Extending the reflect flow formulation to variable-sized one-dimensional cutting and skiving stock problems

Maxence Delorme[(1)], John Martinovic[(2)]

(1) Department of Econometrics and Operations Research, Tilburg University, The Netherlands
(2) Institute of Numerical Mathematics, Technische Universität Dresden, Germany
Corresponding author `m.delorme@tilburguniversity.edu`

Abstract

Flow formulations have been widely studied for the one-dimensional cutting stock problem and several of its extensions. Among these, the so-called reflect model has shown the best empirical performance when solved directly with a general-purpose integer linear programming solver due to its reduced number of variables and constraints. However, existing adaptations of reflect for the skiving stock problem (SSP), the variable-sized cutting stock problem (VSCSP), and the multiple knapsack problem (MKP) leave room for improvement. In this work, we propose new reflect-specific modeling strategies, both for the case where rolls have variable sizes (as in the VSCSP and MKP) and where the objective is to maximize the number of rolls above a certain length threshold (as in the SSP). In particular, we introduce new pattern representations for variable-sized rolls and a new reduction procedure for the SSP. We also show that these strategies can be extended and even combined when solving the variable-sized skiving stock problem, a combination of the VSCSP and SSP that has never been studied in the literature.

Keywords: One-dimensional cutting and packing problems; Integer Linear Programming; Flow formulations; Computational experiments.

1 Introduction

Over the last decades, one-dimensional Cutting and Packing (C&P) problems have attracted significant interest from the research community. Despite their relatively simple structure, typically centered around a set of capacity constraints and a set of demand constraints, C&P problems have numerous practical applications, including areas as diverse as the paper industry [20], cloud computing [9], and group event seating [33]. Even though many of these problems are known to be $\mathcal{NP}$-hard, this has not stopped the research community from developing highly effective tailored exact algorithms, including "combo" [26], a branch-and-bound (B&B) algorithm for the one-dimensional knapsack problem, "VPSolver" [4], an algorithm that generates compressed flow-based Integer Linear Programming (ILP) models for the multiple-choice vector packing problem that are then solved with a general-purpose ILP solver, and "VRPSolver" [35], a branch-and-price (B&P) algorithm for the vehicle routing problem that can also be used to solve the Cutting Stock Problem (CSP) and other related one-dimensional C&P problems.

Although the landscape of exact algorithms is quite broad, many rely on an initial mathematical formulation of the problem, typically an ILP model, which is then solved via a procedure whose implementation complexity can vary significantly. At one extreme, we find approaches that simply call a general-purpose ILP solver such as Gurobi, CPLEX, or Xpress to solve the ILP model, whereas at the other extreme, we find those in which the model is solved via a tailored, fully "home-made" enumerative algorithm, with various levels of hybridization in between.

Whereas, in theory, there are countless ILP models for a given C&P problem, the research community has mostly been interested in those that exhibit certain positively distinctive features. Let us consider the example of the CSP, in which we are given a set of item types, each characterized by a length and a demand, along with an unlimited supply of identical rolls of fixed length. The objective is to determine the minimum number of rolls required to fulfill the demand for all item types. Over the years, no fewer than six ILP formulations have been proposed for the CSP: textbook, set covering, DP-flow, one-cut, arcflow, and reflect. Sometimes attributed to Kantorovich [21], the textbook model—where a variable represents the number of items of a certain type cut from a given roll—is usually regarded as the simplest one, and it has the particularity of involving a polynomial number of variables and constraints. It is also at the basis of some historical B&B algorithms such as MTP [25]. Although it involves an exponential number of variables, the set covering formulation [17]—where a variable represents the number of times a certain cutting pattern is used—is at the core of some of the most effective B&P algorithms [35, 38]. If the patterns used in the formulation are restricted to the set of proper patterns (i.e., if no pattern exceeding the item demands is generated), then the relaxation of the resulting model, which is known as the proper relaxation [34], is the tightest among all six models. As for the other four models—where a variable represents (what could be seen as an encoding of) the cutting position of an item in the roll—they are all said to be pseudo-polynomial, because the number of variables and constraints they involve is linear in the roll length, which is itself defined by a single number L (or $\log_2 L$ bits) in the input file describing the instance. These formulations have the particularity of exhibiting a continuous relaxation as tight as that of the set covering formulation while also involving a typically lower number of variables and constraints, resulting in models that can often be handled by a general-purpose ILP solver for medium-sized roll lengths.

Among these, DP-flow [5] is the only pseudo-polynomial model whose continuous relaxation is as tight as that of the proper relaxation [13], though it also involves more variables and constraints than the others. Originally introduced by different sets of authors, one-cut [15] and arcflow [36] were shown to be equivalent [13, 31], while also involving a similar number of variables and constraints and exhibiting similar empirical performance when the same reduction procedures are applied. Due to its good trade-off between simplicity and performance [14], the arcflow formulation has become the source of extensive investigations, whether to reduce the number of necessary variables and constraints involved in the model [4, 6], or to develop competitive B&P algorithms [11]. Finally, while it shares many similarities with the arcflow formulation, reflect [13] has the distinguishing feature of containing two distinct types of arcs—reflected and standard—each with its own specific implications in the constraints of the model. To date, reflect is the best-performing ILP formulation when solved directly with a general-purpose ILP solver, which can be explained by the fact that its continuous relaxation is close to that of arcflow while also requiring the fewest variables and constraints among all pseudo-polynomial formulations.

Although tailored B&P algorithms have long achieved state-of-the-art results for the CSP [11, 35, 38], even sparking a healthy competition over which code will outperform the others—a battle that is still ongoing [8]—the search for better pseudo-polynomial formulations remains relevant, as these can easily be extended to related problems such as the parallel-batching processing machine scheduling problem [39], the guillotine two-dimensional knapsack problem [16], or the ordered openend bin packing problem [11], just to name a few. In fact, a recent survey [10] dedicated solely to these pseudo-polynomial formulations highlighted dozens of applications, not only in C&P but also in scheduling, routing, and other areas.

In this work, we show how to improve the performance of the reflect formulation for two specific types of one-dimensional C&P problems: those involving rolls of variable sizes [3], and those where the objective is to construct a maximum number of rolls whose length exceeds a given threshold using a limited supply of items (as opposed to the CSP, where the goal is to cut all items using as few rolls as possible) [27]. While a reflect formulation already exists for these problem types [12, 13, 32], we introduce several modeling strategies that significantly reduce the number of variables and constraints in the model, typically resulting in decreased running times when solved with a general-purpose ILP solver compared to the original formulation. We demonstrate the impact of these strategies on three well-known one-dimensional C&P problems: the Skiving Stock Problem (SSP), the Variable-Sized Cutting Stock Problem (VSCSP), and the Multiple Knapsack Problem (MKP). In addition, we use these strategies to develop effective reflect flow formulations for the Variable-Sized Skiving Stock Problem (VSSSP), a problem that, to the best of our knowledge, has not previously been studied in the literature.

The rest of the manuscript is organized as follows. In Section 2, we introduce the necessary notation and provide a unified problem definition along with a descriptive ILP model, both of which can be used to formulate each of the studied problems. In Section 3, we present a concise summary of existing flow formulations for the CSP. In Section 4, we discuss existing and new modeling strategies for the VSCSP and MKP, and do the same for the SSP in Section 5. In Section 6, we show how these strategies can be combined to develop new flow formulations for the VSSSP. Finally, we empirically demonstrate the performance of the resulting models in Section 7 and we conclude in Section 8.

2 Unified problem definition and descriptive ILP formulation

Using the cutting terminology, we are given in all our problems of interest (i) a set of item types $\mathcal{J}$ and (ii) a set of roll types $\mathcal{R}$. Then, depending on the problem studied, item and roll types are associated with certain features, all of which are displayed in Table 1. For a given item type $j \in \mathcal{J}$, the demand (resp. supply) indicates the minimum (resp. maximum) number of items of type j that must be cut for a solution to be feasible. For a given roll type $r \in \mathcal{R}$, the length (resp. threshold) indicates the maximum (resp. minimum) total length of items that must be cut from a roll of type r considered in the solution for it to be feasible. For each problem, we indicate in the table whether each feature is part of the instance file ("inst.") or the fixed value that it takes in all instances.

Table 1: Summary of item/roll type features and notation used according to the problem studied

Problem	Roll type $r \in \mathcal{R}$				Item type $j \in \mathcal{J}$			
	length $L_r(L^*)$	threshold $H_r(H^*)$	supply $S_r(S^*)$	profit P_r	length l_j	demand b_j	supply s_j	profit p_j
CSP	inst.	0	∞	-1	inst.	inst.	∞	0
SSP	∞	inst.	∞	1	inst.	0	inst.	0
VSCSP	inst.	0	inst.	inst.**	inst.	inst.	∞	0
MKP	inst.	0	inst.	0	inst.	0	inst.	inst.
VSSSP	∞	inst.	inst.	inst.	inst.	0	inst.	0

*We omit the index in the notation for the CSP and the SSP as there is only one roll type

**If these values are expressed as costs, as is typically done in literature, a negative sign should be added to convert each cost into a (negative) profit

For example in the CSP, the (unique) roll type has its length specified in the instance file, while its threshold is always zero and its supply is unlimited. The profit associated with using a roll is -1, which is equivalent to minimizing the number of rolls used. Regarding the item types, their length and demand are defined in the instance file, whereas their supply is unlimited and their associated profit is zero. In the VSSSP, each roll type has a given threshold, supply, and profit specified in the instance file, while its length is unlimited. As for the item types, their length and supply are defined in the instance file, whereas their demand and associated profit are both zero. In the following, we assume that $L_r, H_r, S_r \in \mathbb{N}_0$ ($r \in \mathcal{R}$) and $l_j, b_j, s_j \in \mathbb{N}_0$ ($j \in \mathcal{J}$).

We now provide a descriptive ILP formulation that models all five problems when the parameter values from Table 1 are plugged in. This model assumes an upper bound S_r on the supply of each roll type $r \in \mathcal{R}$, which can be set, for example, to $\sum_{j\in\mathcal{J}} b_j$ for the CSP or $\lfloor \frac{\sum_{j\in\mathcal{J}} s_j l_j}{H} \rfloor$ for the SSP. Given such an upper bound, we define the set $\mathcal{S}_r = \{1, 2, \ldots, S_r\}$ for each roll type $r \in \mathcal{R}$, representing all possible copies of the roll of type r. Using a set of binary decision variables y_{rs} taking the value 1 if the s^{th} copy of the roll of type r is used in the solution ($r \in \mathcal{R}, s \in \mathcal{S}_r$), and a set of integer decision variables x_{rsj} indicating the number of times an item of type j is cut from the s^{th} copy of the roll of type r ($r \in \mathcal{R}, s \in \mathcal{S}_r, j \in \mathcal{J}$), the model is defined as follows:

$$\max \quad \sum_{r\in\mathcal{R}} \sum_{s\in\mathcal{S}_r} P_r y_{rs} + \sum_{r\in\mathcal{R}} \sum_{s\in\mathcal{S}_r} \sum_{j\in\mathcal{J}} p_j x_{rsj} \tag{1}$$

$$\text{s.t.} \quad H_r y_{rs} \leq \sum_{j\in\mathcal{J}} l_j x_{rsj} \leq L_r y_{rs} \qquad r \in \mathcal{R}, s \in \mathcal{S}_r, \tag{2}$$

$$b_j \leq \sum_{r\in\mathcal{R}} \sum_{s\in\mathcal{S}_r} x_{rsj} \leq s_j \qquad j \in \mathcal{J}, \tag{3}$$

$$y_{rs} \in \{0, 1\} \qquad r \in \mathcal{R}, s \in \mathcal{S}_r, \tag{4}$$

$$x_{rsj} \in \mathbb{N}_0 \qquad r \in \mathcal{R}, s \in \mathcal{S}_r, j \in \mathcal{J}. \tag{5}$$

The objective function (1) maximizes the total profit, whether it comes from the items or the rolls cut. Constraints (2) ensure that, for each copy of each roll type considered in the solution, the total length of items cut from the copy is always at least as large as the threshold of the roll type, while never exceeding its length. Constraints (3) ensure that, for each item type, the number of items cut from the rolls is no less than the demand and no more than the supply.

3 Existing flow formulations for the CSP

In this section, we provide a concise summary of the arcflow and reflect formulations for the CSP. As underlined in the literature [1], an algorithm using a flow-based ILP formulation is typically composed of three phases: (i) a graph construction phase in which an instance I of the original problem is transformed into an instance I' of a network flow problem; (ii) a solving phase, in which the instance I' is solved (e.g., with an ILP solver); and (iii) a mapping phase, in which the solution of instance I' is mapped back to a solution of instance I. In this work, we mostly focus on the first two phases, as the last one amounts to a flow-decomposition algorithm (though not necessarily a trivial one, especially for reflect-based formulations).

In the following, an arc a is either defined as a triplet (d, e, j) or a quadruplet (d, e, j, r). The first element, d, always refers to the tail of the arc, denoted by $t(a)$. The second element, e, always refers to the head of the arc, denoted by $h(a)$. The third element, j, either refers to the item type modeled by the arc (when $j \geq 1$) or to a model-specific information (typically labeled by some $j \leq 0$). This is denoted by $j(a)$. The fourth element (when present), r, always refers to the roll type modeled by the arc (if any), denoted by $r(a)$.

3.1 Arcflow formulation for the CSP

In the arcflow formulation, a roll is represented by a path. In the graph construction phase, we consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where vertex set $\mathcal{V} = \{0, 1, \dots, L\}$ includes a vertex for each unit of roll length, and where the arc set $\mathcal{A}$ is composed of:

- *item arcs*, where each arc is identified by a triplet $(d, d + l_j, j)$ modeling the fact that one item of type j is cut from position d to position $d + l_j$ in a roll;
- *loss arcs*, where each arc is identified by a triplet $(d, e, 0)$ modeling the fact that trim loss is positioned from position d to position e in a roll;

In this work, we use Algorithm A.2 in Section A.1 of the appendix to generate $\mathcal{V}$ and $\mathcal{A}$, which includes the symmetry reduction procedures originally described by Valério de Carvalho [36].

In the model considered in the solving phase, we use integer decision variables f_a to represent the amount of flow going through arc $a \in \mathcal{A}$ in the solution. The arcflow formulation is as follows:

$$\min \quad \sum_{\substack{a \in \mathcal{A} \\ t(a)=0}} f_a \tag{6}$$

$$\text{s.t.} \quad \sum_{\substack{a \in \mathcal{A} \\ h(a)=v}} f_a = \sum_{\substack{a \in \mathcal{A} \\ t(a)=v}} f_a \qquad v \in \mathcal{V} \setminus \{0, L\}, \tag{7}$$

$$\sum_{\substack{a \in \mathcal{A} \\ t(a)=0}} f_a = \sum_{\substack{a \in \mathcal{A} \\ h(a)=L}} f_a, \tag{8}$$

$$\sum_{\substack{a \in \mathcal{A} \\ j(a)=j}} f_a \geq b_j \qquad j \in \mathcal{J}, \tag{9}$$

$$f_a \in \mathbb{N}_0 \qquad a \in \mathcal{A}. \tag{10}$$

Objective function (6) minimizes the amount of flow leaving node 0, which is equivalent to the number of rolls used in the solution. Flow conservation constraints (7) ensure that the amount of flow entering an intermediate node is equal to the amount of flow leaving that node. Constraint (8) ensures that the amount of flow leaving node 0 is equal to the amount of flow entering L. Finally, constraints (9) ensure that the demand for each item type is fulfilled.

3.2 Reflect formulation for the CSP

In the reflect formulation, a roll is represented by a pair of colliding subpaths, one of which contains a reflected arc. In the graph construction phase, we consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where vertex set $\mathcal{V} = \{0, 1, \dots, \frac{L}{2}\}$ includes a vertex for each unit up to half of the roll length, and where arc set $\mathcal{A}$ is composed of:

- *standard item arcs*, where each arc is identified by a quadruplet $(d, d + l_j, j, 0)$ modeling the fact that one item of type j is either cut from position d to position $d + l_j$ or from position $L - (d + l_j)$ to position $L - d$;

- *reflected item arcs*, where each arc is identified by a quadruplet $(d, L-(d+l_j), j, 1)$ modeling the fact that an item of type j is cut from position d to position $d+l_j$ in a roll;
- *standard loss arcs*, where each arc is identified by a quadruplet $(d, e, 0, 0)$ modeling the fact that trim loss is either positioned from position d to position e or from position $L-e$ to position $L-d$ in a roll;
- *reflected connection arc* $(\frac{L}{2}, \frac{L}{2}, 0, 1)$, which only serves for modeling purpose to link two subpaths colliding in node $\frac{L}{2}$.

In this work, we use Algorithm A.3 in Section A.2 of the appendix to generate $\mathcal{V}$ and $\mathcal{A}$, which was initially proposed by Delorme and Iori [13]. It is worth noting that this algorithm includes a reduction procedure (Step 12) that stems from the fact that there always exists an optimal solution for the reflect formulation that does not contain any reflected item arc $(d, L-(d+l_j), j, 1)$ such that $d > L-(d+l_j)$ [13].

Reusing integer decision variables f_a to represent the amount of flow going through arc $a \in \mathcal{A}$ in the solution, the reflect formulation is as follows:

$$\min \sum_{\substack{a\in\mathcal{A}\\ r(a)=1}} f_a \tag{11}$$

$$\text{s.t.} \sum_{\substack{a\in\mathcal{A}\\ h(a)=v,\, r(a)=0}} f_a = \sum_{\substack{a\in\mathcal{A}\\ t(a)=v}} f_a + \sum_{\substack{a\in\mathcal{A}\\ h(a)=v,\, r(a)=1}} f_a \qquad v \in \mathcal{V}\setminus\{0\}, \tag{12}$$

$$\sum_{\substack{a\in\mathcal{A}\\ t(a)=0}} f_a + \sum_{\substack{a\in\mathcal{A}\\ h(a)=0}} f_a = 2\sum_{\substack{a\in\mathcal{A}\\ r(a)=1}} f_a, \tag{13}$$

$$\sum_{\substack{a\in\mathcal{A}\\ j(a)=j}} f_a \geq b_j \qquad j \in \mathcal{J}, \tag{14}$$

$$f_a \in \mathbb{N}_0 \qquad a \in \mathcal{A}. \tag{15}$$

Objective function (11) minimizes the amount of flow circulating through reflected arcs, which corresponds to the number of rolls used in the solution. Constraints (12) ensure that the amount of flow entering any node (besides 0) through a standard arc is equal to the amount of flow leaving that node through any kind of arc plus the amount of flow entering that node through a reflected arc. Constraint (13) ensures that two subpaths originate from node 0 for each roll considered in the solution, except when a reflected arc enters node 0—in which case, the subpath is considered self-sufficient and does not need to be completed by another subpath. The latter happens if and only if $l_j = L$ holds for some $j \in \mathcal{J}$, otherwise the second sum in constraint (13) is empty. Finally, constraints (14) ensure that the demand for each item type is fulfilled. An example illustrating the arcflow and reflect graph constructions is provided in Section A.3 of the appendix.

4 Existing and reduced flow formulations for the VSCSP and MKP

Both the arcflow and reflect formulations have been extended to tackle the VSCSP [13, 37] and the MKP [12]. In this section, we first briefly describe the modifications these extensions require compared to the previously described graph construction algorithms and ILP models, and then introduce our new modeling techniques to reduce the size of the adapted reflect formulations.

4.1 Existing arcflow and reflect formulations for the VSCSP and MKP

In the following, we consider that the roll types are preliminarily sorted in decreasing order of their length (i.e., $L_1 = \max_{r \in \mathcal{R}}\{L_r\}$). Intuitively, the presence of multiple roll sizes in flow formulations is typically handled by considering a single roll type with length L_1, along with $|\mathcal{R}|$ additional dummy item types with length $L_1 - L_r$ ($r \in \mathcal{R}$). One must then ensure that each roll representation—whether it is a path or a pair of subpaths colliding at the same vertex—includes exactly one of these dummy items that encode the roll type.

In the arcflow formulation, this is implemented as follows. At first, a new target node T representing the end of a roll is created. The arcs in the arc set $\mathcal{A}$ are now represented as quadruplets, with the fourth element in each arc indicating the roll type with which it is associated. Besides the item arcs—now of the form $(d, d + l_j, j, 0)$—and the loss arcs—now of the form $(d, e, 0, 0)$—there are *roll arcs*, where each arc $(L_r, T, 0, r)$ models the usage of a roll of type r. Only minor modifications to Algorithm A.2 are required for the graph creation: in addition to appending a fourth element to the arcs in Steps 7, 8, and 11, all roll lengths must be added to vertex set $\mathcal{V}$ in Step 9, and the roll arcs must be generated. An ILP model valid for both the VSCSP and the MKP once the values from Table 1 are plugged in is described in Section B.1 of the appendix.

In the reflect formulation, it is not as easy to ensure that a roll representation (here, a pair of colliding subpaths) includes exactly one of these dummy items: if one applies the same strategy as in the arcflow formulation and creates standard roll arcs, then it becomes possible to obtain pairs of colliding subpaths in which both subpaths contain such an arc, and preventing this situation from occurring in the solution would appear to require an impractically large number of constraints. To avoid this issue, Delorme and Iori [13] suggested including these dummy items within the reflected (item or connection) arcs, as the model enforces that only one such arc is selected per pair of colliding subpaths. The resulting graph can be constructed using Algorithm 1.

