The one-dimensional cutting stock problem with sequence dependent cut losses

Michele Garraffa and Fabio Salassa

Politecnico di Torino, DAUIN, Corso Duca degli Abruzzi, 24, 10129 Torino, Italy

Wim Vancroonenburg, Greet Vanden Berghe, and Tony Wauters

KU Leuven, Department of Computer Science, CODeS & iMinds-ITEC,
Gebroeders De Smetstraat 1, 9000 Gent, Belgium

April 28, 2014

Abstract

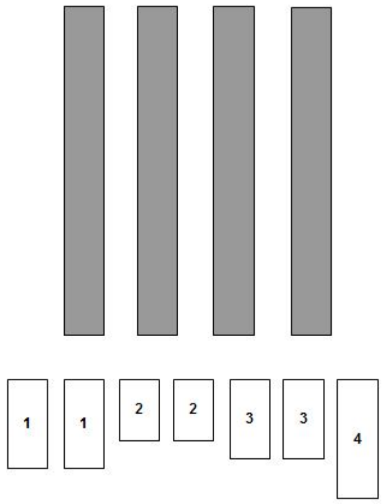
The paper presents a new generalisation of the one-dimensional cutting stock problem (1D-CSP) that considers cut losses that depend on the items' cutting sequence. It is shown that this generalisation can still be solved approximately by standard 1D-CSP approaches. Furthermore, a pattern-based heuristic (denoted HSD) is presented that specifically considers sequence dependent cut losses (SDCL). A computational study shows that whenever some variability in SDCL occurs consideration of SDCL in the HSD heuristic is beneficial. Finally, two case studies illustrate the relevance of this new generalisation.

Keywords: Cutting stock; sequence dependent cut losses; integer programming; pattern based heuristic

1 Introduction

Cutting stock problems arise in many different industries such as in textile, glass, steel, wood and paper. To reduce operating cost, companies strive to minimize waste of stock material when cutting stock down to customer orders. A great deal of research effort has thus focused on developing effective ways for improving operations. The present paper focusses on the one-dimensional cutting stock problem (1D-CSP). Many papers refer to material waste for a 1D-CSP as material that is not used in the cutting patterns (leftovers). However, this is not the only material loss that occurs in practical contexts. Another type of material loss is intrinsically due to the process of cutting the material (by use of a blade, laser, etc.). The overall loss due to cuts is usually negligible, thus it is normally not taken into account. Nevertheless, cut losses cannot always be ignored in practical cases and may even depend on the item order.

We generalize the 1D-CSP to include *sequence dependent cut losses* (SDCL) that may occur between any pair of adjacent items or at the start and end of a cutting pattern. This generalisation allows to further improve efficiency and reduce cut losses that may occur in some situations. The 1D-CSP with SDCL can be approximately solved by any 1D-CSP or 1D bin packing (1D-BPP) approach. This paper investigates the conditions that make it beneficial to consider the SDCL nature of the problem. To this end, we present a heuristic approach specifically considering SDCL, and compare it with a 1D-BPP approach on a range of generated instances, with varying characteristics. Finally, we also describe two applications of the 1D-CSP with SDCL and validate our approach.



i	1	2	3	4
l_i	4.5	3.1	4	6
d_i	2	2	2	1

(a) Length and demand for each item

$$\begin{bmatrix} 0 & 2.4 & 2.5 & 1.9 & 2.6 \\ 1 & 2.5 & 2.5 & 1.5 & 2.7 \\ 1.3 & 1 & 3 & 2.8 & 3 \\ 3.5 & 0.5 & 0.2 & 2.5 & 2.7 \\ 2.2 & 2.8 & 1.4 & 3 & 2.5 \end{bmatrix}$$

