}^{\text{sel}}_{0,\text{tw}}$ refer to the special case with fixation to zero at time $t = 0$.

We will show that the above problem reduces to a minimum-cost flow problem and as such it can be solved in polynomial time. For simplicity, we first show the result for the case that all selection bounds $K^{(t)}$ agree and then explain how to adapt the approach to instances

with different bounds.

Consider an instance of $\text{BND}_{\text{tw}}^{\text{sel}}$ where the selection bounds are assumed to be uniform, say K. Then, this selection constraint enforces that the number of switches that are "on" must be equal to K for each stage and, more importantly, it enforces that between two consecutive stages the number of deactivated switches must equal the number of activated switches. 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. The next result shows the tractability of this special case and serves as an auxiliary lemma to derive the tractability of the general case thereafter.

Lemma 3.13. *Problem* $\text{BND}_{\text{tw}}^{\text{sel}}$ *with* $K^{(t)} = K$ *for all* $t \in [T]$ *can be solved in polynomial time.*

Proof. We describe a reduction of $\text{BND}_{\text{tw}}^{\text{sel}}$ to the minimum-cost flow problem. For this, let an instance $I = (c, (S_1, \ldots, S_{T-1}), K)$ of $\text{BND}_{\text{tw}}^{\text{sel}}$ be given. We then construct a directed network (V, A) as follows.

We first introduce a source q with supply $b_q := K$ and a sink r with demand $-b_r = K$; all other nodes will have zero demand or supply. For each switch $i \in [n]$ and each time $t \in [T]$, we add a node $v_i^{(t)}$ and a copy node $\bar{v}_i^{(t)}$ which are connected through an arc $(v_i^{(t)}, \bar{v}_i^{(t)})$; if the flow travels along such an arc, we interpret the respective switch as being turned on at time t.

The source q is connected with each node $v_i^{(1)}$ while each copy $\bar{v}_i^{(T)}$ is connected with the sink r, for $i \in [n]$. We incorporate the variation bound with the help of arcs $(\alpha^{(t)}, \beta^{(t)})$ for $t \in [T-1]$ that have a capacity $\lfloor S_t/2 \rfloor$. We connect all copies $\bar{v}_i^{(t)}$ with $\alpha^{(t)}$ as well as $\beta^{(t)}$ with all nodes $v_i^{(t+1)}$ of the next stage.

Our goal is to model Problem $\text{BND}_{\text{tw}}^{\text{sel}}$ as a (q, r)-flow of value K in the resulting network.

To summarize, the network is defined by the node set

$$V := \{v_i^{(t)}, \bar{v}_i^{(t)} \colon i \in [n],\ t \in [T]\} \cup \{\alpha_t, \beta_t \colon t \in [T-1]\} \cup \{q, r\} ,$$

the arc set

$$\begin{aligned} A := &\{(v_i^{(t)}, \bar{v}_i^{(t)}) \colon i \in [n],\ t \in [T]\} \\ &\cup \{(\bar{v}_i^{(t)}, \alpha^{(t)}), (\beta^{(t)}, v_i^{(t+1)}) \colon i \in [n],\ t \in [T-1]\} \\ &\cup \{(\alpha^{(t)}, \beta^{(t)}) \colon t \in [T-1]\} \\ &\cup \{(q, v_i^{(1)}), (\bar{v}_i^{(T)}, r) \colon i \in [n]\} , \end{aligned}$$

and capacities $u\colon A \to \mathbb{N}$ as follows:

$$u(a) := \begin{cases} 1 & \text{if } a = (v_i^{(t)}, \bar{v}_i^{(t)}) \\ \lfloor S_t/2 \rfloor & \text{if } a = (\alpha^{(t)}, \beta^{(t)}) \\ K & \text{otherwise.} \end{cases}$$

As a final step, we define the costs $c\colon A \to \mathbb{R}$ of the arcs by

$$c(a) := \begin{cases} c_i^{(t)} & \text{if } a = (v_i^{(t)}, \bar{v}_i^{(t)}) \\ 0 & \text{otherwise.} \end{cases}$$

This construction is clearly polynomial; an illustration for an instance with $n = 3$ and $T = 3$ can be found in Figure 4.

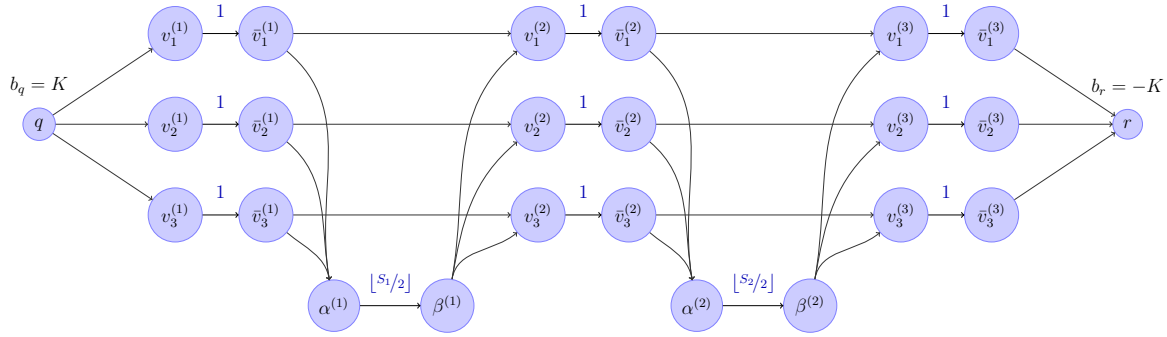


Figure 4: Sketch of the network for $n = 3$, $T = 3$ with arc capacities u different from K.