Algorithm 1 Graph construction for the reflect model for the VSCSP and MKP

Input: $\mathcal{J}$ and $\mathcal{R}$

$\mathcal{V} \leftarrow \{0\}$; $\mathcal{A} \leftarrow \emptyset$ ▷ initialize a graph with only vertex 0
$\mathcal{T} \leftarrow \{0\}$ ▷ keeps track of the possible tails
for $j \in \mathcal{J}$ **do** ▷ for every item type
  for $k = 1, \dots, b_j$ **do** ▷ for every unit of demand
    $\mathcal{T}' \leftarrow \mathcal{T}$ ▷ make a copy of the possible tails
    for $d \in \mathcal{T}'$ **do** ▷ for every possible tail
      if $d + l_j \leq \frac{L_1}{2}$ **then** ▷ if this is a standard item arc
        $\mathcal{V} \leftarrow \mathcal{V} \cup \{d + l_j\}$ ▷ add the head to the original vertex set
        $\mathcal{T} \leftarrow \mathcal{T} \cup \{d + l_j\}$ ▷ add the head to the possible tails
        $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, d + l_j, j, 0)\}$ ▷ create the standard item arc
      for $r \in \mathcal{R}$ **do** ▷ for each roll type
        if $d < \frac{L_r}{2}$ and $d + l_j > \frac{L_r}{2}$ **then** ▷ if this is a potential reflected item arc
          if $d \leq L_r - (d + l_j)$ **then** ▷ and it is not eliminated by the reduction procedure
            $\mathcal{V} \leftarrow \mathcal{V} \cup \{L_r - (d + l_j)\}$ ▷ add the head to the original vertex set
            $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, L_r - (d + l_j), j, r)\}$ ▷ create the reflected item arc
for $r \in \mathcal{R}$ **do** ▷ for each roll type
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{\frac{L_r}{2}\}$ ▷ add $\frac{L_r}{2}$ to the vertex set as it may have not been done before
  $\mathcal{A} \leftarrow \mathcal{A} \cup \{(\frac{L_r}{2}, \frac{L_r}{2}, 0, r)\}$ ▷ create the reflected connection arc
for $d \in \mathcal{V} \setminus \{\frac{L_1}{2}\}$ **do** ▷ for every vertex in the set besides $\frac{L_1}{2}$
  $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, e, 0, 0)\}$ where $e = \min\{v \in \mathcal{V} : v > d\}\}$ ▷ create the standard loss arc

Output: $\mathcal{G} = (\mathcal{V}, \mathcal{A})$

The associated ILP model (A.36)-(A.41), which is also valid for both the VSCSP and the MKP once the values from Table 1 are plugged in, is described in Section B.2 in the appendix. As a final note on existing flow formulations for the VSCSP and MKP, we mention the inclusion of additional dummy variables to model the number of times an item type or a bin type is included in the solution. Delorme and Iori [13] used dummy variables $\omega_r = \sum_{\substack{a \in \mathcal{A} \\ r(a)=r}} f_a$ $(r \in \mathcal{R})$ in their reflect adaptation for the VSCSP, whereas Dell'Amico et al. [12] used dummy variables $t_j \leq \sum_{\substack{a \in \mathcal{A} \\ j(a)=j}} f_a$ $(j \in \mathcal{J})$ in both their arcflow and reflect adaptations for the MKP. When both sets of dummy variables are used, the objective function becomes

$$\max \sum_{r \in \mathcal{R}} P_r \omega_r + \sum_{j \in \mathcal{J}} p_j t_j. \tag{16}$$

Whereas these dummy variables do not appear to serve any purpose from a modeling point of view, we confirm in our computational experiments their necessity in flow models whose objective function involves many arcs, and their dispensability in flow models whose objective function involves only a few.

4.2 Alternative pattern representations in the reflect formulation

The main issue with Algorithm 1 is that it may create a large number of reflected item arcs: $\mathcal{O}(l_j|\mathcal{R}|)$ for a given item type $j \in \mathcal{J}$. In fact, for some MKP instances with many roll types, there are more arcs in the associated reflect graph than in the associated arcflow graph (although the number of nodes is still greatly reduced). In the following, we describe three ways to significantly reduce the total number of reflected item arcs required. This is achieved through the introduction of three new types of arcs—each in a much smaller quantity, so that the trade-off remains favorable: (i) small-roll reflected arcs, (ii) extended reflected connection arcs, and (iii) downgrade arcs.

Small-roll reflected arcs The first strategy replaces all reflected item arcs $(d, L_r - (d + l_j), j, r)$ associated with a roll $r \in \mathcal{R}$ such that $L_r \leq \frac{L_1}{2}$ by what we refer to as a *small-roll (reflected) arc* of the form $(0, L_r, 0, r)$. As per the ILP model described in Section B.2 of the appendix, if selected, such an arc must be completed by a standard 0-L_r subpath composed of standard item and loss arcs, which are already created in Step 10 of Algorithm 1. Hence, for each roll r $(r \in \mathcal{R} : L_r \leq \frac{L_1}{2})$, we remove $\mathcal{O}(l_j)$ reflected item arcs per item type $j \in \mathcal{J}$ at the cost of a single new small-roll reflected arc, a very favorable trade-off.

Extended reflected connection arcs The second strategy replaces all reflected item arcs $(d, L_r - (d + l_j), j, r)$ whose corresponding standard item arc $(d, d + l_j, j, 0)$ is created in Step 10 of Algorithm 1 with what we refer to as an *extended reflected connection arc* of the form $(d + l_j, L_r - (d + l_j), 0, r)$. Indeed, observe that taking successively $(d, d + l_j, j, 0)$ and $(d + l_j, L_r - (d + l_j), 0, r)$ is the same as taking $(d, L_r - (d + l_j), j, r)$ directly. Intuitively, we "split" some reflected item arcs into a standard item arc and an extended reflected connection arc. Observe that the latter is no longer associated with an item type and can therefore be used to split multiple reflected item arcs. In practice, the same extended reflected connection arc, say $(d, L_r - d, 0, r)$, can be used to split all reflected item arcs $(d', L_r - (d' + l_{j'}), j', r)$ such that $d' + l_{j'} = d$. In other words, one extended reflected connection arc replaces between 1 and $|\mathcal{J}|$ reflected item arcs. Regarding the name, it comes from the observation that the previously defined reflected connection arc $(\frac{L_r}{2}, \frac{L_r}{2}, 0, r)$ is a special case of extended reflected connection arc $(d, L_r - d, 0, r)$, where $d = \frac{L_r}{2}$.

Downgrade arcs The third strategy replaces all reflected item arcs $(d, L_r - (d + l_j), j, r)$ where $r \geq 2$ whose corresponding standard item arc $(d, d+l_j, j, 0)$ is not created in Step 10 of Algorithm 1—meaning that reflected item arc $(d, L_1 - (d + l_j), j, 1)$ is created instead—with what we refer to as a *downgrade arc* of the form $(L_1 - (d+l_j), L_r - (d+l_j), -1, r)$. Indeed, observe that taking successively $(d, L_1 - (d+l_j), j, 1)$ and $(L_1 - (d+l_j), L_r - (d+l_j), -1, r)$ is the same as taking $(d, L_r - (d+l_j), j, r)$ directly, if not for the extra roll of type 1 considered in the first arc. Intuitively, we replace some reflected item arcs for roll type $r \geq 2$ by their corresponding reflected item arc for roll type 1, which we "downgrade" afterwards to roll type r. As with the extended reflected connection arcs, observe that a downgrade arc is not associated with an item type and can therefore be used to replace between 1 and $|\mathcal{J}|$ reflected item arcs. Whereas neither the small-roll reflected arcs nor the extended reflected connection arcs require changes to model (A.36)-(A.41) presented in Section B.2 of the appendix, the presence of downgrade arcs requires an update in the mathematical expressions used in objective function (A.36) and constraints (A.40) to count the number of rolls of type 1 cut in the solution, which should be computed as:

$$\sum_{\substack{a \in \mathcal{A} \\ r(a)=1}} f_a - \sum_{\substack{a \in \mathcal{A} \\ j(a)=-1}} f_a$$

so as to account for the downgrade arcs, as well as an update in flow constraints (A.37):

$$\sum_{\substack{a \in \mathcal{A} \\ h(a)=v, r(a)=0}} f_a = \sum_{\substack{a \in \mathcal{A} \\ t(a)=v \\ j(a)\neq -1}} f_a + \sum_{\substack{a \in \mathcal{A} \\ h(a)=v, r(a)\geq 1 \\ j(a)\neq -1}} f_a + \sum_{\substack{a \in \mathcal{A} \\ h(a)=v \\ j(a)=-1}} f_a - \sum_{\substack{a \in \mathcal{A} \\ t(a)=v \\ j(a)=-1}} f_a \qquad v \in \mathcal{V} \setminus \{0\}. \tag{17}$$

The first two elements of the right-hand side are updated to exclude the downgrade arcs, with those being handled by the last two newly added elements (note that no update is needed on the left-hand side, as all downgrade arcs have $r(a) > 0$). Intuitively, a downgrade arc can be seen as an elongation of a reflected item arc. Hence, a downgrade arc a removes one unit of flow coming from a reflected arc from $t(a)$ (fourth element) and transfers it to $h(a)$ (third element). Finally, an additional set of constraints is needed to ensure that every downgrade arc is preceded by a reflected item arc for roll type 1:

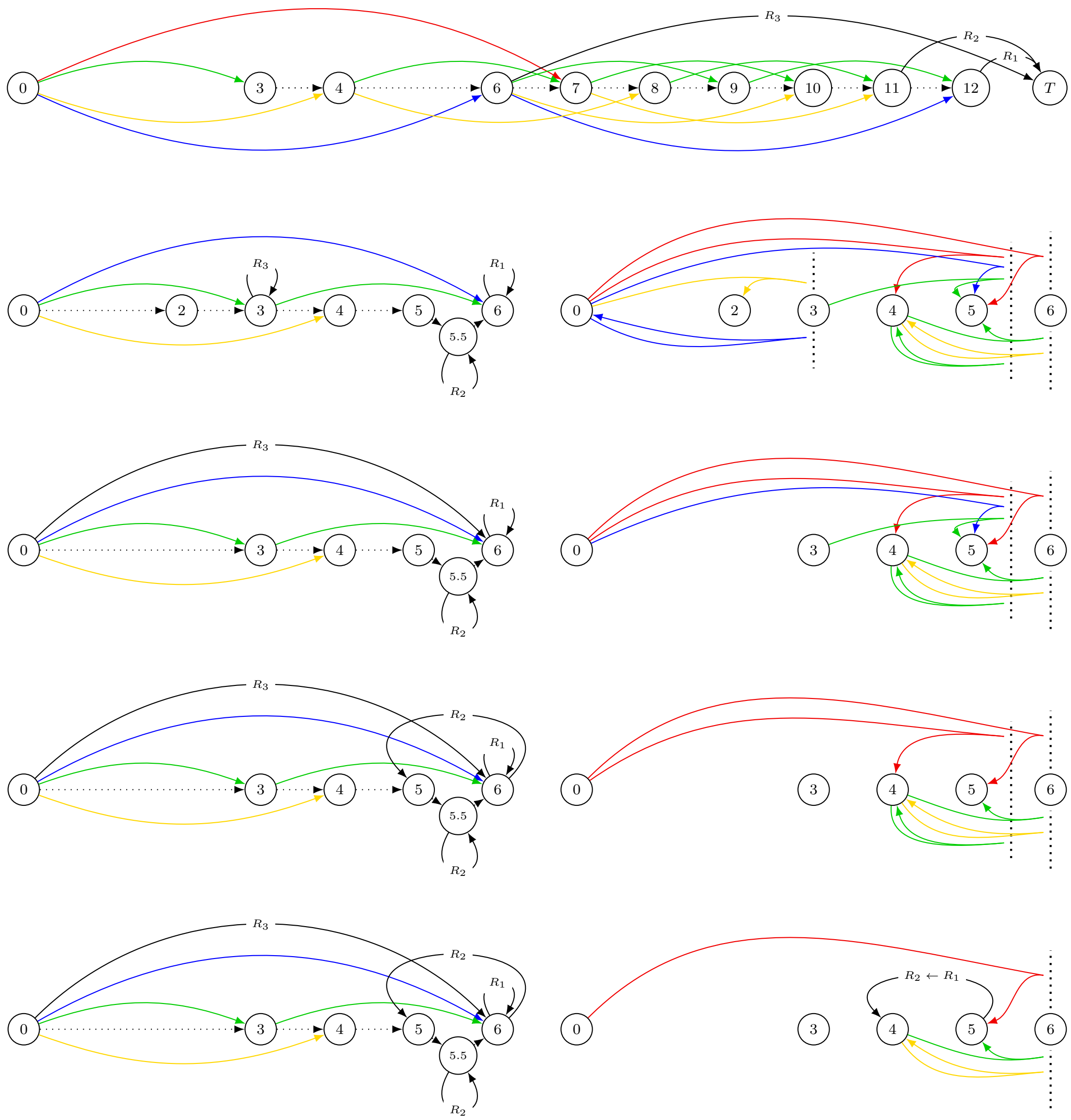
$$\sum_{\substack{a \in \mathcal{A} \\ h(a)=v, r(a)=1}} f_a \geq \sum_{\substack{a \in \mathcal{A} \\ t(a)=v, j(a)=-1}} f_a \qquad v \in \mathcal{V} \setminus \{0\}. \tag{18}$$

The reflect graph using the three newly introduced arc types can be constructed using Algorithm A.4 in Section B.3 of the appendix.

4.3 Illustrative example

Consider a VSCSP instance composed of four item types with lengths $l_1 = 7$, $l_2 = 6$, $l_3 = 4$, and $l_4 = 3$ and corresponding demands $b_1 = 1$, $b_2 = 2$, $b_3 = 2$, and $b_4 = 2$, with three roll types with lengths $L_1 = 12$, $L_2 = 11$, and $L_3 = 6$. The corresponding arcflow graph obtained after applying the adaptation of Algorithm A.2 is shown at the top of Figure 1. The four graphs below are those obtained after applying adapted versions of Algorithms 1 and A.4 to produce the graphs for reflect, reflect with small-roll arcs, reflect with small-roll and extended reflected connection arcs, and reflect with small-roll, extended reflected connection, and downgrade arcs. Item arcs are displayed as colored solid lines—red, blue, yellow, and green for item type 1, 2, 3, and 4, respectively. Loss arcs are shown as black dotted lines, while the remaining arcs (roll for arcflow and small-roll, extended reflected connection, and downgrade for reflect) are represented as black solid lines, with their corresponding roll type labeled at the midpoint of the arc. To improve visibility, each reflected graph is divided into two subgraphs: the one on the right contains all the reflected items arcs and downgrade arcs, while the one on the left contains the remaining arcs.

Figure 1: Arcflow and each of the four reflect graphs corresponding to the example



In this example, the arcflow graph contains 24 arcs and 11 nodes whereas the reflect graph contains only 22 arcs and 7 nodes. When using the small-roll arcs, the resulting graph has 19 arcs and 6 nodes: indeed, the two reflected item arcs $(0, 2, 3, 3)$ and $(0, 0, 2, 3)$ disappear, the reflected connection arc $(3, 3, 0, 3)$ is replaced by the small-roll arc $(0, 6, 0, 3)$, and the loss arcs $(0, 2, 0, 0)$ and $(2, 3, 0, 0)$ are merged into $(0, 3, 0, 0)$ as vertex 2 no longer needs to be reached, having no incoming or outgoing arcs. For example, a roll of type 3 that contains only an item of type 3 is represented by path $(0, 4, 3, 0), (4, 6, 0, 0), (6, T, 0, 3)$ in arcflow, by subpaths $(0, 2, 3, 3)$ and $(0, 2, 0, 0)$ colliding at node 2 in reflect, and by subpaths $(0, 6, 0, 3)$ and $(0, 4, 3, 0)$, $(4, 5, 0, 0)$, $(5, \frac{11}{2}, 0, 0)$, $(\frac{11}{2}, 6, 0, 0)$ colliding at node 6 in reflect with small-roll arcs.

When also using the extended reflected connection arcs, the resulting graph has 18 arcs and 6 nodes as the two reflected item arcs $(0, 5, 2, 2)$ and $(3, 5, 4, 2)$ are now replaced by the extended reflected connection arc $(6, 5, 0, 2)$. For example, a roll of type 2 that contains one item of type 2 and one item of type 3 is represented by path $(0, 6, 2, 0), (6, 10, 3, 0), (10, 11, 0, 0), (11, T, 0, 2)$ in arcflow, by subpaths $(0, 5, 2, 2)$ and $(0, 4, 3, 0)$, $(4, 5, 0, 0)$ colliding at node 5 in reflect with small-roll arcs only, and by subpaths $(0, 6, 2, 0)$, $(6, 5, 0, 2)$ and $(0, 4, 3, 0)$, $(4, 5, 0, 0)$ colliding at node 5 in reflect with small-roll and extended reflected connection arcs.

Finally, when also using the downgrade arcs, the resulting graph has 17 arcs and 6 nodes, as the two reflected item arcs $(0, 4, 1, 2)$ and $(4, 4, 4, 2)$ are now replaced by the downgrade arc $(5, 4, -1, 2)$. For example, a roll of type 2 that contains two items of type 3 and one item of type 4 is represented by path $(0, 4, 3, 0), (4, 8, 3, 0), (8, 11, 4, 0), (11, T, 0, 2)$ in arcflow, by subpaths $(0, 4, 3, 0)$, $(4, 4, 4, 2)$ and $(0, 4, 3, 0)$ colliding at node 4 in reflect with small-roll and extended reflected connection arcs only, and by subpaths $(0, 4, 3, 0)$, $(4, 5, 4, 1)$, $(5, 4, -1, 2)$ and $(0, 4, 3, 0)$ colliding at node 4 in reflect with small-roll, extended reflected connection, and downgrade arcs.

5 Existing and new flow formulations for the SSP

Both the arcflow and reflect formulations have also been extended to tackle the SSP [27, 32]. As in the previous section, we first briefly summarize the key ideas behind these extensions, and then introduce our new reflect-specific alternative modeling techniques. To highlight the similarities in flow formulations for the CSP and the SSP, we will use L to denote the roll threshold from now on, instead of H as defined in Section 2.

5.1 Existing arcflow and reflect formulations for the SSP

In the following, we assume that the item types are preliminarily sorted in decreasing order of their length (i.e., $l_1 = \max_{j\in\mathcal{J}}\{l_j\}$). We also assume that $l_j \leq L$ $(j \in \mathcal{J})$—if this is not the case, then l_j is set to L. The most intuitive adaptation of the arcflow formulation to the SSP simply removes all loss arcs, while also allowing rolls of any length L' such that $L \leq L' \leq L + l_1 - 1$. As for the necessary updates in Algorithm A.2, besides removing Steps 9-11, the if-condition in Step 6 becomes $d + l_j \leq L + l_1 - 1$. The associated ILP model [27] is as follows:

$$\max \quad \sum_{\substack{a\in\mathcal{A}\\ t(a)=0}} f_a \tag{19}$$

$$\text{s.t.} \quad \sum_{\substack{a\in\mathcal{A}\\ h(a)=v}} f_a = \sum_{\substack{a\in\mathcal{A}\\ t(a)=v}} f_a \qquad v \in \mathcal{V} : 0 < v < L, \tag{20}$$

$$\sum_{\substack{a\in\mathcal{A}\\ j(a)=j}} f_a \leq b_j \qquad j \in \mathcal{J}, \tag{21}$$

$$f_a \in \mathbb{N}_0 \qquad a \in \mathcal{A}, \tag{22}$$

which is an adaptation of model (6)-(10) that accounts for the fact that $\mathcal{V}$ now includes several sink nodes v such that $v \geq L$, while also incorporating the necessary updates to the objective function (which now maximizes the amount of flow leaving node 0) and to the demand constraints (which now limit the supply of each item type).

Another arcflow adaptation was proposed by Martinovic et al. [32], who suggested that all item arcs $(d, d+l_j, j)$ for a given item type $j \in \mathcal{J}$ such that $d+l_j \geq L$ could be conveniently merged into a unique item arc $(L-l_j, L, j)$, provided that the appropriate set of (what they referred to as) *reversed loss arcs* was introduced. A reversed loss arc $(e, d, 0)$, where $e, d \in \mathcal{V}$ and $e > d$, models the fact that the sum of item lengths cut from the roll exceeds L by an additional $e - d$ units ("additional" because multiple reversed loss arcs may be included in the same subpath).