(b) Cut losses matrix

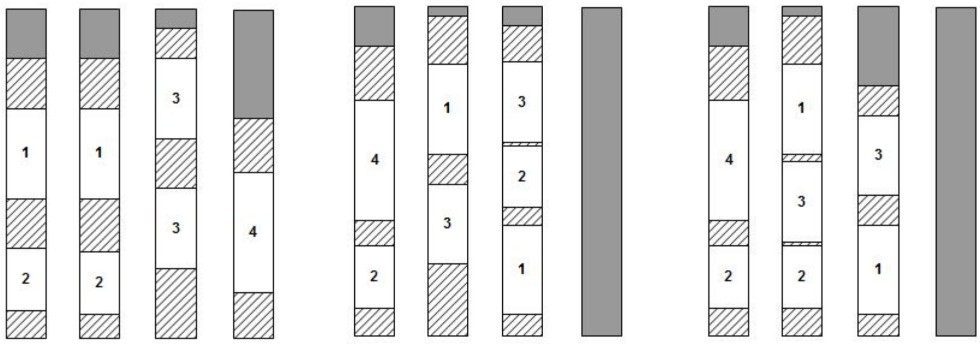
Figure 1: Example of an input instance.

2 Problem formulation

Consider the well known 1D-CSP: a set of items I, each $i \in I$ having length l_i and demand d_i, is to be cut from an unbounded set of larger stock items, each of length L (with $l_i < L \; \forall i \in I$). A feasible cutting pattern p is a subset of items $i \in I$ with multiplicity (denoted a_i) for which the total length $\sum_{i \in p} a_i \cdot l_i$ is at most L. The objective is to find a minimum set of feasible cutting patterns that cover each item's demand.

The 1D-CSP with SDCL generalizes the problem to include sequence dependent cut losses between items: between each pair of adjacent items i and j in a cutting pattern, the presence of an additional *cut loss* c_{ij} should be taken into account. A cutting pattern is then not only determined by the included items and their multiplicity, but also by the sequence of the items within the pattern. Consequently, feasible patterns are cutting patterns for which the sum of item lengths (with multiplicity) and the sum of the SDCL c_{ij} between adjacent items is smaller than L. Finally, the problem also considers start and end cut losses, c_{0i} and c_{i0}. These occur at the start (resp. end) of the pattern before the first (resp. after the last) item i is cut and should also be considered in order to fit within L.

To illustrate the problem, consider the instance shown in Fig. 1 with input data listed in Tables 1(a) and 1(b). The cut losses c_{ij} are of the same order of magnitude as the smallest item and show a very high variability. Fig. 2(a)-2(c) show three feasible solutions for this instance. The example clearly illustrates that changing the order in which the items are cut determines a different amount of material loss due to the cut losses and leftovers. As a result, the solution in Fig. 2(b) requires fewer stock material than the solution in Fig. 2(a). Several solutions may require an equal amount of stock material. For example, Fig. 2(c) shows another solution requiring three cutting patterns. The solution in Fig. 2(c) is more suitable for more industrial purposes, as the leftover in the last pattern may be reusable. Larger leftovers are more likely to be reusable, allowing for further operating cost reductions. We therefore introduce a secondary objective, namely maximization of the sum of the squared leftovers produced by the solution, targeting reusability of the leftover material. The square function is desirable because it allows distinction based on the leftover sizes as shown above. This objective was developed in analogy with Fleszar and Hindi (2002), where maximization of the sum of square loads on a bin packing problem allows to find the optimal space utilization.



(a) First solution, 4 cutting patterns used.

(b) Second solution, 3 cutting patterns used.

(c) Third solution, 3 cutting patterns used and larger leftovers.

Figure 2: Three solutions to the example shown in Fig. 1. The cut losses are drawn with a dashed background, while the leftovers are coloured gray.