Since all capacities u in the constructed network are integer, it now suffices to show that the feasible integer flows correspond bijectively to feasible solutions for the given instance of $\text{BND}_{\text{tw}}^{\text{sel}}$, and that the bijection preserves the objective value. Indeed, given a feasible solution x of $\text{BND}_{\text{tw}}^{\text{sel}}$, we can define an integer feasible flow f in (V, A) as follows:

$$\begin{aligned} f_{(v_i^{(t)}, \bar{v}_i^{(t)})} &:= x_i^{(t)}, \quad \forall i \in [n], t \in [T]\,, \qquad (1) \\ f_{(\alpha^{(t)}, \beta^{(t)})} &:= \sum_{i=1}^{n} |x_i^{(t)} - x_i^{(t+1)}|, \quad \forall t \in [T-1]\,. \end{aligned}$$

Using flow conservation, it is easy to verify that this defines a unique feasible flow f in (V, A) with cost

$$\sum_{i=1}^{n} \sum_{t=1}^{T} c(v_i^{(t)}, \bar{v}_i^{(t)}) f_{(v_i^{(t)}, \bar{v}_i^{(t)})} = \sum_{t=1}^{T} {c^{(t)}}^{\top} x^{(t)}\,.$$

Conversely, given a feasible integer flow f in the constructed network, we can use 1 to define a vector $x \in \{0,1\}^{nT}$. It is easy to see that x is then feasible for the given instance of $\text{BND}_{\text{tw}}^{\text{sel}}$ and that its objective value agrees with the objective value of f. □

We now show that the restriction to uniform selection bounds can be relaxed.

Theorem 3.14. *Problem* $\text{BND}^{\text{sel}}_{\text{tw}}$ *can be solved in polynomial time.*

Proof. We reduce the general case to the uniform one addressed in Lemma 3.13 by introducing additional switches that are fixed by the objective function and consume some of the selection capacity. More specifically, for a given instance of $\text{BND}^{\text{sel}}_{\text{tw}}$, let

$$K^+ = \max_{t\in[T]} K^{(t)}, \quad K^- = \min_{t\in[T]} K^{(t)} \ .$$

We introduce $n' := K^+ - K^-$ auxiliary switches $n+1, \ldots, n+n'$ and define $K := K^+$. By setting

$$c^{(t)}_{n+i} := \begin{cases} -C & \text{if } i \leq K^+ - K^{(t)} \\ C & \text{otherwise} \end{cases}$$

for $i \in [n']$, where C is again large enough, we ensure that at time t exactly $K^+ - K^{(t)}$ auxiliary switches are "on", so the remaining capacity is $K^{(t)}$. Finally, by increasing the value of S_t by the number of switchings that are enforced on the auxiliary switches, i.e., by $|K^{(t+1)} - K^{(t)}|$, we obtain an instance of $\text{BND}^{\text{sel}}_{\text{tw}}$ with uniform selection bound K^+ that is equivalent to the original general instance. □

To conclude this section, we note that Problem $\text{BND}^{\text{sel}}_{0,\text{tw}}$ can also be solved efficiently. For this, it suffices to set $K^{(0)} := 0$.

4 Conclusion and future research

This paper presented a detailed study of three Multi-Stage Selection variants. While the problem version with transition cost can be solved by linear programming, the bounded problem versions turned out to be more involved. In particular, we showed that the definition of the variation greatly impacts the complexity of the problem. With time-wise variation, the resulting problem polynomially reduces to a minimum-cost flow problem and, hence, can be solved efficiently. As a by-product, we obtain an extended LP-formulation via a network flow model. In contrast, for switch-wise variation, the problem already becomes strongly NP-hard in a very restrictive setting. Nevertheless, we presented various special cases in which we can show the problem to be efficiently solvable by exploring different methods such as network flows, matchings, and a reduction to classical selection.

An interesting question for future research concerns the complexity of BND in case the selection constraint is an inequality, i.e., the number of selected items is only upper bounded. Except for one, all our results can be transferred to this modified setting with only little extra effort. In essence, we only need to replace the equality in the selection constraint by an upper bound, delete the dummy switches in the proof of Theorem 3.4, and modify the flow network in the proof of Theorem 3.8 by adding an extra arc (α, ω) with capacity $K^{(T)}$

and no cost while removing the lower capacity bounds for the $(v^{(t)}, v^{(t+1)})$ arcs. However, the reduction proving the polynomial solvability of the problem $\text{BND}^{\text{sel}}_{\text{tw}}$ with time-wise variation bounds fails for an upper bound in the selection constraint and there seems to be no obvious way to adapt it to this situation.