As observed in the previous section, allowing for multiple roll lengths is a significant burden on the reflect formulation. Hence, the direction followed by Martinovic et al. [32] when adapting the reflect formulation to the SSP was to fix the roll length to the threshold L and introduce reversed loss arcs. From the modeling point of view, these reversed loss arcs could either be integrated before the reflected arc (i.e., into the standard subpath), leading to what we refer to as *backward loss arcs*, or after the reflected arc, leading to what could be seen as "reversed reversed loss arcs", or more simply, *forward loss arcs*, but with the constraint that such arcs must follow either other forward loss arcs or reflected item arcs. Having observed that some patterns contain only reflected arcs, the authors opted for the SSP reflect formulation with forward loss arcs, of the form $(e, d, -1, 0)$—to follow the unified notation adopted in this work. Note that the resulting model also requires the reflected connection arc $(\frac{L}{2}, \frac{L}{2}, 0, 1)$ to be allowed to take a negative value. For example, consider an instance with two items of length 6 and a roll with threshold 10: a solution includes two reflected item arcs $(0, 4, 1, 1)$, each followed by a forward loss arc $(4, 5, -1, 0)$. A negative flow on the reflective connection arc $(\frac{L}{2}, \frac{L}{2}, 0, 1)$ then transforms one of these reflected subpaths into a standard subpath, resulting in the desired pair of colliding subpaths, one of which with a reflected arc. The algorithm used to create the reflect graph with forward loss arcs for the SSP is almost the same as Algorithm A.3 described in Section A.2 the appendix, except for Step 12 that should be removed, as Martinovic et al. [32] demonstrated that the reduction procedure does not hold for the SSP (e.g., consider an instance with three items of length 9 and a roll with threshold 20: an optimal solution must use reflected item arc $(9, 2, 1, 1)$ whose tail, 9, is strictly greater than its head, 2). The associated ILP model [32] is:

$$\max \sum_{\substack{a \in \mathcal{A} \\ r(a)=1}} f_a \tag{23}$$

$$\text{s.t.} \sum_{\substack{a \in \mathcal{A} \\ h(a)=v, r(a)=0 \\ j(a)\neq -1}} f_a = \sum_{\substack{a \in \mathcal{A} \\ t(a)=v \\ j(a)\neq -1}} f_a + \sum_{\substack{a \in \mathcal{A} \\ h(a)=v, r(a)=1}} f_a + \sum_{\substack{a \in \mathcal{A} \\ h(a)=v \\ j(a)=-1}} f_a - \sum_{\substack{a \in \mathcal{A} \\ t(a)=v \\ j(a)=-1}} f_a \qquad v \in \mathcal{V} \setminus \{0\}, \tag{24}$$

$$\sum_{\substack{a \in \mathcal{A} \\ h(a)=v, r(a)=1}} f_a + \sum_{\substack{a \in \mathcal{A} \\ h(a)=v \\ j(a)=-1}} f_a \geq \sum_{\substack{a \in \mathcal{A} \\ t(a)=v \\ j(a)=-1}} f_a \qquad v \in \mathcal{V} \setminus \{0\}, \tag{25}$$

$$\sum_{\substack{a \in \mathcal{A} \\ t(a)=0}} f_a + \sum_{\substack{a \in \mathcal{A} \\ h(a)=0}} f_a = 2 \sum_{\substack{a \in \mathcal{A} \\ r(a)=1}} f_a, \tag{26}$$

$$\sum_{\substack{a \in \mathcal{A} \\ j(a)=j}} f_a \leq b_j \qquad j \in \mathcal{J}, \tag{27}$$

$$f_a \in \mathbb{N}_0 \qquad a \in \mathcal{A} \setminus \{(L/2, L/2, 0, 1)\}, \tag{28}$$

$$f_a \in \mathbb{Z} \qquad a = (L/2, L/2, 0, 1). \tag{29}$$

In addition to incorporating the necessary changes to the objective function and the demand constraints, an update was also made in the flow constraints to handle the forward loss arcs. The existing elements were updated so as to exclude the forward loss arcs (except for the second element in the right-hand side as all forward loss arcs have $r(a) = 0$), with those being handled by the two newly added elements. Since a forward loss arc a elongates a reflected item arc, it removes one unit of flow coming from a reflected arc at $t(a)$ (fourth element) and transfers it to $h(a)$ (third element). Finally, additional constraints (25) ensure that every forward loss arc is preceded by another forward loss arc or by a reflected item arc. We highlight the similarities (from a modeling point of view) between these forward loss arcs and the downgrade arcs introduced in the previous section.

5.2 Alternative pattern representations in the reflect formulation

We identified two main issues with the existing reflect formulation for the SSP. First, considering forward loss arcs significantly increases the number of constraints in the model, and the fact that these arcs must be positioned after a reflected item arc creates additional complications when trying to extend the formulation to problems closely related to the SSP. For the VSSSP, we confirm that, while feasible, an adaptation of the reflect formulation with forward loss arcs performs significantly worse than an adaptation of the arcflow formulation. For the so-called "overflowing bin packing problem", Martinovic and Stradat [28] highlighted the difficulty of even designing a correct model based on the reflect formulation with forward loss arcs. Second, the fact that the reduction procedure used in Step 12 of Algorithm A.3 could not be extended in any form to the SSP naturally increases the number of variables in the model compared to its corresponding CSP version. In the following, we first describe a valid reflect formulation with backward loss arcs, then demonstrate how a reduction procedure close to that of the CSP is also valid for the SSP, and finally show how such a procedure can be fully utilized to significantly decrease the number of variables used in the model.

Backward loss arcs When backward loss arcs are used, it is now the standard subpath that should be adjusted so that it collides at the vertex that is the head of the reflected subpath. As there is no way to make a reflected subpath standard again in the absence of forward loss arcs, the standard subpath should not include any reflected arc. This is problematic since, as previously discussed, some patterns (e.g., those containing two items, each with a length greater than half the threshold) need a reflected item arc in both subpaths. To solve this issue, we introduce a new set of arcs, one per item type $j \in \mathcal{J}$, of the form $(\max\{0, \frac{L}{2} - l_j\}, \frac{L}{2}, j, 0)$. Note that such an arc is created for an item j only if there also exists a reflected item arc for j. When the standard subpath would have used reflected item arc $(d, L - (d + l_j), j, 1)$ followed by forward loss arcs and a negative flow on the reflected connection arc, it now uses backward loss arcs from d to $\max\{0, \frac{L}{2} - l_j\}$, followed by the newly introduced arc $(\max\{0, \frac{L}{2} - l_j\}, \frac{L}{2}, j, 0)$, followed by more backward loss arcs from $\frac{L}{2}$ until the standard subpath collides with the reflected subpath. Reusing the example with two items of length 6 and a roll with threshold 10: a solution now includes reflected item arc $(0, 4, 1, 1)$, the newly introduced standard item arc $(0, 5, 1, 0)$, and a backward loss arc $(5, 4, 0, 0)$. Unlike forward loss arcs, backward loss arcs do not need to be treated separately from the other arcs, hence, we do not use "-1" as a distinguishable third element. Using backward loss arcs instead of forward loss arcs removes up to L constraints at the cost of up to $|\mathcal{J}|$ additional variables. It also removes the need for the reflected connection arc to be allowed to take a negative value.

Adapted reduction procedure As observed in the literature [32], some SSP patterns can only be represented in the reflect formulation (independently of the kind of loss arcs used) by a pair of colliding subpaths in which the reflected item arc a is such that $t(a) > h(a)$, making the reduction procedure used in Step 12 of Algorithm A.3 invalid for the SSP in its current form. In the following, we propose a new theorem from which we derive a valid adaptation of the reduction procedure for the SSP. To this end, we first provide a preparatory statement that will be used in the proof of the theorem.

Proposition 1. *Any feasible SSP pattern that is represented in the reflect formulation by a pair of colliding subpaths in which each subpath contains a reflected item arc whose head is lower than its tail (i.e., $a = (t(a), h(a), j(a), 1)$ with $t(a) > h(a)$ and $a' = (t(a'), h(a'), j(a'), 1)$ with $t(a') > h(a')$) is a non-minimal pattern (i.e., it contains at least one redundant item).*

Proof. Given in Section C.1 of the appendix. □

Theorem 1. *Any feasible minimal SSP pattern with length $L' = L + k$ ($k \geq 0$) can be represented in the reflect formulation with forward loss arcs by a pair of colliding subpaths in which either (I) each subpath contains a reflected item arc, one satisfying $t(a) \leq h(a)$ and the other satisfying $t(a') \leq h(a') + k$, or (II) only one subpath contains a reflected item arc, and this arc satisfies $t(a) \leq h(a) + k$.*

Proof. Given in Section C.2 of the appendix. □

A similar theorem can be derived for the reflect formulation with backward loss arcs, although the associated proof (which is analogous to the one provided in the appendix) no longer needs to consider case (I). Observing that any network containing all minimal SSP patterns results in a correct ILP formulation, a useful application of the theorem is the following: if a reflected item arc $a \in \mathcal{A}$ with $t(a) > h(a)$ is required, then it must be to create a pattern of length $L' \geq L + t(a) - h(a)$, that is, a pattern that includes at least $t(a) - h(a)$ units of loss. We therefore suggest incorporating this loss directly into the reflected item arc by setting $h(a)$ to $t(a)$ for every such arc. Reusing the example with three items of length 9 and a roll of length 20, the reflected item arc $(9, 2, 1, 1)$ becomes $(9, 9, 1, 1)$, and the solution is thus composed of arcs $(0, 9, 1, 0)$ and $(9, 9, 1, 1)$ for the reflected subpath, and arc $(0, 9, 1, 0)$ for the standard subpath, both of which collide at vertex 9. While this procedure does not immediately remove any reflected item arc (though it may slightly reduce the number of loss arcs as the original $h(a)$ is not included in $\mathcal{V}$ anymore), we empirically observe that applying it is beneficial.

Conversion variables Our last strategy comes from the observation that, after applying the adapted reduction procedure, multiple reflected item arcs associated with different item types may have the same tail and head. When that situation occurs, we suggest keeping only the arc related to the smallest item type j among those with the same tail and head, with the idea that, if the arc was needed for a longer item type $j' < j$, then one could always transform an item of type j' into an item of type j without violating the feasibility condition of the considered pattern. In practice, we introduce a new set of integer decision variables τ_j ($j \in \mathcal{J} \setminus \{|\mathcal{J}|\}$) that we call *conversion variables*, which indicate the number of times an item of type j is converted into an item of type $j + 1$. Using conversion variables potentially merges up to $|\mathcal{J}| - 1$ reflected item arcs for each vertex that is the tail of a reflected arc, at the cost of $|\mathcal{J}| - 1$ additional variables in the model. We also point out that it is possible for the newly introduced arcs $(\max\{0, \frac{L}{2} - l_j\}, \frac{L}{2}, j, 0)$, which are necessary for using backward loss arcs, to have the same tail (0) and head ($\frac{L}{2}$) while being associated with different item types. If such a situation occurs, we also keep only the arc related to the smallest item type among those with tail 0 and head $\frac{L}{2}$. Finally, we highlight that conversion variables could also be used without the adapted reduction procedure, following an idea originally developed for dual optimal inequalities in column generation based approaches [23], but this serves no purpose from a modeling point of view, as it adds more variables and increases symmetry. Still, we also empirically evaluate the impact of using conversion variables without the adapted reduction procedure.

The reflect graph incorporating all these procedures can be constructed using Algorithm A.5 in Section C.3 of the appendix, and the associated ILP model is described in Section C.4 of the appendix.

5.3 Illustrative example

Consider an SSP instance composed of four item types with lengths $l_1 = 8$, $l_2 = 5$, $l_3 = 4$, and $l_4 = 3$ and corresponding demands $b_1 = 2$, $b_2 = 2$, $b_3 = 2$, and $b_4 = 2$, with a roll threshold $L = 10$. The corresponding arcflow graph without loss arcs is shown at the top of Figure 2, followed by the arcflow graph with reversed loss arcs. Note that the pseudocode used to construct these graphs is available in the paper by Martinovic et al. [32], in which these two models were introduced. The four graphs below are those obtained after applying adapted versions of Algorithms A.3 and A.5 to produce the graph for reflect with forward loss arcs (also introduced by Martinovic et al. [32]), and the three new graphs for reflect with backward loss arcs, reflect with backward loss arcs and the adapted reduction procedure, and reflect with backward loss arcs, the adapted reduction procedure, and conversion variables. Item arcs are displayed as colored solid lines—red, blue, yellow, and green for item type 1, 2, 3, and 4, respectively—whereas the other arcs are shown as black dotted lines. To improve visibility, each reflected graph is divided into two subgraphs: the one on the right contains all the reflected items arcs, connection arcs, and forward loss arcs, while the one on the left contains the remaining arcs.

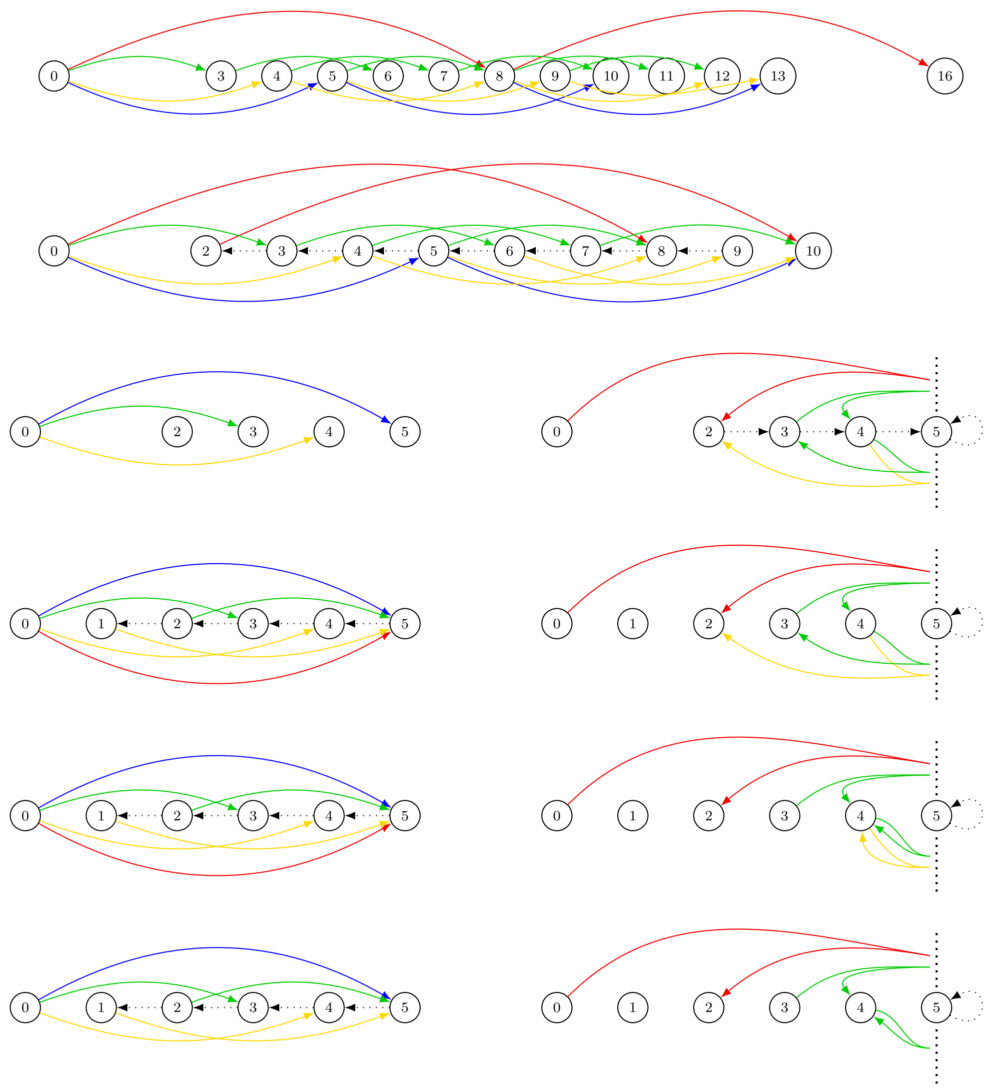
In this example, the arcflow graph contains 17 arcs and 13 nodes, the arcflow graph with reversed loss arcs contains 20 arcs (including 7 reversed loss arcs) and 10 nodes, and the graph associated with the reflect formulation with forward loss arcs contains 11 arcs and 5 nodes. For example, a roll that contains two items of type 1 is represented by path $(0, 8, 1), (8, 16, 1)$ in arcflow, by path $(0, 8, 1)$, $(8, 7, 0)$, $(7, 6, 0)$, $\ldots$, $(3, 2, 0)$, $(2, 10, 1)$ in arcflow with reversed loss arcs, and by subpaths $(0, 2, 1, 1)$, $(2, 3, 0, 0)$, $(3, 4, 0, 0)$, $(4, 5, 0, 0)$, negative $(5, 5, 0, 1)$ and $(0, 2, 1, 1)$, $(2, 3, 0, 0)$, $(3, 4, 0, 0)$, $(4, 5, 0, 0)$ colliding at node 5 in reflect with forward loss arcs.

When using backward loss arcs, standard item arcs $(0, 5, 1, 0), (1, 5, 3, 0)$, and $(2, 5, 4, 0)$ must be added, in addition to the swap from forward to backward loss arcs. While forward loss arcs start from vertex 2 (the lowest vertex that is the head of a reflected arc), backward loss arcs end at vertex 1 (to reach the lower vertex that is either the head of a reflected arc or the tail of one of the newly added standard item arcs). The same roll that contains two items of type 1 is now represented by subpaths $(0, 2, 1, 1)$ and $(0, 5, 1, 0)$, $(5, 4, 0, 0)$, $(4, 3, 0, 0)$, $(3, 2, 0, 0)$ colliding at node 2.

When using the adapted reduction procedure, reflected item arc $(4, 2, 3, 1)$ becomes $(4, 4, 3, 1)$ and reflected item arc $(4, 3, 4, 1)$ becomes $(4, 4, 4, 1)$. For example, a roll that contains two items of type 3 and one item of type 4 is represented by subpaths $(0, 4, 3, 0), (4, 3, 4, 1)$ and $(0, 4, 3, 0)$, $(4, 3, 0, 0)$ colliding at node 3 without the adapted reduction procedure, and by subpaths $(0, 4, 3, 0)$, $(4, 4, 4, 1)$ and $(0, 4, 3, 0)$ colliding at node 4 with it.

Finally, when also using the conversion variables, reflected item arc $(4, 4, 3, 1)$ is removed because a corresponding arc for a smaller item type, $(4, 4, 4, 1)$ already exists. Similarly, standard item arc

Figure 2: The two arcflow and four reflect graphs corresponding to the example



(0, 5, 1, 0) is removed because of the presence of (0, 5, 2, 0). For example, a roll that contains one item of type 2 and two items of type 3 is represented by subpaths (0, 4, 3, 0), (4, 4, 3, 1) and (0, 5, 2, 0), (5, 4, 0, 0) colliding at node 4 without the conversion variables, and by subpaths (0, 4, 3, 0), (4, 4, 4, 1) and (0, 5, 2, 0), (5, 4, 0, 0) colliding at node 4 with them, together with the conversion of an item of type 3 into an item of type 4.

6 New flow formulations for the VSSSP

Whereas it has never been formally studied in the literature, the VSSSP fits the unified problem definition introduced in Section 2, and can be seen as the counterpart of the VSCSP where one aims to construct rolls above certain thresholds, each with a certain profit, given a limited supply of items. The VSSSP finds applications in cognitive radio networks for spectrum assignment. There, small unused pieces (the item types) in a frequency band must be merged into communication channels (the roll types) exceeding specific size thresholds [29, 30]. The VSSSP is also the basis of the optimization problem faced by users of certain loyalty programs. There, the user has a list of items they wish to purchase (the item types), and buying certain quantities earns them points (e.g., 1 point for €10 spent, 2 points for €20). Buying all items at once is impossible (e.g., due to capacity issues).

The goal is to split the items into bundles to maximize the number of points earned.

In the following, we first describe extensions of the arcflow, arcflow with reversed loss arcs, reflect with forward loss arcs, and reflect with backward loss arcs formulations to the VSSSP. Next, we discuss how the reflect-specific alternative pattern representations for the VSCSP/MKP and SSP extend to the VSSSP, and highlight certain interactions to consider when applying particular combinations of these new representations. As in the previous section, we will use L_r to denote the threshold of roll type $r \in \mathcal{R}$ instead of H_r. The previously considered orderings on both the roll and the item types also hold in this section. In addition, we assume $l_j \leq L_1$ $(j \in \mathcal{J})$; if not, l_j is set to L_1.

6.1 Adapting the arcflow formulations

As in the VSCSP/MKP extension, a straightforward adaptation of the arcflow formulation to the VSSSP adds a new target node T representing the end of a roll, along with a set of roll arcs. One could think of (at least) two variants for these roll arcs:

- A unique set of $|\mathcal{R}|$ arcs $(L_r, T, 0, r)$, together with a set of reversed loss arcs $(e, d, 0, 0)$, where $e, d \in \mathcal{V}$ and $e > d$, in order to allow path shortening towards one of the L_r vertices that is the tail of a roll arc;
- one roll arc $(d, T, 0, r)$ for each roll type $r \in \mathcal{R}$ and each vertex $d \in \mathcal{V}$ such that $d \geq L_r$.

Because of the smaller number of resulting arcs, we opted for the first variant. A second choice concerns merging all item arcs $(d, d + l_j, j, 0)$ such that $d < L_1$ and $d + l_j \geq L_1$ into a single item arc $(L_1 - l_j, L_1, j, 0)$. Note that such a merging is not directly possible for shorter rolls, as the original arcs may still be required in paths corresponding to the longest roll—and keeping both merged and unmerged versions would defeat the purpose of arc merging, which is to reduce the total number of arcs. In our experiments, we tested (i) a version without merging, corresponding to (what we consider as) the adaptation of the standard SSP arcflow extension, and (ii) a version with merging, corresponding to the adaptation of the SSP arcflow with reversed loss arcs.

6.2 Adapting the reflect formulations

Adapting the reflect formulation with forward loss arcs to the VSSSP can be done as follows: starting from Algorithm 1, remove the reduction procedure in Step 13, as it is not valid for the VSSSP in its current form, and update the arc definition in Step 15 to $\max\{0, L_r - (d + l_j)\}$, since the head of a newly created arc may now be negative if $l_j > L_r$. As for the definition of loss arcs in Steps 19-20, since they extend a reflected item arc, they must now carry the roll type information. Thus, one should create a set of forward loss arcs for each roll type $r \in \mathcal{R}$, allowing the reflected item arcs associated with r to reach every vertex $v \in \mathcal{V}$ such that $v \leq \frac{L_r}{2}$. We note that this extension requires $\mathcal{O}(L_r)$ additional variables per roll type $r \in \mathcal{R}$ corresponding to the roll-specific forward loss arcs, and $\mathcal{O}(L_r)$ additional constraints per roll type to ensure that each forward loss arc associated with r is preceded by another forward loss arc associated with r or a reflected item arc associated with r. As the number of roll types $|\mathcal{R}|$ increases, these additional variables and constraints quickly become prohibitive, the size of the resulting model becoming even larger than that of the arcflow formulations.

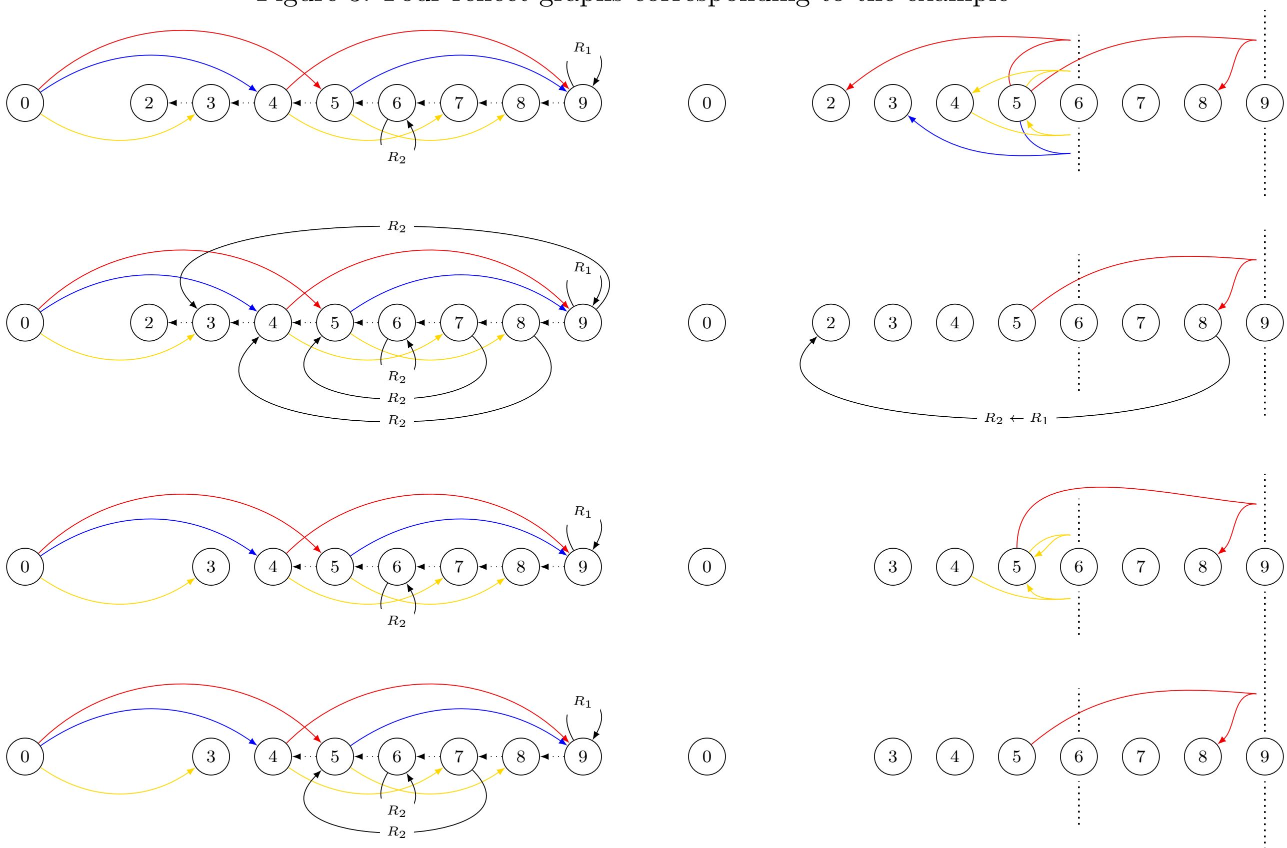
Adapting the reflect formulation with backward loss arcs to the VSSSP can be done as follows: starting from Algorithm 1, remove Step 13 and update Step 15 as previously described, create the arc $(\max\{0, \frac{L_1}{2} - l_j\}, \frac{L_1}{2}, j, 0)$ for each item type $j \in \mathcal{J}$ as done in the SSP, and add the set of backward loss arcs linking every vertex $v \in \mathcal{V} \setminus \{0\}$ as done in Steps 23-24 of Algorithm A.5. Note that additional arcs are not needed for shorter rolls, as one can either use the corresponding standard item arc created at Step 10 of Algorithm 1 followed by backward loss arcs, or use $(\max\{0, \frac{L_1}{2} - l_j\}, \frac{L_1}{2}, j, 0)$ if the standard item arc was not created. We note that this extension does not require any additional constraints and only adds up to $|\mathcal{J}|$ new variables (as was the case for the SSP).

6.3 Adapting (and combining) the newly introduced pattern representations

If we consider the reflect formulation with backward loss arcs, the concepts of small-roll reflected arcs, extended reflected connection arcs, and downgrade arcs naturally extend from the VSCSP/MKP to the VSSSP. Similarly, the concepts of adapted reduction procedure and conversion variables naturally extend from the SSP to the VSSSP.

It is worth highlighting that when all these are used simultaneously, the potential interactions require a bit of attention. To illustrate this, let us consider a VSSSP instance composed of three item types with lengths $l_1 = 5$, $l_2 = 4$, and $l_3 = 3$ and corresponding demands $b_1 = 2$, $b_2 = 1$, and $b_3 = 1$, with two roll types with thresholds $L_1 = 18$ and $L_2 = 12$. The corresponding reflect graph with backward loss arcs is depicted at the top of Figure 3, followed by a version using the VSCSP/MKP-specific improvements, followed by a version using the SSP-specific improvements, followed by a version using all improvements combined. We reuse the same convention for displaying the arcs as in previous figures.

Figure 3: Four reflect graphs corresponding to the example



Comparing the first two graphs, reflected item arc $(5, 2, 1, 2)$ in the first graph is replaced by downgrade arc $(8, 2, -1, 2)$, and reflected item arcs $(5, 3, 2, 2)$, $(5, 4, 3, 2)$, and $(4, 5, 3, 2)$ in the first graph are replaced by extended reflected connection arcs $(9, 3, 0, 2)$, $(8, 4, 0, 2)$, and $(7, 5, 0, 2)$, respectively.

Comparing the first and the third graphs, reflected item arcs $(5, 2, 1, 2)$, $(5, 3, 2, 2)$, and $(5, 4, 3, 2)$ in the first graph become $(5, 5, 1, 2)$, $(5, 5, 2, 2)$, and $(5, 5, 3, 2)$, respectively, of which only the arc for the smallest item type, $(5, 5, 3, 2)$, is kept because these three arcs are associated with the same roll type and share the same head and tail. Note that reversed loss arcs terminate at vertex $v = \min\{5, 4\}$, 5 being the lowest vertex that is the head of a reflected item arc, here $(4, 5, 3, 2)$, and 4 being the lowest tail of one of the newly added standard item arcs (those ending at $\frac{L_1}{2}$), here $(4, 9, 1, 0)$.

Comparing the third and the fourth graphs, reflected item arcs $(4, 5, 3, 2)$ and $(5, 5, 3, 2)$ in the third graph are both replaced by extended reflected connection arc $(7, 5, 0, 2)$. For example, a roll of type 2 that contains two items of type 1 and one item of type 3 is represented by subpaths $(0, 5, 1, 0)$, $(5, 5, 3, 2)$ and $(0, 5, 1, 0)$ colliding at node 5 in the third graph and by subpaths $(0, 5, 1, 0)$, $(5, 8, 3, 0)$, $(8, 7, 0, 0)$, $(7, 5, 0, 2)$ and $(0, 5, 1, 0)$ colliding at node 5 in the fourth graph. Observe that the units of loss integrated in the reflected item arcs by the adapted reduction procedure, here $(5, 5, 3, 2)$, are replaced by reversed loss arcs inserted between the associated standard item arc, here $(5, 8, 3, 0)$, and the associated extended reflected connection arc, here $(7, 5, 0, 2)$. From an algorithmic perspective, one must ensure that the head of every corresponding extended reflected connection arc belongs to $\mathcal{V}$ so that it can always be reached with reversed loss arcs. Another point of attention, which concerns the downgrade arcs, is described in Section D.1 of the appendix.

7 Computational experiments

In this section, we empirically evaluate the performance of each of the proposed reflect-specific alternative pattern representations, together with the usefulness of additional dummy variables to model the number of times an item type or a roll type is included in the solution. Regarding additional dummy variables, we tested three configurations: (i) not including them at all (V0), (ii) including them (V1), and (iii) including them and giving them a higher branching priority using the `GRB_IntAttr_BranchPriority` feature (V2). If additional dummy variables are included, it is always for the element considered in the objective function (i.e., the item type for the MKP and the roll type for the CSP, SSP, VSCSP, and VSSSP), and they are linked with the f_a variables by an inequality, as done by Dell'Amico et al. [12].

For each problem among the CSP, SSP, VSCSP, MKP, and VSSSP, we selected a set of widely used datasets for which we provide: (i) the results of the existing arcflow and reflect formulations as reported in the literature, (ii) the results of these formulations obtained from a rerun using our computational environment, (iii) the results of the reflect formulation using the newly proposed alternative pattern representations, and (iv) the results of the state-of-the-art ("SOTA") approach as reported in the literature. Elements (i) and (iv) are either recomputed from a source file made available to us or taken directly from the relevant paper. Our algorithms were all implemented in `C++` and can be downloaded from `https://github.com/mdelorme2/Extending-the-reflect-flow-formulation-to-variable-sized-1D-cutting-and-skiving-stock-problems`. All computational tests were executed on a virtual machine `AMD EPYC-Rome Processor` with 2.00 GHz and 64 GB of RAM, running under `Ubuntu 20`. The ILP models were solved using `Gurobi 12.0.2`. A single core was used for the tests and the barrier algorithm was used to solve the root nodes of the models. For every run, a time limit of 3600 seconds was imposed. For the sake of conciseness, we sometimes refer to "the performance of a model" in the following, which means the "performance obtained by solving the model with the specified ILP solver".

7.1 Results for the CSP

For the CSP, we used the same datasets as those employed by Delorme and Iori [13], namely (1) a set of 1615 instances proposed in various articles over the last decades, which from today's perspective can be considered "easily solvable", (2) a set of 300 instances proposed by Delorme et al. [14] characterized by their structural difficulty (it is either difficult to find an optimal solution—dataset AI—or it is difficult to prove that a given solution is optimal—dataset ANI), and (3) two sets of 60 instances proposed by Gschwind and Irnich [18] characterized by very large roll lengths (the difficulty here lies in the number of variables and constraints required by flow formulations).

We present in Table 2 the results reported in the literature and those obtained after rerunning the flow formulations (reruns are denoted with attribute "(r)"). The first two columns of the table identify the name of the dataset and the number of instances it includes, whereas the following columns provide, for each approach, the number of optimal solutions found (column "#opt") and the average CPU time in seconds over each run of the dataset, including the ones terminated within the time limit (column "T(s)"). Note that because of rounding, it is possible for the solution time displayed in the table to be equal to 0, which actually means that it was below 0.05 seconds. Columns highlighted in gray are those for which the results were obtained with a different computational setup from the one used in this work.

Table 2: Comparison of the CSP results with those reported in the literature

Dataset	#inst.	arcflow V0 [13]		arcflow V0 (r)		reflect V0 [13]		reflect V0 (r)		SOTA [8]	
		#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)
Falkenauer U	80	80	0.1	80	0.1	80	0.0	80	0.0	80	0.0
Falkenauer T	80	80	1.3	80	0.5	80	0.3	80	0.4	80	0.1
Hard28	28	28	10.9	28	5.8	28	19.0	28	3.0	28	6.4
Scholl 1	720	720	0.1	720	0.1	720	0.0	720	0.0	720	0.1*
Scholl 2	480	480	84.3	480	9.1	480	9.6	480	2.0	480	0.1*
Scholl 3	10	10	324.2	10	367.4	10	8.6	10	6.1	10	0.1*
Schwerin1	100	100	0.9	100	0.4	100	0.3	100	0.2	100	0.1*
Schwerin2	100	100	0.7	100	0.4	100	0.2	100	0.2	100	0.1*
Waescher	17	9	1780.5	17	305.0	17	555.5	17	135.4	17	0.3
Total (1)	1615	1607	46.2	1615	8.4	1615	9.1	1615	2.2	1615	0.2
AI 200	50	50	233.7	50	33.2	50	45.5	50	9.1	50	0.4
AI 400	50	19	2461.3	31	2132.5	21	2297.4	43	1077.4	50	5.0
AI 600	50	0**	3600.0	0	3600.0	0**	3600.0	10	3290.0	50	57.1
ANI 200	50	35	1397.7	47	508.7	50	67.2	50	49.8	50	0.4
ANI 400	50	3	3474.4	2	3520.3	10	3083.6	13	3134.4	50	2.2
ANI 600	50	0**	3600.0	0	3600.0	0**	3600.0	0	3600.0	50	11.8
Total (2)	300	107	2461.2	130	2232.5	131	2115.6	166	1860.1	300	12.8
GI AA	60	20	2699.3	20	2601.0	60	179.8	60	36.2	60	14.5
GI BA	60	20	2727.3	20	2637.1	60	380.1	60	37.6	59	102.0
Total (3)	120	40	2713.3	40	2619.1	120	280.0	120	36.9	119	58.3
Total (1,2,3)	2035	1754	559.5	1785	490.2	1866	335.6	1901	278.1	2034	5.4

*We copied the averages over all 1210 Scholl and 200 Schwerin instances reported in [8].
**These results were not directly reported in [13], but we assume that none of these instances could be solved at that time.

Although none of the contributions of this work directly relate to the CSP, these experiments allow us to make a number of interesting observations. First, we find that using dummy variables to model the number of rolls used in the solution has no impact on the performance of arcflow and reflect for the CSP: the computing times were always within a fraction of a second for the three tested configurations, which is why we do not report the results of the other two configurations in the table. Next, both the arcflow and reflect formulations significantly benefit from the improvements made to the computational setup. Differences in computing power are often highlighted in the literature, but one should also acknowledge the significant improvements made to ILP solvers (for comparison, Delorme and Iori [13] used `Gurobi 6.5` in their experiments). In particular, we highlight that the arcflow formulation can now solve all 1615 instances of the first dataset, that both the arcflow and the reflect formulations now perform much better on the AI instances, and that reflect performs much better on the tested GI instances. On the other hand, it is also interesting to note that little improvement is observed for the ANI instances. Finally, while we confirm the empirical advantage of reflect over arcflow, we also acknowledge that the state-of-the-art approach, a sophisticated B&P algorithm proposed by da Silva and Schouery [8], clearly outperforms both formulations. We further highlight that this B&P algorithm is also able to solve GI, AI, and ANI instances that were not tested in our experiments (preliminary experiments showed that none of these instances could be solved by arcflow or reflect).

7.2 Results for the SSP

For the SSP, we reused the exact same datasets as the ones tested for the experiments on the CSP. We first report in Table 3 the results obtained by the reflect formulation with forward loss arcs (reflect+f) introduced by Martinovic et al. [32], as well as those obtained by the three new variants: the reflect formulation with backward loss arcs (reflect+b), the reflect formulation with backward loss arcs and the adapted reduction procedure (reflect+b+red), and the reflect formulation with backward loss arcs, the adapted reduction procedure, and conversion variables (reflect+b+red+con). Since there are different ways to concatenate the reduction methods presented in Section 5, we also include the reflect formulation with backward loss arcs and conversion variables alone (reflect+b+con). For each model, we also report the average number of variables (column "#var") and constraints (column "#cons") involved, in thousands.

Table 3: Detailed SSP results for the tested reflect formulations

Dataset	#inst.	reflect+f V0				reflect+b V0				reflect+b+red V0				reflect+b+con V0				reflect+b+red+con V0			
		#opt	T(s)	#var	#cons	#opt	T(s)	#var	#cons	#opt	T(s)	#var	#cons	#opt	T(s)	#var	#cons	#opt	T(s)	#var	#cons
Falkenauer U	80	80	0.1	1.5	0.2	80	0.0	1.6	0.2	80	0.0	1.6	0.1	80	0.0	1.7	0.2	80	0.1	0.9	0.1
Falkenauer T	80	80	0.4	9.0	1.0	80	0.4	9.1	0.6	80	0.4	9.0	0.4	80	0.3	9.2	0.6	80	0.7	2.9	0.4
Hard28	28	28	1.7	14.6	1.1	28	1.2	14.8	0.7	28	0.9	14.7	0.6	28	0.8	14.9	0.7	28	2.8	11.3	0.6
Scholl 1	720	720	0.0	0.9	0.2	720	0.0	0.9	0.1	720	0.0	0.9	0.1	720	0.0	1.0	0.1	720	0.1	0.6	0.1
Scholl 2	480	480	8.0	17.2	0.7	480	2.4	17.3	0.5	480	2.4	17.3	0.5	480	2.9	17.4	0.5	480	12.0	13.3	0.5
Scholl 3	10	10	24.9	120.3	41.6	10	10.7	120.7	21.1	10	4.2	110.7	11.1	10	10.1	120.9	21.1	10	2.5	33.8	11.1
Schwerin1	100	100	0.2	3.7	0.4	100	0.1	3.8	0.3	100	0.1	3.8	0.3	100	0.1	3.8	0.3	100	0.2	3.5	0.3
Schwerin2	100	100	0.2	4.0	0.5	100	0.1	4.0	0.3	100	0.1	4.0	0.3	100	0.1	4.1	0.3	100	0.1	3.8	0.3
Waescher	17	17	35.8	57.8	7.2	17	32.1	57.9	4.3	17	32.5	57.9	4.3	17	31.3	57.9	4.3	17	59.4	51.4	4.3
Total (1)	1615	1615	3.0	8.1	0.8	1615	1.2	8.2	0.5	1615	1.1	8.1	0.4	1615	1.3	8.3	0.5	1615	4.3	5.8	0.4
AI 200	50	50	8.5	48.8	2.0	50	11.2	49.1	1.1	50	8.8	49.1	1.1	50	9.7	49.3	1.1	50	12.0	43.2	1.1
AI 400	50	42	992.7	381.8	7.0	44	942	382.3	3.8	47	745.1	382.3	3.8	44	909.0	382.7	3.8	43	974.1	336.8	3.8
AI 600	50	6	3468.8	1211.3	14.5	3	3539.1	1212.2	7.7	7	3380.4	1212.2	7.7	5	3467.6	1212.7	7.7	4	3536.3	1066.2	7.7
ANI 200	50	50	106.4	48.0	2.0	50	61.7	48.3	1.1	50	45.5	48.3	1.1	50	61.0	48.4	1.1	50	102.7	42.4	1.1
ANI 400	50	11	3249.0	379.7	7.0	12	3263.9	380.2	3.8	20	2888.4	380.2	3.8	13	3121.5	380.6	3.8	15	3121.2	335.3	3.8
ANI 600	50	0	3600.0	1206.9	14.5	0	3600.0	1207.8	7.7	0	3600.0	1207.8	7.7	0	3600.0	1208.3	7.7	0	3600.0	1062.9	7.7
Total (2)	300	159	1904.2	546.1	7.8	159	1903.0	546.6	4.2	174	1778.0	546.6	4.2	162	1861.5	547.0	4.2	162	1891.1	481.1	4.2
GI AA	60	59	634.7	288.8	123.2	60	147.8	289.2	61.9	60	83.7	251.2	23.9	60	130.0	289.5	61.9	60	148.1	90.7	23.9
GI BA	60	59	566.4	364.8	230.6	60	144.3	365.2	115.6	60	68.2	289.8	40.1	60	133.4	365.5	115.6	60	162.6	110.4	40.1
Total (3)	120	118	600.5	326.8	176.9	120	146.1	327.2	88.8	120	76.0	270.5	32.0	120	131.7	327.5	88.8	120	155.4	100.6	32.0
Total (1,2,3)	2035	1892	318.5	106.2	12.2	1894	290.1	106.4	6.2	1909	267.5	103.0	2.8	1897	283.2	106.5	6.2	1897	291.4	81.5	2.8

First, we observe that swapping forward loss arcs with backward loss arcs has a positive effect, especially on the average computing time for the GI instances. As expected, the number of constraints is roughly halved, at the cost of only a few hundred extra variables on average. Using the adapted reduction procedure also has a positive effect on average, again especially for the GI instances, for which we observe a significant decrease in both the average number of variables and constraints. This decrease is caused by a reduction in the number of vertices included in $\mathcal{V}$, which results in fewer loss arcs and flow-conservation constraints. Typically, a small vertex $0 < v \ll L$ belongs to $\mathcal{V}$ only if (i) v is the head of a standard item arc, which only occurs if the instance contains small item types, or (ii) v is the head of a reflected item arc, which is less frequent when the adapted reduction procedure is used. It is also interesting to observe that the adapted reduction procedure seems to have a positive impact when solving the AI and ANI instances, which is not related to a reduction in the model size at all. This could be due to the structure of these instances, whose best lower bound assumes the absence of loss arcs in the solution, whereas the adapted reduction procedure integrates loss arcs into many reflected item arcs. Unsurprisingly, using conversion variables without the adapted reduction procedure does not seem to bring any major improvement. What comes as a surprise is that, despite a significant decrease in both the average number of variables and constraints, adding conversion variables to the adapted reduction procedure seems to deteriorate the performance on the tested instances. This suggests that not every reduction in model size directly translates into a performance increase, highlighting the importance of extensive empirical evaluations.