Further research could study the complexity of the bounded problem version for other types of variation. For example, one could consider the *total variation*, in which the number of changes over all items and stages is bounded. It is not difficult to develop an efficient dynamic programming scheme for the case that K is not part of the input. If instead the number of stages T is not part of the input, we can solve the problem by enumerating all possibilities to partition the total number of allowed changes to the stages and use the tractability of the time-wise problem version. However, the complexity of the problem is unknown if none of the parameters is fixed. Beyond that, it might be interesting to consider the natural problem variant where cost functions of future stages are uncertain, and address this via robust or stochastic optimization.

References

[1] Evripidis Bampis, Dimitris Christou, Bruno Escoffier, Alexander Kononov, and Kim Thang Nguyen. A simple rounding scheme for multistage optimization. *Theoretical Computer Science*, 907:1–10, 2022.

[2] Evripidis Bampis, Bruno Escoffier, and Alexander Kononov. LP-based algorithms for multistage minimization problems. In *Approximation and Online Algorithms (WAOA 2020)*, pages 1–15, 2021.

[3] Evripidis Bampis, Bruno Escoffier, 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.

[4] Evripidis Bampis, Bruno Escoffier, Kevin Schewior, and Alexandre Teiller. Online multistage subset maximization problems. *Algorithmica*, 83(8):2374–2399, 2021.

[5] Evripidis Bampis, Bruno Escoffier, and Alexandre Teiller. Multistage knapsack. *Journal of Computer and System Sciences*, 126:106–118, 2022.

[6] Claude Berge and Alain Ghouila-Houri. *Programmes, jeux et réseaux de transport*. Dunod, Paris, 1962.

[7] Kateřina Böhmová, Yann Disser, Matúš Mihalák, and Peter Widmayer. Interval selection with machine-dependent intervals. In Frank Dehne, Roberto Solis-Oba, and Jörg-Rüdiger Sack, editors, *Algorithms and Data Structures*, pages 170–181, Berlin, Heidelberg, 2013. Springer.

[8] Robert Bredereck, Till Fluschnik, and Andrzej Kaczmarczyk. When votes change and committees should (not). In *31st International Joint Conference on Artificial Intelligence (IJCAI 2022)*, pages 144–150, 2022.

[9] Christoph Buchheim and Maja Hügging. The polytope of binary sequences with bounded variation. *Discrete Optimization*, 48:100776, 2023.

[10] Markus Chimani and Niklas Troost. Multistage shortest path: Instances and practical evaluation. In *2nd Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2023)*, pages 14:1–14:19, 2023.

[11] Markus Chimani, Niklas Troost, and Tilo Wiedera. Approximating multistage matching problems. *Algorithmica*, 84(8):2135–2153, 2022.

[12] Markus Chimani, Niklas Troost, and Tilo Wiedera. A general approximation for multistage subgraph problems. *Procedia Computer Science*, 223:334–342, 2023.

[13] Till Fluschnik. A multistage view on 2-satisfiability. In *Algorithms and Complexity (CIAC 2021)*, pages 231–244, 2021.

[14] Till Fluschnik and Pascal Kunz. Bipartite temporal graphs and the parameterized complexity of multistage 2-coloring. In *1st Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2022)*, pages 16:1–16:18, 2022.

[15] Till Fluschnik, Rolf Niedermeier, Valentin Rohm, and Philipp Zschoche. Multistage vertex cover. *Theory of Computing Systems*, 66(2):454–483, 2022.

[16] Till Fluschnik, Rolf Niedermeier, Carsten Schubert, and Philipp Zschoche. Multistage s–t path: Confronting similarity with dissimilarity. *Algorithmica*, 85(7):2028–2064, 2023.

[17] Marc Goerigk and Michael Hartisch. *An Introduction to Robust Combinatorial Optimization: Concepts, Models and Algorithms for Decision Making under Uncertainty*. Springer Nature Switzerland, 2024.

[18] 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 (ICALP 2014)*, pages 563–575, 2014.

[19] Adam Kasperski and Paweł Zieliński. Robust recoverable and two-stage selection problems. *Discrete Applied Mathematics*, 233:52–64, December 2017.

[20] Leon Kellerhals, Malte Renken, and Philipp Zschoche. Parameterized algorithms for diverse multistage problems. In *29th Annual European Symposium on Algorithms (ESA 2021)*, pages 55:1–55:17, 2021.

[21] Stefan Lendl, Britta Peis, and Veerle Timmermans. Matroid bases with cardinality constraints on the intersection. *Mathematical Programming*, 194:661–684, 2022.

[22] Daniel Lokshtanov and Amer E. Mouawad. The complexity of independent set reconfiguration on bipartite graphs. *ACM Transactions on Algorithms (TALG)*, 15(1):1–19, 2018.

[23] Alexander Schrijver. *Theory of Linear and Integer Programming*. Wiley, Chichester, 1986.

[24] Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, Berlin, Heidelberg, 2003.