We compare in Table 4 the results reported in the literature with those obtained after rerunning the flow formulations. Approach "larcflow" corresponds to the arcflow with reversed loss arcs, whereas "arcflow" corresponds to the version without. For the sake of comparison, we also include the results of our best approach, reflect+b+red,

Table 4: Comparison of the SSP results with those reported in the literature

Dataset	#inst.	arcflow V0 [32]		arcflow V0 (r)		larcflow V0 [32]		larcflow V0 (r)		reflect+f V0 [32]		reflect+f V0 (r)		reflect+b+red V0		SOTA [8]	
		#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)
Falkenauer U	80	80	0.2	80	0.1	80	0.2	80	0.1	80	0.1	80	0.1	80	0.0	80	0.0
Falkenauer T	80	80	4.9	80	1.5	80	1.1	80	0.7	80	0.9	80	0.4	80	0.4	80	0.5
Hard28	28	28	6.3	28	2.2	28	7.4	28	3.1	28	3.0	28	1.7	28	0.9	28	0.1
Scholl 1	720	720	0.1	720	0.1	720	0.1	720	0.0	720	0.0	720	0.0	720	0.0	720	0.2^*
Scholl 2	480	480	40.2	480	9.9	480	47.4	480	8.7	480	20.8	480	8.0	480	2.4	480	0.2^*
Scholl 3	10	0	3600.0	0	3600.0	9	2306.6	10	827.0	10	65.1	10	24.9	10	4.2	10	0.2^*
Schwerin1	100	100	0.5	100	1.2	100	0.4	100	0.7	100	0.1	100	0.2	100	0.1	100	0.0^*
Schwerin2	100	100	2.6	100	1.2	100	1.3	100	0.7	100	0.4	100	0.2	100	0.1	100	0.0^*
Waescher	17	17	83.0	17	234.1	17	117.6	17	264.3	17	257.3	17	35.8	17	32.5	17	0.0
Total (1)	1615	1605	35.7	1605	28.0	1614	30.0	1615	10.7	1615	9.4	1615	3.0	1615	1.1	1615	0.2
AI 200	50	50	99.2	50	37.4	50	117.5	50	74.8	50	26.3	50	8.5	50	8.8	50	0.5
AI 400	50	15	3219.5	34	2205.1	10	3298.2	25	2442.4	31	2016.9	42	992.7	47	745.1	50	8.9
AI 600	50	$0^\dagger$	3600.0	0	3600.0	0	3600.0	0	3600.0	1	3588.2	6	3468.8	7	3380.4	48	214.9
ANI 200	50	43	1036.7	48	459.5	18	2843.5	36	1643.1	50	238.5	50	106.4	50	45.5	50	0.7
ANI 400	50	0	3600.0	3	3508.9	0	3600.0	0	3600.0	0	3600.0	11	3249.0	20	2888.4	50	5.7
ANI 600	50	$0^\dagger$	3600.0	0	3600.0	$0^\dagger$	3600.0	0	3600.0	$0^\dagger$	3600.0	0	3600.0	0	3600.0	49	99.3
Total (2)	300	108	2525.9	135	2235.2	78	2843.2	111	2493.4	132	2178.3	159	1904.2	174	1778.0	297	55.0
GI AA 125	20	3	3376.4	6	3092.3	20	918.4	20	453.3	20	4.1	20	1.2	20	0.7	20	14.8^{**}
GI AA 250	20	$0^\dagger$	3600.0	0	3600.0	$0^\dagger$	3600.0	0	3600.0	20	103.9	20	18.7	20	14.5	20	34.6^{**}
GI AA 500	20	$0^\dagger$	3600.0	0	3600.0	$0^\dagger$	3600.0	0	3600.0	0	3600.0	19	1884.1	20	428.2	20	95.2^{**}
GI BA 125	20	4	2964.6	7	2751.8	19	1190	20	603.2	20	4.3	20	1.4	20	1.0	20	14.8^{**}
GI BA 250	20	$0^\dagger$	3600.0	0	3600.0	$0^\dagger$	3600.0	0	3600.0	20	112.2	20	33.0	20	15.2	20	34.6^{**}
GI BA 500	20	$0^\dagger$	3600.0	0	3600.0	$0^\dagger$	3600.0	0	3600.0	2	3536.7	19	1664.9	20	416.8	20	95.2^{**}
Total (3)	120	7	3456.8	13	3374.0	39	2751.4	40	2576.1	82	1226.9	118	600.5	120	76.0	120	48.2^{**}
Total (1,2,3)	2035	1720	604.5	1753	550.7	1731	605.2	1766	528.0	1829	400.9	1892	318.5	1909	267.5	2032	11.1

*We copied the averages over all 1210 Scholl and 200 Schwerin instances reported in [8].
**We copied the averages over all four GI datasets reported in [8], including GI AB and GI BB.
†These results were not directly reported in [32], but we assume that none of these instances could be solved at that time.

Interestingly, all the comments made regarding the results of flow formulations for the CSP also apply to the SSP: (i) using dummy variables has no impact, (ii) improvements in the computational setup can be observed for all formulations, especially reflect, (iii) these improvements are particularly noticeable on the AI instances for all models and on the GI instances for reflect, (iv) it is confirmed that reflect outperforms both arcflow formulations, and (v) the B&P of da Silva and Schouery [8] clearly outperforms all the tested flow formulations.

7.3 Results for the VSCSP

For the VSCSP, we also used the same datasets as those employed by Delorme and Iori [13], namely:

(1) a set of 200 instances proposed by Belov and Scheithauer [3] where $L_1 \leq 10\,000$, $\sum_{j \in \mathcal{J}} b_j \approx 5000$, $|\mathcal{R}| \leq 16$, and the roll types have a linear cost function;

(2) a set of 840 instances proposed by Crainic et al. [7] where $L_1 \leq 300$, $\sum_{j \in \mathcal{J}} b_j \leq 1000$, $|\mathcal{R}| \leq 12$, and the roll types have either a linear or a concave cost function;

(3) a set of 200 instances proposed by Hemmelmayr et al. [19] where $L_1 \leq 250$, $\sum_{j \in \mathcal{J}} b_j \leq 2000$, $|\mathcal{R}| \leq 7$, and the roll types have either a linear, a concave, or a convex cost function.

We adopt the terminology used by Hemmelmayr et al. [19] to describe the cost function for the roll types, which can always be written as $P_r = kL_r^\alpha$ ($k \in \mathbb{R}, \alpha \in \mathbb{R}^+, r \in \mathcal{R}$). This cost function is said to be linear if $\alpha = 1$, concave if $\alpha < 1$, and convex if $\alpha > 1$. In the tested instances, we have $\alpha \in \{0.5, 1, 1.5\}$.

We report in Table 5 the results obtained by the arcflow formulation introduced by Valério de Carvalho [37], the reflect formulation introduced by Delorme and Iori [13], and our most complete new variant—the reflect formulation with small-roll, extended reflected connection, and downgrade arcs (reflect+sr+erc+d)—for all three configurations V0, V1, and V2.

Table 5: VSCSP results for the three configurations of arcflow and reflect formulations

Dataset	#inst.	arcflow						reflect						reflect+sr+erc+d					
		V0		V1		V2		V0		V1		V2		V0		V1		V2	
		#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)
Belov 1	50	49	138.8	49	139.1	49	139.2	50	65.9	50	10.2	50	10.0	48	162.3	50	27.6	50	12.8
Belov 2	50	49	174.4	49	174.2	49	174.4	50	34.3	49	100.2	50	42.3	50	22.3	49	108.6	49	94.2
Belov 3	50	50	112.4	50	112.5	50	112.5	50	21.8	50	58.9	50	45.3	50	15.3	50	75.2	50	22.0
Belov 4	50	50	93.3	50	93.3	50	93.2	50	34.0	50	63.7	50	26.7	50	20.9	50	89.5	50	44.3
Total (1)	200	198	129.7	198	129.8	198	129.8	200	39.0	199	58.3	200	31.1	198	55.2	199	75.2	199	43.3
Crainic 1	300	300	0.1	300	0.1	300	0.1	300	0.1	300	0.1	300	0.1	300	0.0	300	0.1	300	0.1
Crainic 2	60	60	2.1	60	2.1	60	2.1	56	241.6	60	0.5	60	0.4	52	486.3	60	0.9	60	0.8
Crainic 3	480	480	0.1	480	0.1	480	0.1	419	458.4	480	0.1	480	0.1	418	472.2	480	0.1	480	0.1
Total (2)	840	840	0.3	840	0.3	840	0.3	775	279.2	840	0.1	840	0.1	770	304.6	840	0.1	840	0.1
Hemmelmayr 1	150	147	76.0	147	73.7	147	73.7	146	102.4	150	0.5	150	0.4	147	82.2	150	0.4	150	0.4
Hemmelmayr 2	50	50	0.3	50	0.3	50	0.3	50	0.1	50	0.1	50	0.1	50	0.1	50	0.1	50	0.1
Total (3)	200	197	57.1	197	55.3	197	55.3	196	76.8	200	0.4	200	0.4	197	61.7	200	0.4	200	0.3
Total (1,2,3)	1240	1235	30.3	1235	30.0	1235	30.0	1171	207.8	1239	9.5	1240	5.1	1165	225.2	1239	12.3	1239	7.1

These experiments clearly show that using dummy variables significantly improves the empirical behavior of the reflect formulations, even though these variables do not serve any purpose from a modeling point of view. Further analysis showed that every instance in datasets (2) and (3) that could not be solved to optimality by configuration V0 of a reflect formulation has a concave cost function, and that the solution found by the solver for all such instances except one turned out to be optimal—the solver simply could not raise the lower bound and return a certificate of optimality. In contrast, when one of the configurations of a reflect formulation could not solve an instance of dataset (1) to optimality, it always had the correct lower bound—the solution found by the solver was simply not the optimal one. Since the unsolved instances of dataset (1) were generally not the same across the tested approaches, we believe that randomness plays a role in determining whether an instance of the dataset is solved to optimality.

For the arcflow formulation, using dummy variables does not seem to have any significant impact. To explain this difference, let us recall that the objective function in the arcflow formulation is composed of only $|\mathcal{R}|$ elements—one per roll arc $(L_r, T, 0, r)$—whereas in reflect formulations, the objective function is composed of many more elements—one per reflected arc. Hence, adding these dummy variables simplifies the objective function of the reflect formulations, reducing it to $|\mathcal{R}|$ elements, whereas it does not have the same effect for the arcflow formulation.

Let us also recall that instances of datasets (2) and (3) do not produce large graphs for flow formulations (especially for reflect) since L_1 is relatively small. Hence, solving a node in the B&B tree is very fast for the solver. The issue seems to be that, while an optimal solution is almost always quickly found for these instances, there appears to be a random component as to whether the solver "gets lucky" (e.g., derives the right cut, performs the right branching, or expands the right node) and eventually ends with a certificate of optimality, or instead spends all its computation time exploring the search tree without raising the lower bound. This random component is also evidenced by the inconsistency with which instances remain unsolved across the different models: 44 out of the 73 instances of datasets (2) and (3) that could not be solved by reflect+rar+erc+d V0 could be solved by reflect V0 (and 40 the other way around), indicating that these instances are not structurally difficult.

We therefore postulate that using dummy variables helps the solver improve the dual bound when: (i) the objective function involves non-unit coefficients, which is the case for the VSCSP but not for the SSP and the CSP, and (ii) it significantly reduces the number of elements in the objective function, which is the case for reflect formulations but not for the arcflow formulation in the VSCSP. The tested VSCSP instances did not demonstrate any advantage in assigning higher branching priority to the dummy variables, as the results of configurations V1 and V2 are relatively similar for all models.

We report in Table 6 the results and model size of all tested reflect formulations.

Table 6: Detailed VSCSP results for the tested reflect formulations

Dataset	#inst.	reflect V2				reflect+sr V2				reflect+sr+erc V2				reflect+sr+erc+d V2			
		#opt	T(s)	#var	#cons	#opt	T(s)	#var	#cons	#opt	T(s)	#var	#cons	#opt	T(s)	#var	#cons
Belov 1	50	50	10.0	36.7	4.3	50	10.0	36.7	4.3	50	8.5	35.3	4.3	50	12.8	35.2	4.4
Belov 2	50	50	42.3	44.8	4.4	50	27.9	44.7	4.4	49	87.3	40.4	4.4	49	94.2	40.1	4.7
Belov 3	50	50	45.3	48.5	4.5	50	29.2	48.0	4.5	50	25.7	40.7	4.5	50	22.0	40.2	4.8
Belov 4	50	50	26.7	53.7	4.4	50	36.1	53.0	4.4	50	98.9	41.7	4.4	50	44.3	40.9	4.8
Total (1)	200	200	31.1	45.9	4.4	200	25.8	45.6	4.4	199	55.1	39.5	4.4	199	43.3	39.1	4.7
Crainic 1	300	300	0.1	0.8	0.1	300	0.1	0.8	0.1	300	0.1	0.7	0.1	300	0.1	0.7	0.1
Crainic 2	60	60	0.4	2.9	0.2	60	0.8	2.7	0.2	60	0.8	2.5	0.2	60	0.8	2.5	0.2
Crainic 3	480	480	0.1	0.6	0.1	480	0.1	0.5	0.1	480	0.1	0.5	0.1	480	0.1	0.4	0.1
Total (2)	840	840	0.1	0.8	0.1	840	0.1	0.8	0.1	840	0.1	0.7	0.1	840	0.1	0.7	0.1
Hemmelmayr 1	150	150	0.4	6.2	0.3	150	0.4	5.9	0.3	150	0.3	4.4	0.3	150	0.4	4.2	0.4
Hemmelmayr 2	50	50	0.1	1.5	0.1	50	0.1	1.5	0.1	50	0.1	1.2	0.1	50	0.1	1.2	0.2
Total (3)	200	200	0.4	5.0	0.3	200	0.3	4.8	0.3	200	0.2	3.6	0.3	200	0.3	3.4	0.3
Total (1,2,3)	1240	1240	5.1	8.8	0.8	1240	4.3	8.7	0.8	1239	9.0	7.4	0.8	1239	7.1	7.3	0.9

As expected, we observe a decrease in the average number of variables when using small-roll arcs, a further decrease when additionally using reflected connection arcs, and another decrease when also using downgrade arcs, although the latter comes at the expense of additional constraints. However, this decrease in model size does not translate into empirical improvements for the tested instances. Our hypothesis is that the decrease is not significant enough to outweigh the (previously mentioned) random component for dataset (1), whereas datasets (2) and (3) become very easy once dummy variables are included, so reducing the (already quite small) model size produces barely noticeable effects.

Finally, we compare in Table 7 the results reported in the literature with those obtained after rerunning the flow formulations. Note that, since Delorme and Iori [13] presented the results of their flow formulations in configuration V1, we also report the outcomes of our reruns in configuration V1. For the sake of comparison, we include the best results obtained by one of our approaches, namely those of reflect+sr (this time in configuration V2).

Also for the VSCSP, the improvements in the computational setup are observable, especially for instances of dataset (1), which can now almost all be solved to optimality. A surprising behavior is observed for three instances of dataset (3), which could be solved with the experimental setup from a few years ago, but not with the one used in these experiments. While it could be interesting to identify the reason for this behavior (e.g., a difference in the implementation, a change in the solver parameters used, or a decrease in solver performance on these specific instances), the most important takeaway from these experiments is that existing flow formulations can now solve these three datasets, and that other (more challenging and widely available) instances are needed.

Table 7: Comparison of the VSCSP results with those reported in the literature

Dataset	#inst.	arcflow V1 [13]		arcflow V1 (r)		reflect V1 [13]		reflect V1 (r)		reflect+sr V2 (r)		SOTA [13]	
		#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)
Belov 1	50	38	1230.6	49	139.1	48	258.5	50	10.2	50	10.0	50	6.1
Belov 2	50	34	1679.7	49	174.2	41	874.3	49	100.2	50	27.9	50	4.1
Belov 3	50	31	1673.3	50	112.5	44	786.0	50	58.9	50	29.2	50	1.6
Belov 4	50	27	2009.2	50	93.3	35	1613.1	50	63.7	50	36.1	50	4.0
Total (1)	200	130	1648.2	198	129.8	168	883.0	199	58.3	200	25.8	200	3.9
Crainic 1	300	300	1.1	300	0.1	300	0.3	300	0.1	300	0.1	300	0.3
Crainic 2	60	60	3.4	60	2.1	60	2.2	60	0.5	60	0.8	60	2.5
Crainic 3	480	480	0.6	480	0.1	480	0.2	480	0.1	480	0.1	480	0.2
Total (2)	840	840	1.0	840	0.3	840	0.4	840	0.1	840	0.1	840	0.4
Hemmelmayr 1	150	150	2.7	147	73.7	150	1.2	150	0.5	150	0.4	150	1.1
Hemmelmayr 2	50	50	1.5	50	0.3	50	0.5	50	0.1	50	0.1	50	0.8
Total (3)	200	200	2.4	197	55.3	200	1.0	200	0.4	200	0.3	200	1.0
Total (1,2,3)	1240	1170	266.9	1235	30.0	1208	142.9	1239	9.5	1240	4.3	1240	1.1

7.4 Results for the MKP

For the MKP, we considered the datasets used by Dell'Amico et al. [12], consisting of four sets of 480 instances each. In these instances, $|\mathcal{J}| \leq 100\alpha$, $|\mathcal{R}| \leq 30\alpha$, and the instance-specific average number of items per roll, $\frac{\sum_{j\in\mathcal{J}} b_j}{\sum_{r\in\mathcal{R}} S_r}$, takes value in $\{2, 3, 4, 5, 6, 10\}$. These datasets were generated such that (i) many b_j ($j \in \mathcal{J}$) and S_r ($r \in \mathcal{R}$) values are equal to 1, (ii) the profit p_j of each item type $j \in \mathcal{J}$ is either uncorrelated with, weakly correlated with, strongly correlated with, or equal to its length l_j, and (iii) the supply of rolls is sufficient to cut approximately half of the items. The latter was enforced when determining the roll lengths, which vary for each instance (although L_1 remains of the same order of magnitude, typically a few thousand). Regarding α, it is equal to 1 in the first dataset, 2 in the second, 3 in the third, and 5 in the fourth.

As for the VSCSP, we first report in Table 8 the results obtained by the arcflow and reflect formulations of Dell'Amico et al. [12] and our most complete new variant, reflect+sr+erc+d, for all three configurations.

Table 8: MKP results for the three configurations of arcflow and reflect formulations

Dataset	#inst.	arcflow						reflect						reflect+sr+erc+d					
		V0		V1		V2		V0		V1		V2		V0		V1		V2	
		#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)
30_60	80	80	0.2	80	0.2	80	0.2	80	0.1	80	0.1	80	0.1	80	0.1	80	0.1	80	0.1
15_45	80	80	5.8	80	3.0	80	3.8	79	49.9	80	9.7	80	0.9	80	1.6	80	1.5	80	1.5
12_48	80	75	399.5	79	130.7	80	31.3	73	417.3	79	103.4	79	55.9	78	152.2	80	36.0	80	26.6
15_75	80	52	1371.0	76	682.4	78	384.8	49	1505.8	73	547.4	76	323.5	63	940.0	79	188.9	79	101.5
10_60	80	46	1623.5	72	643.9	78	310.1	51	1401.1	69	741.2	73	521.5	59	1091.1	78	249.3	78	166.4
10_100	80	23	2608.6	32	2503.2	46	2146.2	31	2329.3	59	1595.8	69	1148.7	38	2016.3	65	1288.8	69	997.0
Total (1)	480	356	1001.4	419	660.6	442	479.4	363	950.6	440	499.6	457	341.8	398	700.2	462	294.1	466	215.5

These experiments show that using dummy variables significantly improves the performance of the reflect formulations for the MKP as well. Interestingly, this observation also holds for the arcflow formulation this time. To explain this change, recall that, unlike in the VSCSP, the objective function of the arcflow formulation for the MKP consists—like that of reflect—of many elements: one per item arc. Therefore, using dummy variables reduces the objective function to $|\mathcal{J}|$ elements (one per item type) in both the arcflow and reflect formulations. It is also worth noting that assigning a higher branching priority to the dummy variables is beneficial for all formulations, as already observed by Dell'Amico et al. [12].

We report in Table 9 the results and model size of all tested reflect formulations.

For the MKP as well, we confirm the expected decrease in the average number of variables each time one of the newly introduced modeling features is added, but this time the reduction does translate into a significant improvement in model performance. While using small-roll arcs brings only

Table 9: Detailed MKP results for the tested reflect formulations

Dataset	#inst.	reflect V2				reflect+sr V2				reflect+sr+erc V2				reflect+sr+erc+d V2			
		#opt	T(s)	#var	#cons	#opt	T(s)	#var	#cons	#opt	T(s)	#var	#cons	#opt	T(s)	#var	#cons
30_60	80	80	0.1	3.2	0.6	80	0.1	2.2	0.5	80	0.1	2.2	0.5	80	0.1	2.2	0.5
15_45	80	80	0.9	2.4	0.5	80	1.2	2.3	0.5	80	1.9	2.1	0.5	80	1.5	2.0	0.5
12_48	80	79	55.9	4.2	0.5	80	18.3	4.2	0.5	80	18.3	3.3	0.5	80	26.6	3.1	0.6
15_75	80	76	323.5	17.0	0.8	75	368.7	16.6	0.8	77	179.8	10.8	0.8	79	101.5	9.7	0.9
10_60	80	73	521.5	12.9	0.8	72	575.3	12.8	0.8	78	130.5	9.3	0.8	78	166.4	8.5	0.9
10_100	80	69	1148.7	83.4	1.6	71	1104.1	82.0	1.6	70	974	55.3	1.6	69	997	47.4	1.8
Total (1)	480	457	341.8	20.5	0.8	458	344.6	20.0	0.8	465	217.4	13.8	0.8	466	215.5	12.1	0.9
60_120	80	80	0.7	19.5	0.8	80	0.3	9.8	0.7	80	0.2	8.3	0.7	80	0.2	8.3	0.7
30_90	80	77	246.1	14.6	0.8	76	251.6	12.3	0.8	77	216.1	8.4	0.8	75	342.4	8.3	0.8
24_96	80	79	217.7	21.5	0.7	79	212.6	21.4	0.7	80	57.1	11.8	0.7	80	74.7	9.6	0.9
30_150	80	66	1126.9	79.7	1.0	70	928.4	76.3	1.0	76	548.8	36.9	1.0	73	663.9	27.1	1.1
20_120	80	68	950.5	63.3	1.1	68	922.3	60.6	1.1	76	464.9	33.5	1.1	77	510.5	28.0	1.2
20_200	80	41	2272.5	336.3	1.8	41	2241.1	334.5	1.8	53	1908.7	155.6	1.8	51	1931.6	111.7	2.1
Total (2)	480	411	802.4	89.2	1.0	414	759.4	85.8	1.0	442	532.7	42.4	1.0	436	587.2	32.2	1.1
90_180	80	80	2.2	50.7	1.0	80	0.7	20.7	0.9	80	0.6	17.7	0.9	80	0.6	17.7	0.9
45_135	80	76	392.0	37.7	0.9	77	383.6	29.8	0.9	77	241.8	15.8	0.9	76	274.9	15.6	0.9
36_144	80	73	619.7	56.8	0.9	74	624.8	53.4	0.9	79	292.4	26.6	0.9	78	332.2	19.7	1.0
45_225	80	47	2039.2	201.6	1.2	47	2055.4	184.0	1.2	66	1227.2	80.1	1.2	62	1266.1	51.6	1.3
30_180	80	51	1819.1	151.2	1.2	48	1865.2	143.7	1.2	66	1166.3	67.2	1.2	63	1165.7	46.8	1.4
30_300	80	13	3262.7	809.0	2.1	18	3091.5	751.0	2.1	31	2527.4	318.5	2.1	32	2466.8	222.5	2.4
Total (3)	480	340	1355.8	217.8	1.2	344	1336.9	197.1	1.2	399	909.3	87.6	1.2	391	917.7	62.3	1.3
150_300	80	80	9.3	166.9	1.2	80	2.2	51.6	1.1	80	1.4	38.1	1.1	80	1.4	38.1	1.2
75_225	80	65	1292.6	125.1	1.1	66	1019.2	86.5	1.1	73	726.9	36.2	1.1	74	658.4	35.3	1.1
60_240	80	48	1940.5	183.4	1.1	50	1899.3	160.2	1.1	62	1387.9	65.7	1.1	61	1372.4	44.2	1.2
75_375	80	26	2668.0	648.6	1.6	28	2611.6	491.7	1.5	49	2062.2	203.2	1.5	47	2028.3	128.9	1.7
50_300	80	28	2643.3	464.0	1.4	29	2570.6	423.0	1.4	43	2119.4	164.7	1.4	53	1766.0	96.7	1.6
50_500	80	6	3535.4	2394.3	2.5	9	3398.9	2095.4	2.5	13	3106.7	708.4	2.5	16	3125.1	446.8	2.7
Total (4)	480	253	2014.8	663.7	1.5	262	1917.0	551.4	1.5	320	1567.4	202.7	1.5	331	1491.9	131.7	1.6
Total (1,2,3,4)	1920	1461	1128.7	247.8	1.1	1478	1089.5	213.6	1.1	1626	806.7	86.6	1.1	1624	803.1	59.6	1.2

a marginal gain in both the number of instances solved to optimality and the average number of variables, using extended reflected connection arcs proves to be the most beneficial according to both measures. The impact of downgrade arcs appears more nuanced: they seem advantageous for instances with the highest number of roll types—where the reduction in the average number of variables is most significant—but inconsequential or even detrimental for other instances.

As usual, we finish by comparing in Table 10 the results reported in the literature with those obtained after rerunning the flow formulations. For the sake of comparison, we also include the best results obtained by one of our approaches, namely those of reflect+sr+erc. Because the existing literature [12, 24] used 1200s as a time limit in their experiments, we updated our results to simulate what would have happened under this time limit (i.e., if an instance took longer than 1200s to be solved, we considered the instance unsolved). We also mention that the flow formulations of Dell'Amico et al. [12] include a few MKP-specific improvements, such as an upper bound on the maximum number of items included in the solution and a set of preprocessing techniques (instance reduction, capacity lifting, and item dominance), which are not integrated in our rerun.

Once again, the improvements in the computational setup are evident, especially for large-size instances. However, the state-of-the-art approach—a tailored B&P algorithm proposed by Lalonde et al. [24]—still outperform all the tested models.

7.5 Results for the VSSSP

As the VSSSP has never been studied in the literature, we created new benchmark instances derived from known datasets for related problems. We first experimented with the datasets used for the VSCSP (referred to as dataset (1)), which are directly applicable to the VSSSP, but these turned out to be relatively easy (as was already the case for the VSCSP). Hence, we decided to also experiment with the datasets used for the MKP. To construct a VSSSP instance from an MKP one, we removed the profit of each item type and added a supply and a profit to each roll type. Given a roll type r

Table 10: Comparison of the MKP results with those reported in the literature

Dataset	#inst.	arcflow V2 [12]		arcflow V2 (r)		reflect V2 [12]		reflect V2 (r)		reflect+sr+erc V2 (r)		SOTA [24]	
		#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)
30_60	80	80	0.4	80	0.2	80	0.4	80	0.1	80	0.1	80	0.0
15_45	80	80	13.5	80	3.8	80	3.0	80	0.9	80	1.9	80	2.0
12_48	80	80	70.1	80	31.3	80	14.8	79	25.9	80	18.3	80	1.5
15_75	80	66	397.0	72	261.9	73	238.1	73	190.2	77	89.8	80	1.1
10_60	80	70	390.1	76	234.4	78	156.2	66	279.3	78	70.5	80	0.3
10_100	80	7	1175.3	29	853.9	32	925.9	55	638.4	58	543.6	80	0.0
Total (1)	480	383	341.1	417	230.9	423	223.1	433	189.1	453	120.7	480	0.8
60_120	80	80	3.1	80	0.8	80	3.1	80	0.7	80	0.2	80	0.2
30_90	80	77	175.2	70	218.5	77	136.5	76	143.3	77	126.1	80	46.1
24_96	80	68	376.7	78	157.1	70	282.2	77	177.1	80	57.1	80	5.1
30_150	80	28	939.6	55	600.5	23	965.0	55	587.1	69	356.6	80	4.2
20_120	80	23	992.5	46	708.3	32	866.5	61	534.1	73	306.0	80	0.5
20_200	80	0	1200.0	24	947.2	0	1200.0	28	940.3	31	824.8	80	0.0
Total (2)	480	276	614.7	353	438.7	282	576.0	377	397.1	410	278.5	480	9.4
90_180	80	80	6.2	80	1.7	80	6.2	80	2.2	80	0.6	80	0.5
45_135	80	47	608.0	70	280.7	51	577.5	72	241.5	76	139.4	80	58.3
36_144	80	40	764.4	67	405.6	36	787.9	70	379.5	74	212.3	79	34.7
45_225	80	8	1219.5	39	813.1	10	1164.4	29	852.4	55	657.5	80	19.8
30_180	80	4	1242.5	28	847.6	11	1149.0	40	804.8	54	619.0	80	1.5
30_300	80	0	971.7	14	1066.9	0	217.8	5	1159.2	24	951.0	80	0.0
Total (3)	480	179	802.0	298	569.3	188	650.5	296	573.3	363	430.0	479	19.1
150_300	80	80	12.6	80	3.5	80	13.0	80	9.3	80	1.4	80	2.9
75_225	80	10	1137.5	61	526.5	12	1114.5	47	621.3	64	436.8	60	360.3
60_240	80	6	1166.9	34	819.3	5	1177.2	32	831.6	50	676.1	77	91.8
75_375	80	0	1200.0	20	975.5	0	1200.0	20	976.0	25	886.9	75	102.7
50_300	80	0	1200.0	18	957.1	0	1200.0	19	962.4	29	851.5	79	42.3
50_500	80	0	1200.0	7	1153.2	0	1200.0	1	1197.3	10	1080.9	80	0.0
Total (4)	480	96	986.6	220	739.2	97	986.2	199	766.3	258	655.6	451	100.0
Total (1,2,3,4)	1920	934	686.1	1288	494.5	990	609.0	1305	481.4	1484	371.2	1890	32.3

in an MKP instance with supply S_r and length L_r, the corresponding roll type r' has supply $S_{r'} \in \{3S_r, 10S_r, \infty\}$ and profit $P_{r'} \in \{L_r, 100\sqrt{L_r}, 0.1L_r^{1.5}\}$. The three supply functions represent different levels of heterogeneity in the roll types, whereas the three profit functions mimic those identified by Hemmelmayr et al. [19] for the VSCSP (linear, concave, and convex, respectively). We took MKP dataset (1), which is composed of 480 instances, kept only the instances with uncorrelated item profit (bringing the number of instances to 120), and then applied each combination of roll supply and profit functions, leading to a total of $120 \times 3 \times 3 = 1080$ VSSSP instances (referred to as dataset (2)).

We first present in Table 11 (resp. Table A.13 in Section E.1 of the appendix) the results for (resp. the model size of) all the flow formulations discussed in Section 6, namely:

- arcflow without (arcflow) and with (larcflow) merging of item arcs $(d, d + l_j, j, 0)$ such that $d < L_1$ and $d + l_j \geq L_1$ into a single item arc $(L_1 - l_j, L_1, j, 0)$;
- reflect with forward (reflect+f) and backward (reflect+b) loss arcs;
- reflect with backward loss arcs augmented with the SSP-specific improvements: the adapted reduction procedure alone (rb+red), conversion variables alone (rb+con), and both combined (rb+red+con);
- reflect with backward loss arcs augmented with the VSCSP/MKP-specific improvements: small-roll arcs alone (rb+sr), combined with extended reflected connection arcs (rb+sr+erc), and combined with both extended reflected connection and downgrade arcs (rb+sr+erc+d);
- reflect with backward loss arcs augmented with all SSP-specific and VSCSP/MKP-specific improvements combined (rb+all).

As hinted earlier when motivating the backward-loss-arc version of reflect, the forward-loss-arc variant is not competitive, performing even worse than the arcflow formulations due to its large number of constraints. For the SSP-specific features, the adapted reduction procedure slightly improves

Table 11: VSSSP results for all tested flow formulations in configuration V2

Dataset	#inst.	arcflow		larcflow		reflect+f		reflect+b		rb+red		rb+con		rb+red+con		rb+sr		rb+sr+erc		rb+sr+erc+d		rb+all	
		#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)
Belov 1	50	50	62.3	50	72.8	50	21.7	50	11.3	50	12.6	50	21.3	50	17	50	12.0	50	12.6	50	14.3	50	17.7
Belov 2	50	49	153.0	50	146.2	49	159.3	50	19.9	50	23.8	50	44.3	50	21.1	49	92.2	50	14.9	50	23.3	50	28.2
Belov 3	50	50	80.4	50	138.3	50	52.3	50	37.4	50	36.5	50	23.5	50	29.5	50	74.0	49	91.6	50	27.4	49	92.4
Belov 4	50	50	87.2	50	92.7	50	104.8	50	20.8	50	22.3	50	60.6	50	28.1	50	19.6	50	23.7	50	22.2	50	24.9
Crainic 1	300	300	0.2	300	0.1	300	0.2	300	0.1	300	0.1	300	0.1	300	0.1	300	0.1	300	0.1	300	0.1	300	0.1
Crainic 2	60	59	1.2	60	1.2	60	0.6	60	0.5	60	0.5	60	0.7	60	0.7	60	0.3	60	0.3	60	0.3	60	0.5
Crainic 3	480	480	0.1	480	0.1	480	0.1	480	0.1	480	0.1	480	0.1	480	0.1	480	0.1	480	0.1	480	0.1	480	0.1
Hemmelmayr 1	150	150	1.1	150	0.7	150	0.8	150	2.9	150	0.8	150	0.7	150	0.4	150	11.2	150	0.4	150	0.4	150	0.4
Hemmelmayr 2	50	50	0.4	50	0.3	50	0.3	50	0.2	50	0.2	50	0.3	50	0.3	50	0.2	50	0.2	50	0.1	50	0.2
Total (1)	1240	1238	15.7	1240	18.4	1239	13.9	1240	4.0	1240	4.0	1240	6.2	1240	4.0	1239	9.4	1239	5.9	1240	3.6	1239	6.7
30_60	180	179	28.8	179	50.3	165	387.4	175	124.4	178	62.1	178	65.0	179	52.1	179	23.8	180	18.1	179	23.7	180	10.4
15_45	180	180	3.1	180	6.0	172	192.9	180	7.5	180	3.3	179	26.6	180	21.8	180	10.8	179	21.4	179	21.6	179	21.8
12_48	180	176	132.8	177	118.1	169	377.0	180	35.5	179	37.5	178	70.8	180	30.3	179	43.4	180	47.9	180	52.6	180	44.9
15_75	180	145	1107.0	137	1184.7	121	1554.1	159	756.9	165	600.2	162	648.6	160	599.8	159	763.5	166	630.8	173	467.4	179	178.3
10_60	180	152	919.3	153	911.9	151	977.6	176	332.2	178	276.5	178	227.4	177	198.5	176	334.1	179	244.2	177	264.2	178	162.7
10_100	180	52	3116.2	55	3008.9	73	2755.5	104	2141.2	105	2122.0	97	2206.6	126	1842.8	105	2171.9	109	2038.6	101	2139.1	131	1459.2
Total (2)	1080	884	884.5	881	880.0	851	1040.7	974	566.3	985	516.9	972	540.8	1002	457.5	978	557.9	993	500.2	989	494.8	1027	312.9
Total (1,2)	2320	2122	420.2	2121	419.5	2090	491.9	2214	265.8	2225	242.8	2212	255.1	2242	215.1	2217	264.7	2232	236.0	2229	232.3	2266	149.2

performance, while adding conversion variables on top improves it further. For the VSCSP/MKP-specific features, small-roll arcs bring only a marginal improvement, extended reflected connection arcs provide a more significant benefit, and downgrade arcs cause a slight performance decrease. Interestingly, the version of reflect that combines all features outperforms both the version using only the SSP-specific features and the version using only the VSCSP/MKP-specific features. This can be explained by the fact that the reduction in the average number of variables achieved by one set of features is additive to that achieved by the other.

We display in Table 12 the results obtained by each of the tested flow formulations when the instances are aggregated by the supply/profit function used for the roll types. For this table only, we also report the results obtained by reflect+b in configurations V0 and V1.

Table 12: VSSSP results for Dataset (2) aggregated by supply and profit functions

Supply/profit function	#inst.	arcflow		larcflow		reflect+f		reflect+b						rb+red		rb+con		rb+red+con		rb+sr		rb+sr+erc		rb+sr+erc+d		rb+all	
		V2		V2		V2		V0		V1		V2		V2		V2		V2		V2		V2		V2		V2	
		#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)	#opt	T(s)
$S_{r'} = 3S_r$	360	288	972.8	289	949.2	277	1168.4	206	1602.0	298	915.1	319	680.2	322	634.1	321	633.9	332	512.1	318	674.0	325	592.5	329	551.7	343	316.3
$S_{r'} = 10S_r$	360	296	871.8	292	895.7	283	1033.9	231	1360.1	314	689.8	323	578.0	330	498.9	319	566.1	331	482.6	327	565.6	328	519.3	325	519.9	341	312.2
$S_{r'} = \infty$	360	300	809.1	300	795.1	291	920.0	243	1214.6	323	530.3	332	440.7	333	417.8	332	422.5	339	377.9	333	434.2	340	388.7	335	412.7	343	310.1
$P_{r'} = L_r$	360	291	826.7	287	862.2	322	641.6	309	596.0	316	614.9	317	627.7	327	525.5	316	593.3	326	518.8	318	613.8	324	555.5	327	528.5	324	505.1
$P_{r'} = 100\sqrt{L_r}$	360	279	1006.9	281	989.4	240	1413.0	210	1532.4	303	781.5	324	534.4	327	491.9	317	594.9	327	535.8	326	517.8	328	490.9	326	505.2	347	266.5
$P_{r'} = 0.1L_r^{1.5}$	360	314	820.1	313	788.3	289	1067.6	161	2048.2	316	738.7	333	536.8	331	533.4	339	434.4	349	318.0	334	542.2	341	454.1	336	450.7	356	167.1

A striking observation concerns the respective outcomes of the three tested configurations of reflect+b depending on the profit function of the roll types: for concave and convex functions, not using dummy variables (configuration V0) is hugely detrimental, and not giving these variables higher branching priority (configuration V1) is also detrimental, but to a lesser extent. For linear profits however, the configuration used seems to have only a marginal impact (if any) on performance overall: the number of optimal solutions found is slightly higher in configurations V1 and V2, but we also observe a slight decrease in the average computation time in configuration V0.

Focusing now on the results for all formulations (in configuration V2), it appears that convex-profit instances are easier to solve to optimality for all models except reflect+f, while concave-profit instances usually come second. It is worth noting that convex and concave profits tend to have opposite effects on the structure of an optimal solution. Convex profits tend to promote the use of large rolls (instead of, say, two small rolls of the same total size), meaning that fewer roll arcs are typically required in an optimal solution. In the network, this might lead to many active arcs, but only a relatively small flow on each of them. Convex profits may also enable some preprocessing: for example, since a set of items that forms two rolls with threshold $\frac{L_1}{2}$ will always also be able to form one roll with threshold L_1, it is clear that at most one roll with threshold $\frac{L_1}{2}$ can be included in an optimal solution as long as the supply of rolls with threshold L_1 is not exhausted. Conversely, concave profits tend to favor the use of small rolls, which leads to a larger number of rolls being required. This could mean that a smaller set of arcs is active, but each carries a larger flow. Regarding the roll supply function, instances with very large supplies tend to be slightly easier to solve than those with low supplies, although the difference is barely noticeable for rb+all, our best approach. This is probably because the combinatorial hardness is slightly reduced when infinite supply is available: one highly effective roll type (typically the one with the highest profit-to-threshold ratio) can be used extensively, usually resulting in an optimal solution that uses only a few roll types. This phenomenon has also been observed, for example, in the knapsack problem, where the unbounded variant is generally regarded as easier to solve than the 0-1 variant [2, 22].

8 Conclusions

We studied flow formulations for four extensions of the one-dimensional Cutting Stock Problem (CSP): the Skiving Stock Problem (SSP), the Variable-Sized Cutting Stock Problem (VSCSP), the Multiple Knapsack Problem (MKP), and the Variable-Sized Skiving Stock Problem (VSSSP). We proposed both SSP-specific and VSCSP/MKP-specific features aimed at reducing the size of the reflect formulation and showed that these features can all be combined to solve the VSSSP. In particular, we demonstrated that the reduction procedure stating that no reflected arc whose tail is higher than its head needs to be created for the CSP can be adapted to the SSP. We also presented alternative ways—requiring fewer variables than the original reflect formulation—to represent patterns when rolls have variable sizes.

Our work highlights the rich variety of modeling possibilities one encounters when designing a flow formulation for a cutting and packing problem, and while formulations already exist for many of these problems, there is still room for improvements in some cases. Another striking observation is that the empirical behavior of a model does not always align perfectly with theoretical expectations. For example, the use of dummy variables, which serve no purpose from a purely modeling perspective, proved essential for solving certain types of VSCSP and MKP instances. In contrast, the use of conversion variables, which significantly reduce the number of variables and constraints when used together with the adapted reduction procedure, was, on average, detrimental when solving SSP instances, although it proved useful for VSSSP instances.

Another relevant aspect we emphasized is the general improvement of a model's performance over time. The literature often mentions advances in computing power but rarely highlights or measures differences in solver performance. For example, we showed that significant improvements are observable for previously introduced flow formulations across various datasets for all the studied problems. Still, it is important to note that, for most of these problems, solving a flow formulation (even one that uses our newly introduced features) with a general-purpose integer linear programming solver yields results that are inferior to those produced by the most competitive algorithms in the literature. That being said, the latter are often highly specialized and not easily adaptable (if they are even publicly available), whereas flow formulations are generally considered more accessible while still delivering competitive results for most datasets, striking a balance between performance and simplicity.

Acknowledgments

This work used the Dutch national e-infrastructure with the support of the SURF Cooperative using grant no. EINF-12272.

References

[1] M. Barkel and M. Delorme. Arcflow formulations and constraint generation frameworks for the two bar charts packing problem. *INFORMS Journal on Computing*, 35:475–494, 2023.

[2] H. Becker and L.S. Buriol. An empirical analysis of exact algorithms for the unbounded knapsack problem. *European Journal of Operational Research*, 277:84–99, 2019.

[3] G. Belov and G. Scheithauer. A cutting plane algorithm for the one-dimensional cutting stock problem with multiple stock lengths. *European Journal of Operational Research*, 141:274–294, 2002.

[4] F. Brandão and J.P. Pedroso. Bin packing and related problems: General arc-flow formulation with graph compression. *Computers & Operations Research*, 69:56–67, 2016.

[5] H. Cambazard and B. O'Sullivan. Propagating the bin packing constraint using linear programming. In *Principles and Practice of Constraint Programming–CP 2010*, volume 6308 of *Lecture Notes in Computer Science*, pages 129–136. Springer-Verlag, 2010.

[6] J. Côté and M. Iori. The meet-in-the-middle principle for cutting and packing problems. *INFORMS Journal on Computing*, 30:646–661, 2018.

[7] T.G. Crainic, G. Perboli, W. Rei, and R. Tadei. Efficient lower bounds and heuristics for the variable cost and size bin packing problem. *Computers & Operations Research*, 38:1474–1482, 2011.

[8] R. F. da Silva and R. Schouery. Solving cutting stock problems via an extended ryan-foster branching scheme and fast column generation. Technical report, 2023. available at `https://arxiv.org/pdf/2308.03595`.

[9] M. De Cauwer, D. Mehta, and B. O'Sullivan. The temporal bin packing problem: an application to workload management in data centres. In *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 157–164, 2016.

[10] V.L. de Lima, C. Alves, F. Clautiaux, M. Iori, and J.M. Valério de Carvalho. Arc flow formulations based on dynamic programming: Theoretical foundations and applications. *European Journal of Operational Research*, 296: 3–21, 2022.

[11] V.L. de Lima, M. Iori, and F.K. Miyazawa. Exact solution of network flow models with strong relaxations. *Mathematical Programming*, 197:813–846, 2023.

[12] M. Dell'Amico, M. Delorme, M. Iori, and S. Martello. Mathematical models and decomposition methods for the multiple knapsack problem. *European Journal of Operational Research*, 274:886–899, 2019.

[13] M. Delorme and M. Iori. Enhanced pseudo-polynomial formulations for bin packing and cutting stock problems. *INFORMS Journal on Computing*, 32:101–119, 2020.

[14] M. Delorme, M. Iori, and S. Martello. Bin packing and cutting stock problems: Mathematical models and exact algorithms. *European Journal of Operational Research*, 255:1–20, 2016.

[15] H. Dyckhoff. A new linear programming approach to the cutting stock problem. *Operations Research*, 29:1092–1104, 1981.

[16] F. Furini, E. Malaguti, and D. Thomopulos. Modeling two-dimensional guillotine cutting problems via integer programming. *INFORMS Journal on Computing*, 28:736–751, 2016.

[17] P.C. Gilmore and R.E. Gomory. A linear programming approach to the cutting-stock problem. *Operations research*, 9:849–859, 1961.

[18] T. Gschwind and S. Irnich. Dual inequalities for stabilized column generation revisited. *INFORMS Journal on Computing*, 28:175–194, 2016.

[19] V. Hemmelmayr, V. Schmid, and C. Blum. Variable neighbourhood search for the variable sized bin packing problem. *Computers & Operations Research*, 39:1097–1108, 2012.

[20] M.P. Johnson, C. Rennick, and E. Zak. Case studies from industry: Skiving addition to the cutting stock problem in the paper industry. *SIAM Review*, 39:472–483, 1997.

[21] L.V. Kantorovich. Mathematical methods of organizing and planning production. *Management Science, English translation of a 1939 paper written in Russian*, 6:366–422, 1960.

[22] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack problems.* Springer Berlin, Heidelberg, 2004.

[23] L. Korbacher, S. Irnich, J. Martinovic, and N. Strasdat. Solving the skiving stock problem by a combination of stabilized column generation and the reflect arc-flow model. *Discrete Applied Mathematics*, 334:145–162, 2023.

[24] O. Lalonde, J.-F. Côté, and B. Gendron. A branch-and-price algorithm for the multiple knapsack problem. *INFORMS Journal on Computing*, 34:3134–3150, 2022.

[25] S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations.* John Wiley & Sons, Chichester, 1990.

[26] S. Martello, D. Pisinger, and P. Toth. Dynamic programming and strong bounds for the 0-1 knapsack problem. *Management Science*, 45:414–424, 1999.

[27] J. Martinovic and G. Scheithauer. Integer linear programming models for the skiving stock problem. *European Journal of Operational Research*, 251:356–368, 2016.

[28] J. Martinovic and N. Strasdat. The overflowing bin packing problem: Theoretical results and a new flow formulation. Technical report, MATH-NM-01-2024, Technische Universität Dresden, 2024. URL `https://optimization-online.org/?p=27340`.

[29] J. Martinovic, E. Jorswieck, and G. Scheithauer. The skiving stock problem and its application to cognitive radio networks. *IFAC-PapersOnLine*, 49:99–104, 2016.

[30] J. Martinovic, E. Jorswieck, G. Scheithauer, and A. Fischer. Integer linear programming formulations for cognitive radio resource allocation. *IEEE Wireless Communications Letters*, 6:494–497, 2017.

[31] J. Martinovic, G. Scheithauer, and J.M. Valério de Carvalho. A comparative study of the arcflow model and the one-cut model for one-dimensional cutting stock problems. *European Journal of Operational Research*, 266: 458–471, 2018.

[32] J. Martinovic, M. Delorme, M. Iori, G. Scheithauer, and N. Strasdat. Improved flow-based formulations for the skiving stock problem. *Computers & Operations Research*, 113:104770, 2020.

[33] S. Mehrani, C. Cardonha, and D. Bergman. Models and algorithms for the bin-packing problem with minimum color fragmentation. *INFORMS Journal on Computing*, 34:1070–1085, 2022.

[34] C. Nitsche, G. Scheithauer, and J. Terno. Tighter relaxations for the cutting stock problem. *European Journal of Operational Research*, 112:654–663, 1999.

[35] A. Pessoa, R. Sadykov, E. Uchoa, and F. Vanderbeck. A generic exact solver for vehicle routing and related problems. *Mathematical Programming*, 183:483–523, 2020.

[36] J.M. Valério de Carvalho. Exact solution of bin-packing problems using column generation and branch-and-bound. *Annals of Operations Research*, 86:629–659, 1999.

[37] J.M. Valério De Carvalho. LP models for bin packing and cutting stock problems. *European Journal of Operational Research*, 141:253–273, 2002.

[38] L. Wei, Z. Luo, R. Baldacci, and A. Lim. A new branch-and-price-and-cut algorithm for one-dimensional bin-packing problems. *INFORMS Journal on Computing*, 32:428–443, 2020.

[39] S. Zheng, N. Xie, Q. Wu, and C. Liu. Novel mathematical formulations for parallel-batching processing machine scheduling problems. *Computers & Operations Research*, 173:106859, 2025.

APPENDIX

A Supplementary material for the CSP

A.1 Graph construction algorithm for the CSP arcflow model

Algorithm A.2 Graph construction for the arcflow model

Input: $\mathcal{J}$ and L

$\mathcal{V} \leftarrow \{0\}$; $\mathcal{A} \leftarrow \emptyset$ ▷ initialize a graph with only vertex 0
for $j \in \mathcal{J}$ **do** ▷ for every item type
    for $k = 1, \dots, b_j$ **do** ▷ for every unit of demand
        $\mathcal{V}' \leftarrow \mathcal{V}$ ▷ make a copy of the vertex set
        for $d \in \mathcal{V}'$ **do** ▷ for every possible tail
            if $d + l_j \leq L$ **then** ▷ if the possible head does not exceed the roll length
                $\mathcal{V} \leftarrow \mathcal{V} \cup \{d + l_j\}$ ▷ add the head to the original vertex set
                $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, d + l_j, j)\}$ ▷ create the item arc
$\mathcal{V} \leftarrow \mathcal{V} \cup \{L\}$ ▷ add L to the vertex set as it may not have been done before
for $d \in \mathcal{V} \setminus \{0, L\}$ **do** ▷ for every vertex in the set besides 0 and L
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, e, 0)\}$ where $e = \min\{v \in \mathcal{V} : v > d\}\}$ ▷ create the loss arc

Output: $\mathcal{G} = (\mathcal{V}, \mathcal{A})$

We highlight that (i) $\mathcal{A}$ and $\mathcal{V}$ are also sets in the algorithmic sense; therefore, duplicate entries are automatically removed, and (ii) the order in which the item types are considered influences the number of arcs in $\mathcal{A}$; hence, as done in the literature, we preliminarily sort the item types in decreasing order of their length. These comments also apply to all graph construction algorithms.

A.2 Graph construction algorithm for the CSP reflect model

Algorithm A.3 Graph construction for the reflect model

Input: $\mathcal{J}$ and L

1: $\mathcal{V} \leftarrow \{0\}$; $\mathcal{A} \leftarrow \emptyset$ ▷ initialize a graph with only vertex 0
2: $\mathcal{T} \leftarrow \{0\}$ ▷ keeps track of the possible tails
3: **for** $j \in \mathcal{J}$ **do** ▷ for every item type
4:     **for** $k = 1, \dots, b_j$ **do** ▷ for every unit of demand
5:         $\mathcal{T}' \leftarrow \mathcal{T}$ ▷ make a copy of the possible tails
6:         **for** $d \in \mathcal{T}'$ **do** ▷ for every possible tail
7:             **if** $d + l_j \leq \frac{L}{2}$ **then** ▷ if this is a standard item arc
8:                 $\mathcal{V} \leftarrow \mathcal{V} \cup \{d + l_j\}$ ▷ add the head to the original vertex set
9:                 $\mathcal{T} \leftarrow \mathcal{T} \cup \{d + l_j\}$ ▷ add the head to the possible tails
10:                 $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, d + l_j, j, 0)\}$ ▷ create the standard item arc
11:             **else** ▷ if this is a potential reflected item arc
12:                 **if** $d \leq L - (d + l_j)$ **then** ▷ and it is not eliminated by the reduction procedure
13:                     $\mathcal{V} \leftarrow \mathcal{V} \cup \{L - (d + l_j)\}$ ▷ add the head to the original vertex set
14:                     $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, L - (d + l_j), j, 1)\}$ ▷ create the reflected item arc
15: $\mathcal{V} \leftarrow \mathcal{V} \cup \{\frac{L}{2}\}$ ▷ add $\frac{L}{2}$ to the vertex set as it may have not been done before
16: **for** $d \in \mathcal{V} \setminus \{\frac{L}{2}\}$ **do** ▷ for every vertex in the set besides $\frac{L}{2}$
17:     $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, e, 0, 0)\}$ where $e = \min\{v \in \mathcal{V} : v > d\}\}$ ▷ create the standard loss arc
18: $\mathcal{A} \leftarrow \mathcal{A} \cup \{(\frac{L}{2}, \frac{L}{2}, 0, 1)\}$ ▷ create the reflected connection arc

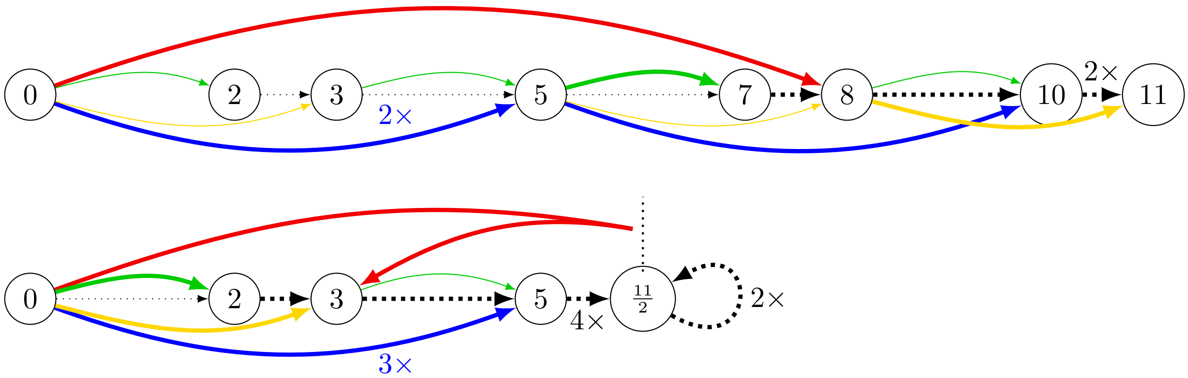
Output: $\mathcal{G} = (\mathcal{V}, \mathcal{A})$

We highlight that the case where L is an odd number can be handled either by doubling every item length and L (this scaling does not increase the size of $\mathcal{V}$ or $\mathcal{A}$), or by explicitly introducing a dummy (fractional) node $\frac{L}{2}$, which is connected only to standard loss arcs (Steps 15-17) and to the reflected connection arc (Step 18).

A.3 Example for the arcflow and reflect graph constructions for the CSP

Consider a CSP instance composed of four item types with lengths $l_1 = 8$, $l_2 = 5$, $l_3 = 3$, and $l_4 = 2$, and corresponding demands $b_1 = 1$, $b_2 = 3$, $b_3 = 1$, and $b_4 = 1$, with a roll length of $L = 11$. The corresponding arcflow (resp. reflect) graph obtained after applying Algorithm A.2 (resp. Algorithm A.3) is shown at the top (resp. bottom) of Figure A.4. Item arcs are displayed as colored solid lines—red, blue, yellow, and green for item type 1, 2, 3, and 4, respectively—whereas the other arcs are shown as black dotted lines. Reflected arcs are visually reflected. The arcs selected in the optimal solution are highlighted with increased thickness. Unless indicated otherwise by a marker (e.g., "2×"), each thick arc is selected once in the solution.

Figure A.4: Arcflow and reflect graphs corresponding to the example, along with their solutions



An optimal solution uses three rolls, which are cut as follows: (i) one copy of item 1 and one copy of item 3 in (what we arbitrarily considered as) the first roll, (ii) two copies of item 2 in the second roll, leaving one unit of trim loss, and (iii) one copy of item 2 and one copy of item 4 in the third roll, leaving four units of trim loss. The first roll corresponds to the path formed by arcs $(0, 8, 1)$ and $(8, 11, 3)$ in the arcflow graph, and two subpaths that collide at vertex 3 in the reflect graph—one formed by arc $(0, 3, 1, 1)$, and the other by arc $(0, 3, 3, 0)$. The second roll corresponds to the path formed by arcs $(0, 5, 2)$, $(5, 10, 2)$, and $(10, 11, 0)$ in the arcflow graph, and two subpaths that collide at vertex $\frac{11}{2}$ in the reflect graph—one formed by arcs $(0, 5, 2, 0)$, $(5, \frac{11}{2}, 0, 0)$, and $(\frac{11}{2}, \frac{11}{2}, 0, 1)$ and the other by arcs $(0, 5, 2, 0)$ and $(5, \frac{11}{2}, 0, 0)$. The third roll corresponds to the path formed by arcs $(0, 5, 2)$, $(5, 7, 4)$, $(7, 8, 0)$, $(8, 10, 0)$, and $(10, 11, 0)$ in the arcflow graph, and two subpaths that collide at vertex $\frac{11}{2}$ in the reflect graph—one formed by arcs $(0, 5, 2, 0)$, $(5, \frac{11}{2}, 0, 0)$, and $(\frac{11}{2}, \frac{11}{2}, 0, 1)$ and the other by arcs $(0, 2, 4, 0)$, $(2, 3, 0, 0)$, $(3, 5, 0, 0)$, $(5, \frac{11}{2}, 0, 0)$.

B Supplementary material for the VSCSP and the MKP

B.1 ILP model for the VSCSP/MKP arcflow formulation

$$\max \quad \sum_{\substack{a \in \mathcal{A} \\ r(a) \geq 1}} P_{r(a)} f_a + \sum_{\substack{a \in \mathcal{A} \\ j(a) \geq 1}} p_{j(a)} f_a \tag{A.30}$$

$$\text{s.t.} \quad \sum_{\substack{a \in \mathcal{A} \\ h(a) = v}} f_a = \sum_{\substack{a \in \mathcal{A} \\ t(a) = v}} f_a \qquad v \in \mathcal{V} \setminus \{0, T\}, \tag{A.31}$$

$$\sum_{\substack{a \in \mathcal{A} \\ t(a)=0}} f_a = \sum_{\substack{a \in \mathcal{A} \\ h(a)=T}} f_a, \tag{A.32}$$

$$b_j \leq \sum_{\substack{a \in \mathcal{A} \\ j(a)=j}} f_a \leq s_j \qquad j \in \mathcal{J}, \tag{A.33}$$

$$\sum_{\substack{a \in \mathcal{A} \\ r(a)=r}} f_a \leq S_r \qquad r \in \mathcal{R}, \tag{A.34}$$

$$f_a \in \mathbb{N}_0 \qquad a \in \mathcal{A}, \tag{A.35}$$

B.2 ILP model for the VSCSP/MKP reflect formulation

$$\max \quad \sum_{\substack{a \in \mathcal{A} \\ r(a)\geq 1}} P_{r(a)} f_a + \sum_{\substack{a \in \mathcal{A} \\ j(a)\geq 1}} p_{j(a)} f_a \tag{A.36}$$

$$\text{s.t.} \quad \sum_{\substack{a \in \mathcal{A} \\ h(a)=v,\, r(a)=0}} f_a = \sum_{\substack{a \in \mathcal{A} \\ t(a)=v}} f_a + \sum_{\substack{a \in \mathcal{A} \\ h(a)=v,\, r(a)\geq 1}} f_a \qquad v \in \mathcal{V} \setminus \{0\}, \tag{A.37}$$

$$\sum_{\substack{a \in \mathcal{A} \\ t(a)=0}} f_a + \sum_{\substack{a \in \mathcal{A} \\ h(a)=0}} f_a = 2 \sum_{\substack{a \in \mathcal{A} \\ r(a)\geq 1}} f_a, \tag{A.38}$$

$$b_j \leq \sum_{\substack{a \in \mathcal{A} \\ j(a)=j}} f_a \leq s_j \qquad j \in \mathcal{J}, \tag{A.39}$$

$$\sum_{\substack{a \in \mathcal{A} \\ r(a)=r}} f_a \leq S_r \qquad r \in \mathcal{R}, \tag{A.40}$$

$$f_a \in \mathbb{N}_0 \qquad a \in \mathcal{A}. \tag{A.41}$$

B.3 Graph construction algorithm for the VSCSP/MKP reflect model using the three newly introduced arc types

Algorithm A.4 Graph construction for the reflect model using the three newly introduced arc types for the VSCSP and MKP

Input: $\mathcal{J}$ and $\mathcal{R}$

$\mathcal{V} \leftarrow \{0\}$; $\mathcal{A} \leftarrow \emptyset$ ▷ initialize a graph with only vertex 0
$\mathcal{T} \leftarrow \{0\}$ ▷ keeps track of the possible tails
for $j \in \mathcal{J}$ **do** ▷ for every item type
  for $k = 1, \ldots, b_j$ **do** ▷ for every unit of demand
    $\mathcal{T}' \leftarrow \mathcal{T}$ ▷ make a copy of the possible tails
    for $d \in \mathcal{T}'$ **do** ▷ for every possible tail
      if $d + l_j \leq \frac{L_1}{2}$ **then** ▷ if this is a standard item arc
        $\mathcal{V} \leftarrow \mathcal{V} \cup \{d + l_j\}$ ▷ add the head to the original vertex set
        $\mathcal{T} \leftarrow \mathcal{T} \cup \{d + l_j\}$ ▷ add the head to the possible tails
        $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, d + l_j, j, 0)\}$ ▷ create the standard item arc
        for $r \in \mathcal{R} \setminus \{1\}$ **do** ▷ for each roll type except the longest
          if $L_r > \frac{L_1}{2}$ **then** ▷ if this is not a small roll
            if $d < \frac{L_r}{2}$ and $d + l_j > \frac{L_r}{2}$ **then** ▷ and it is a potential reflected item arc
              if $d \leq L_r - (d + l_j)$ **then** ▷ and it is not eliminated by the reduction procedure
                $\mathcal{V} \leftarrow \mathcal{V} \cup \{L_r - (d + l_j)\}$ ▷ add the head to the original vertex set
                $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d + l_j, L_r - (d + l_j), 0, r)\}$ ▷ create the extended refl. connection arc
      else ▷ create the reflected item arc for the largest roll
        if $d \leq L_1 - (d + l_j)$ **then** ▷ but only if it is not eliminated by the reduction procedure
          $\mathcal{V} \leftarrow \mathcal{V} \cup \{L_1 - (d + l_j)\}$ ▷ add the head to the original vertex set
          $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, L_1 - (d + l_j), j, 1)\}$ ▷ create the reflected item arc
          for $r \in \mathcal{R} \setminus \{1\}$ **do** ▷ for each roll type except the longest
            if $L_r > \frac{L_1}{2}$ **then** ▷ if this is not a small roll
              if $d < \frac{L_r}{2}$ and $d + l_j > \frac{L_r}{2}$ **then** ▷ and it is a potential reflected item arc
                if $d \leq L_r - (d + l_j)$ **then** ▷ and it is not eliminated by the reduction procedure
                  $\mathcal{V} \leftarrow \mathcal{V} \cup \{L_r - (d + l_j)\}$ ▷ add the head to the original vertex set
                  $\mathcal{A} \leftarrow \mathcal{A} \cup \{(L_1 - (d + l_j), L_r - (d + l_j), -1, r)\}$ ▷ create the downgrade arc
for $r \in \mathcal{R}$ **do** ▷ for each roll type
  if $L_r \leq \frac{L_1}{2}$ **then** ▷ if this is a small roll
    $\mathcal{V} \leftarrow \mathcal{V} \cup \{L_r\}$ ▷ add L_r to the vertex set as it may have not been done before
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{(0, L_r, 0, r)\}$ ▷ create the small-roll reflected arc
  else
    $\mathcal{V} \leftarrow \mathcal{V} \cup \{\frac{L_r}{2}\}$ ▷ add $\frac{L_r}{2}$ to the vertex set as it may have not been done before
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{(\frac{L_r}{2}, \frac{L_r}{2}, 0, r)\}$ ▷ create the reflected connection arc
for $d \in \mathcal{V} \setminus \{\frac{L_1}{2}\}$ **do** ▷ for every vertex in the set besides $\frac{L_1}{2}$
  $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, e, 0, 0)\}$ where $e = \min\{v \in \mathcal{V} : v > d\}\}$ ▷ create the standard loss arc

Output: $\mathcal{G} = (\mathcal{V}, \mathcal{A})$

C Supplementary material for the SSP

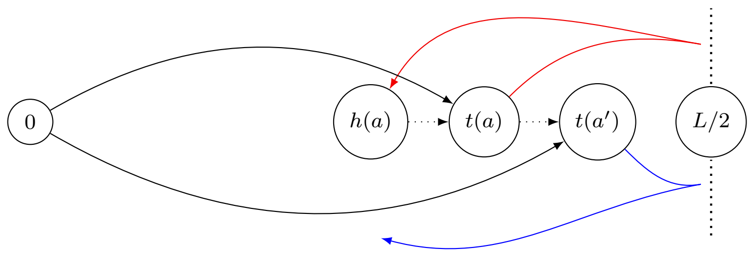
C.1 Proof of Proposition 1

Let us consider a feasible SSP pattern formed by two subpaths with the given properties. We distinguish two cases, either $t(a) < t(a')$ or $t(a) \geq t(a')$, but detail only the first case, as the reasoning for the second is identical. In the first case, where $h(a) < t(a) < t(a')$, the item $j(a')$ represented by the reflected item arc $(t(a'), h(a'), j(a'), 1)$ is not required to meet the threshold condition. Instead, after using the reflected item arc $(t(a), h(a), j(a), 1)$, we can move from $h(a)$ to $t(a')$ via forward loss arcs (or from $t(a')$ to $h(a)$ via backward loss arcs if those are used instead), so that the two subpaths collide at $t(a')$ (or at $h(a)$ if backward loss arcs are used). The pattern composed of these new subpaths is still feasible, because its length is given by

$$t(a') + \left[t(a) + \left(\frac{L}{2} - t(a)\right) + \left(\frac{L}{2} - h(a)\right)\right] = L - h(a) + t(a') > L.$$

An example using forward loss arcs is illustrated in Figure A.5, where two subpaths using only standard item arcs (in black) arrive at $t(a)$ and $t(a')$. For simplicity, the vertices between the starting and ending points on these paths are not shown. One of these subpaths is continued by a reflected item arc $(t(a), h(a), j(a), 1)$ (in red), and the other by $(t(a'), h(a'), j(a'), 1)$ (in blue). Note that the head $h(a')$ of the blue arc is not drawn explicitly, because its position among the remaining nodes is not clear from the assumptions.

Figure A.5: Schematic of the situation relevant to the first case of the proof.



C.2 Proof of Theorem 1

We start by noting the correctness of the claim for $k = 0$. Indeed, any pattern in this case can also be considered a CSP pattern (for the same instance), and it was shown by Delorme and Iori [13] that $t(a) \leq h(a)$ is always possible for the associated reflected item arc.

Hence, $k \geq 1$ can be assumed from now on, meaning that the length $L' = L + k$ of the considered minimal SSP pattern is strictly larger than the threshold. In addition, the pattern must contain more than one item since $l_j \leq L$ holds for all $j \in \mathcal{J}$. Any such pattern falls into one of the two cases highlighted in the theorem:

(I) The pattern can be represented by two subpaths in which each subpath contains a reflected item arc. In both subpaths, forward loss arcs are added to the node representing the head of the reflected item arc until the node $\frac{L}{2}$ is reached. Then, a negative flow on the reflected connection arc $(\frac{L}{2}, \frac{L}{2}, 0, 1)$ is used to merge the two subpaths.

(II) The pattern can be represented by two subpaths in which only one subpath contains a reflected item arc. The other subpath consists of standard item arcs only. In that case, the reflected connection arc $(\frac{L}{2}, \frac{L}{2}, 0, 1)$ is not needed.

Let us start with the first case, meaning that there is a decomposition of the pattern according to (I). Thanks to the minimality of the pattern and Proposition 1, we know that at least one of these reflected item arcs, say $a' = (t(a'), h(a'), j(a'), 1)$, satisfies $t(a') \leq h(a')$. For the sake of contradiction, let us assume that the other reflected item arc, say $a = (t(a), h(a), j(a), 1)$, is such that $t(a) - h(a) > k$. Then, the length of the subpath containing that arc is given by

$$\frac{L}{2} + \left(\frac{L}{2} - t(a)\right) + (t(a) - h(a)) > \frac{L}{2} + (t(a) - h(a)) > \frac{L}{2} + k.$$

Since the second subpath also uses a reflected item arc, its length is strictly larger than $\frac{L}{2}$. Hence, the items represented in the two subpaths would form a pattern of total length larger than $L' = L + k$, leading to a contradiction. As a consequence, the first case of the theorem is dealt with.

Let us now consider a pattern that falls exclusively into case (II), and let us refer to its items by their lengths $l'_1 \geq l'_2 \geq \ldots \geq l'_q$. Note that, for the sake of simplicity, items of equal length will be treated separately. We again distinguish two cases:

- Let us assume that there is a subset $I' \subset \{1, \ldots, q\}$ with $\sum_{i \in I'} l'_i = \frac{L}{2}$. Then, the associated standard item arcs (arranged in non-increasing order according to their item lengths) form the first subpath. Since $k \geq 1$, the second subpath must contain a reflected item arc, say $(t(a), h(a), j(a), 1)$. For the sake of contradiction, let us assume that $t(a) - h(a) > k$. Then, the length of the second subpath is given by

$$\frac{L}{2} + \left(\frac{L}{2} - t(a)\right) + (t(a) - h(a)) > \frac{L}{2} + (t(a) - h(a)) > \frac{L}{2} + k.$$

  As in case (I), the contradiction now follows by merging the two subpaths and summing their lengths.

- Let us assume that there is no subset $I' \subset \{1, \ldots, q\}$ with $\sum_{i \in I'} l'_i = \frac{L}{2}$. We now show how to construct the two subpaths so that only one of them contains a reflected item arc, and this arc satisfies $t(a) \leq h(a) + k$. We initialize two (originally empty) subsets I_1 and I_2 of $\{1, \ldots, q\}$, process the items one at a time in the given order, and partition them based on the following rules with $j \in \{1, \ldots, q\}$ acting as the currently considered item:

  – If

$$\sum_{i \in I_1} l'_i + l'_j < \frac{L}{2} \qquad \text{or} \qquad \sum_{i \in I_1} l'_i - \frac{L}{2} + \frac{l'_j}{2} \leq \frac{k}{2} \tag{A.42}$$

  holds, then item j is added to set I_1.

  – Otherwise, i.e., if

$$\sum_{i \in I_1} l'_i + l'_j > \frac{L}{2} \qquad \text{and} \qquad \sum_{i \in I_1} l'_i - \frac{L}{2} + \frac{l'_j}{2} > \frac{k}{2},$$

  then item j is added to set I_2.

  According to (A.42), item j is added to I_1 if this item would lead to an additional standard item arc in the associated subpath (first option) or if this item would lead to a reflected item arc satisfying $t(a) - h(a) \leq k$. Indeed, note that $\sum_{i \in I_1} l'_i + l'_j > \frac{L}{2}$ leads to a reflected item arc with

$$t(a) = \sum_{i \in I_1} l'_i \qquad \text{and} \qquad h(a) = L - \sum_{i \in I_1} l'_i - l'_j,$$

  meaning that $t(a) - h(a) \leq k$ holds if and only if

$$\sum_{i \in I_1} l'_i - \left(L - \sum_{i \in I_1} l'_i - l'_j\right) = 2 \cdot \sum_{i \in I_1} l'_i - L + l'_j \leq k,$$

  which is precisely the second option in (A.42), scaled by a factor of two.

After applying this partitioning algorithm, there must be some item, say j, that is responsible for a reflected item arc, either in set I_1 or in set I_2. Since we assumed that the pattern falls exclusively into case (II), it is impossible for the total length of both item sets to exceed $\frac{L}{2}$. If j belongs to I_1, then it immediately follows that the associated reflected item arc satisfies $t(a) - h(a) \leq k$, because j fulfilled the second option in (A.42). Otherwise, j belongs to I_2 and hence must have violated the second option in (A.42). Let us backtrack to the moment just before j was included in I_2 and, for the sake of contradiction, assume that inserting j into I_2 would create a reflected item arc violating the condition $t(a) - h(a) \leq k$, meaning that

$$\sum_{i \in I_2} l'_i - \frac{L}{2} + \frac{l'_j}{2} > \frac{k}{2}$$

using the same calculation as above. Combining this observation with the fact that j violated the second option in (A.42), we obtain

$$\left(\sum_{i \in I_1 \cup I_2} l'_i + l'_j \right) - L = \left(\sum_{i \in I_1} l'_i - \frac{L}{2} + \frac{l'_j}{2} \right) + \left(\sum_{i \in I_2} l'_i - \frac{L}{2} + \frac{l'_j}{2} \right) > \frac{k}{2} + \frac{k}{2} = k,$$

which contradicts the fact that $\sum_{i=1}^{q} l_i = L' = L + k$. Therefore, the reflected item arc originating from I_2 must satisfy the claim, and the proof is complete.

C.3 Graph construction algorithm for the SSP reflect model using the newly introduced procedures

Algorithm A.5 Graph construction for the reflect model using the newly introduced procedures for the SSP

Input: $\mathcal{J}$ and L

$\mathcal{V} \leftarrow \{0\}$; $\mathcal{A} \leftarrow \emptyset$ ▷ initialize a graph with only vertex 0
$\mathcal{T} \leftarrow \{0\}$ ▷ keeps track of the possible tails
for $j \in \mathcal{J}$ **do** ▷ for every item type
  for $k = 1, \ldots, b_j$ **do** ▷ for every unit of demand
    $\mathcal{T}' \leftarrow \mathcal{T}$ ▷ make a copy of the possible tails
    for $d \in \mathcal{T}'$ **do** ▷ for every possible tail
      if $d + l_j \leq \frac{L}{2}$ **then** ▷ if this is a standard item arc
        $\mathcal{V} \leftarrow \mathcal{V} \cup \{d + l_j\}$ ▷ add the head to the original vertex set
        $\mathcal{T} \leftarrow \mathcal{T} \cup \{d + l_j\}$ ▷ add the head to the possible tails
        $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, d + l_j, j, 0)\}$ ▷ create the standard item arc
      else ▷ if this is a reflected item arc
        if $d \leq L - (d + l_j)$ **then** ▷ and it is of the form $d \leq e$
          $\mathcal{V} \leftarrow \mathcal{V} \cup \{L - (d + l_j)\}$ ▷ add the head to the original vertex set
          $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, L - (d + l_j), j, 1)\}$ ▷ create the reflected item arc
        else ▷ if it is of the form $d > e$, we may add arc $(d, d, j, 1)$
          if $j = |\mathcal{J}|$ or $d < L - (d + l_{j+1})$ **then** ▷ but only if $(d, d, j+1, 1)$ will not be created
            $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, d, j, 1)\}$ ▷ create the reflected item arc
  $d \leftarrow \max\{0, \frac{L}{2} - l_j\}$ ▷ Compute the origin of arc $(\max\{0, \frac{L}{2} - l_j\}, \frac{L}{2}, j, 0)$
  if $d \neq 0$ or $j = |\mathcal{J}|$ or $L - l_{j+1} > 0$ **then** ▷ but only add it if the same arc for $j + 1$ will not be created
    $\mathcal{V} \leftarrow \mathcal{V} \cup \{d\}$ ▷ add d to the original vertex set as it may have not been done before
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{(d, \frac{L}{2}, j, 0)\}$ ▷ create the standard item arc
$\mathcal{V} \leftarrow \mathcal{V} \cup \{\frac{L}{2}\}$ ▷ add $\frac{L}{2}$ to the vertex set as it may have not been done before
for $d \in \mathcal{V} \setminus \{0, \frac{L}{2}\}$ **do** ▷ for every vertex in the set besides 0 and $\frac{L}{2}$
  $\mathcal{A} \leftarrow \mathcal{A} \cup \{(e, d, 0, 0)\}$ where $e = \min\{v \in \mathcal{V} : v > d\}\}$ ▷ create the standard loss arc
$\mathcal{A} \leftarrow \mathcal{A} \cup \{(\frac{L}{2}, \frac{L}{2}, 0, 1)\}$ ▷ create the reflected connection arc

Output: $\mathcal{G} = (\mathcal{V}, \mathcal{A})$

C.4 ILP model for the SSP reflect formulation using the newly introduced procedures

$$\max \sum_{\substack{a\in\mathcal{A}\\ r(a)=1}} f_a \tag{A.43}$$

$$\text{s.t.} \quad \sum_{\substack{a\in\mathcal{A}\\ h(a)=v,\, r(a)=0}} f_a = \sum_{\substack{a\in\mathcal{A}\\ t(a)=v}} f_a + \sum_{\substack{a\in\mathcal{A}\\ h(a)=v,\, r(a)=1}} f_a \qquad v \in \mathcal{V}\setminus\{0\}, \tag{A.44}$$

$$\sum_{\substack{a\in\mathcal{A}\\ t(a)=0}} f_a + \sum_{\substack{a\in\mathcal{A}\\ h(a)=0}} f_a = 2 \sum_{\substack{a\in\mathcal{A}\\ r(a)=1}} f_a, \tag{A.45}$$

$$\sum_{\substack{a\in\mathcal{A}\\ j(a)=j}} f_a + \tau_j \le b_j \qquad j = 1, \tag{A.46}$$

$$\sum_{\substack{a\in\mathcal{A}\\ j(a)=j}} f_a + \tau_j \le b_j + \tau_{j-1} \qquad j \in \mathcal{J}\setminus\{1, |\mathcal{J}|\}, \tag{A.47}$$

$$\sum_{\substack{a\in\mathcal{A}\\ j(a)=j}} f_a \le b_j + \tau_{j-1} \qquad j = |\mathcal{J}|, \tag{A.48}$$

$$f_a \in \mathbb{N}_0 \qquad a \in \mathcal{A}, \tag{A.49}$$

$$\tau_j \in \mathbb{N}_0 \qquad j \in \mathcal{J}\setminus\{|\mathcal{J}|\}. \tag{A.50}$$

D Supplementary material for the VSSSP

D.1 Point of attention regarding downgrade arcs when combining the newly introduced pattern representations for the VSSSP

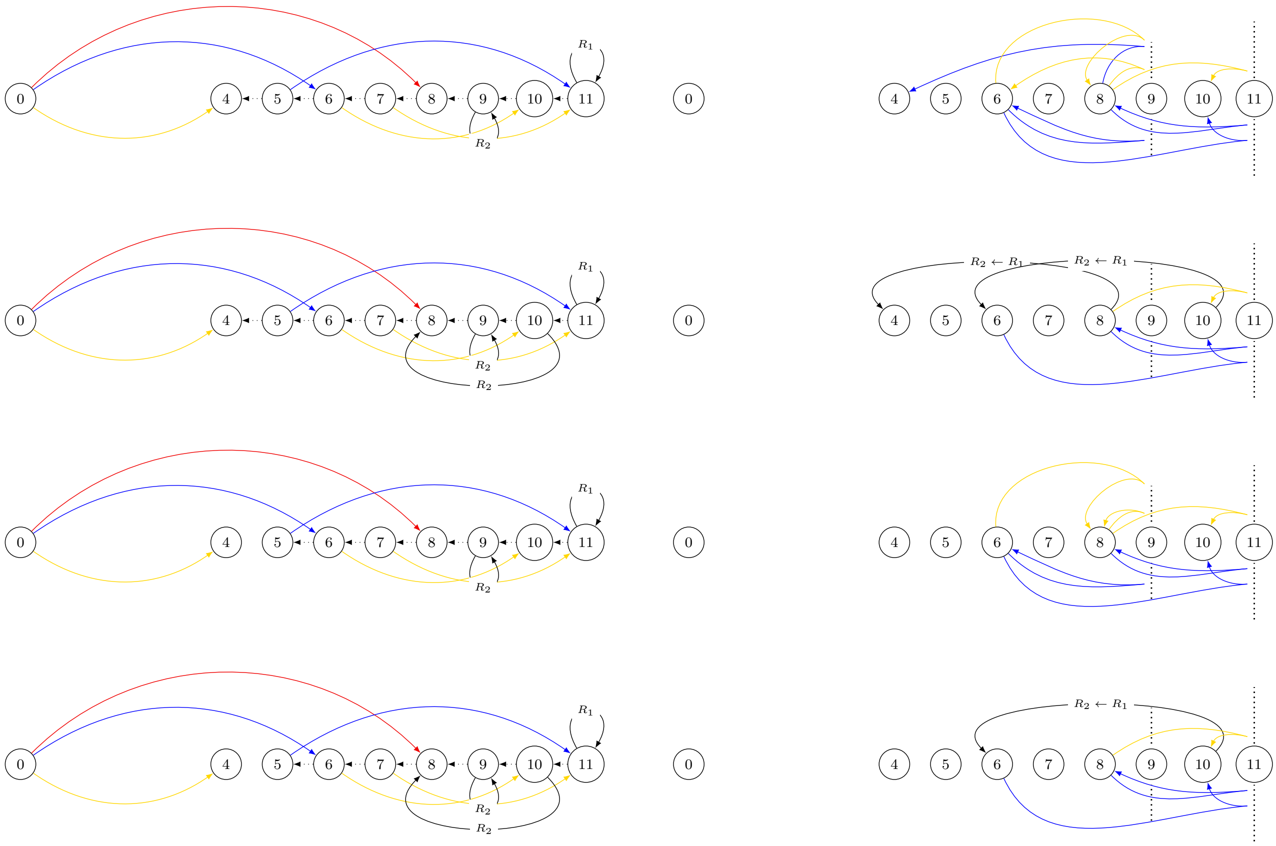
Let us consider a VSSSP instance composed of three item types with lengths $l_1 = 8$, $l_2 = 6$, and $l_3 = 4$ and corresponding demands $b_1 = 1$, $b_2 = 2$, and $b_3 = 1$, with two roll types with thresholds $L_1 = 22$ and $L_2 = 18$. The same four reflect graphs are shown in Figure A.6.

Comparing the first two graphs, reflected item arc $(8, 4, 2, 2)$ in the first graph is replaced by $(8, 4, -1, 2)$, reflected item arcs $(6, 6, 2, 2)$ and $(8, 6, 3, 2)$ are replaced by the same downgrade arc $(10, 6, -1, 2)$, and reflected item arc $(6, 8, 3, 2)$ is replaced by extended reflected connection arcs $(10, 8, 0, 2)$.

Comparing the first and the third graphs, reflected item arcs $(8, 4, 2, 2)$ and $(8, 6, 3, 2)$ in the first graph become $(8, 8, 2, 2)$, and $(8, 8, 3, 2)$, respectively, of which only the arc for the smallest item type $(8, 8, 3, 2)$, is kept because these two arcs are associated with the same roll type and share the same head and tail. Note that reversed loss arcs terminate at vertex $v = \min\{6, 5\}$, 6 being the lowest vertex that is the head of a reflected item arc, here $(6, 6, 2, 2)$, and 5 being the lowest tail of one of the newly added standard item arcs, here $(5, 11, 2, 0)$.

Comparing the third and the fourth graphs, reflected item arc $(6, 8, 3, 2)$ is replaced by extended reflected connection arcs $(10, 8, 0, 2)$, reflected item arc $(6, 6, 2, 2)$ in the third graph is replaced by downgrade arc $(10, 6, -1, 2)$, and reflected item arc $(8, 8, 3, 2)$ should, in theory, be replaced by downgrade arc $(10, 8, -1, 2)$: indeed, taking successively $(8, 10, 3, 1)$ and $(10, 8, -1, 2)$ is conceptually equivalent to taking $(8, 8, 3, 2)$. However, since there is already a downgrade arc for roll type 2 starting from vertex 10, $(10, 6, -1, 2)$, we do not create $(10, 8, -1, 2)$. From an algorithmic perspective, this

Figure A.6: Four reflect graphs corresponding to the example



means that downgrade arcs are not created immediately (as in Step 26 of Algorithm A.4). Instead, we keep track of the minimum head associated with each combination of tail and roll type, and generate the corresponding downgrade arcs only after all item arcs have been created.

E Supplementary material for the computational experiments

E.1 VSSSP size of all tested flow formulations

Table A.13: VSSSP size of all tested flow formulations

Dataset	#inst.	arcflow		larcflow		reflect+f		reflect+b		rb+red		rb+con		rb+red+con		rb+sr		rb+sr+erc		rb+sr+erc+d		rb+all	
		#var	#cons	#var	#cons	#var	#cons	#var	#cons	#var	#cons	#var	#cons	#var	#cons	#var	#cons	#var	#cons	#var	#cons	#var	#cons
Belov 1	50	184.5	9.3	137.9	9.3	48.8	11.7	45.9	4.4	45.8	4.3	46.0	4.4	37.8	4.3	45.8	4.4	42.9	4.4	42.2	5.0	35.9	4.4
Belov 2	50	193.0	9.4	146.1	9.4	68.0	17.5	59.6	4.6	59.5	4.4	59.7	4.6	46.6	4.4	59.3	4.6	50.9	4.6	48.6	5.6	40.9	4.7
Belov 3	50	192.8	9.5	144.6	9.5	86.5	25.5	70.3	4.6	70.1	4.5	70.4	4.6	51.1	4.5	69.1	4.6	54.2	4.6	50.5	6.0	41.1	4.9
Belov 4	50	194.1	9.4	144.8	9.4	107.8	33.2	83.9	4.7	83.6	4.4	84.0	4.7	57.2	4.4	82.4	4.7	60.2	4.7	53.6	6.3	41.9	4.9
Crainic 1	300	3.1	0.2	1.9	0.2	1.7	0.3	1.6	0.1	1.6	0.1	1.7	0.1	0.9	0.1	1.6	0.1	1.3	0.1	1.1	0.2	0.7	0.1
Crainic 2	60	5.2	0.3	5.1	0.3	3.3	0.3	3.3	0.2	3.3	0.2	3.3	0.2	2.9	0.2	2.9	0.2	2.5	0.2	2.5	0.2	2.5	0.2
Crainic 3	480	1.2	0.1	1.0	0.1	0.9	0.2	0.8	0.1	0.8	0.1	0.8	0.1	0.6	0.1	0.7	0.1	0.6	0.1	0.5	0.1	0.5	0.1
Hemmelmayr 1	150	21.6	0.4	11.5	0.4	12.4	0.9	12.2	0.3	12.2	0.3	12.4	0.3	6.5	0.3	11.3	0.3	8.5	0.3	6.3	0.4	4.5	0.4
Hemmelmayr 2	50	4.6	0.2	3.3	0.2	2.5	0.3	2.5	0.1	2.5	0.1	2.5	0.1	1.6	0.1	2.5	0.1	2.0	0.1	1.8	0.2	1.3	0.2
Total (1)	1240	35.1	1.7	25.7	1.7	15.1	3.8	12.9	0.9	12.9	0.8	13.0	0.9	9.2	0.8	12.6	0.8	10.2	0.8	9.3	1.1	7.5	0.9
30_60	180	11.9	1.0	12.3	1.1	13.1	8.1	6.3	0.7	6.3	0.7	6.4	0.7	3.9	0.7	3.7	0.6	3.5	0.6	3.5	0.6	2.6	0.6
15_45	180	7.3	0.9	8.1	1.0	8.3	5.7	3.7	0.6	3.7	0.5	3.8	0.6	2.6	0.5	3.5	0.6	3.1	0.6	3.0	0.7	2.1	0.6
12_48	180	10.1	1.0	10.4	1.0	11.4	5.7	7.0	0.6	6.9	0.5	7.0	0.6	4.5	0.5	6.9	0.6	5.4	0.6	4.5	0.8	3.1	0.6
15_75	180	35.0	1.5	26.2	1.5	38.5	9.3	30.9	0.9	30.9	0.8	31.0	0.9	17.4	0.8	29.8	0.9	19.4	0.9	13.9	1.3	9.4	0.9
10_60	180	33.4	1.7	25.2	1.7	28.0	7.5	22.3	0.9	22.2	0.9	22.3	0.9	13.6	0.9	21.7	0.9	15.9	0.9	12.0	1.4	8.6	1.0
10_100	180	162.5	3.1	133.2	3.1	145.9	9.3	139.8	1.6	139.6	1.6	139.9	1.6	82.3	1.6	134.8	1.6	87.7	1.6	57.2	2.2	46.0	1.9
Total (2)	1080	43.4	1.6	35.9	1.6	40.9	7.6	35.0	0.9	34.9	0.8	35.1	0.9	20.7	0.8	33.4	0.9	22.5	0.9	15.7	1.2	12.0	0.9
Total (1,2)	2320	38.9	1.6	30.5	1.6	27.1	5.6	23.2	0.9	23.1	0.8	23.3	0.9	14.6	0.8	22.3	0.9	15.9	0.9	12.3	1.1	9.6	0.9