2.1 Mathematical modelling

One possible (well known) integer linear programming formulation for the 1D-CSP with SDCL is based on the selection of a subset of patterns from the set of all possible feasible cutting patterns P, that covers each item's demand. A cutting pattern p of n_p items is defined as a sequence $(i_1, i_2, \dots, i_{n_p})$ with $i_1, i_2, \dots i_{n_p} \in I$. Feasible cutting patterns are patterns $(i_1, i_2, \dots, i_{n_p})$ for which the following inequality holds:

$$l_p = \sum_{j=1}^{n_p} l_i + \sum_{j=1}^{n_p} c_{i_j i_{j+1}} + c_{0 i_1} + c_{i_{n_p} 0} \leq L \tag{1}$$

with l_p denoting the length of the pattern p, including SDCL.

The following model minimizes the number of patterns used, and then maximizes the sum of the squared leftovers. Let variables x_p ($p \in P$) denote the number of occurrences of the pattern p in the cutting plan. Let $r_p \in \mathbb{R}_+$ denote the leftover of pattern p (i.e $r_p = L - l_p$), and $a_{ip} \in \mathbb{N}$ the multiplicity of the item i in the pattern p. The integer model is then defined as:

$$\min \sum_{p \in P} (M - r_p^2) x_p \tag{2}$$

s.t. :

$$\sum_{p \in P} a_{ip} x_p = d_i \qquad \forall i \in I \tag{3}$$

$$x_p \in \mathbb{N} \qquad \forall p \in P \tag{4}$$

where $M \in \mathbb{R}_+$ is a suitably large constant.

2.2 Related work and similar problems

According to Dyckhoff's (1990) typology, the 1D-CSP with SDCL can be categorised as 1/V/I/R. It is a one-dimensional problem with an unlimited supply of objects of identical size and a set of items to be cut. The problem can also be classified as a Single Stock Size Cutting Stock Problem (SSSCSP) according to the typology introduced in Wäscher et al. (2007). In both cases, however, the classification is not pure since the addition of sequence dependent waste should be included.

The literature on this type of problems is vast, starting as early as the 1960's with the seminal work of Gilmore and Gomory (1963). Many different models and approaches to the 1D-CSP have been studied since then, and many extensions to the problem definition include more practical considerations. Consideration of multiple stock lengths (e.g. Holthaus, 2002), sequencing and minimization of different cutting patterns to avoid machine setup costs (e.g. Armbruster, 2002; Yanasse and Limeira, 2006; Mobasher and Ekici, 2013), consideration of reusing leftovers (e.g. Trkman and Gradisar, 2007; Cui and Yang, 2010) have been of particular interest.

The inclusion of ordering significance within patterns has also been reported. Lewis et al. (2011) studied the truss cutting problem (TCP), a problem originating from the roofing industry that has strong connections to this work. Profiles of equal width having trapezoidal shapes, have to be cut from wooden boards with the aim of minimizing area waste. Lewis et al. show that the TCP is a special case of 2D-CSP and can actually be solved with a 1D packing approach. Sequence dependent cut losses (inter-item wastage) in this setting are specifically due to the shape of the items that may fit better together in certain arrangements. Given that the items have equal width, Lewis et al. solve the problem as a 1D grouping/packing problem using an algorithmic framework for bin packing problems. Feasibility of a grouping/packing for the TCP is determined by solving some form of a sequencing problem on a losses matrix. In the present contribution the concept of SDCL is more abstract, and we do not restrict losses to the geometric considerations applicable to the TCP. In the same paper, the possibility of including two different orientations for each item is investigated. This, however, cannot be modelled by 1D-CSP with SDCL. Another fundamental difference with our work is that we include considerations about the size of the leftovers in the objective. The 1D-CSP with SDCL cannot be solved directly by applying the procedures of Lewis et al. (2011). Finally, the algorithmic framework of Lewis et al. focusses on 1D-BPP problems. In our work, we specifically model items to have a certain demand, typically larger than one. Regardless, we compare this paper's approach and the work of Lewis et al. on a restricted formulation of the TCP, disallowing orientation changes and disregarding leftover considerations, in Section 5.2.

The 1D-CSP with SDCL also presents interesting analogies with other combinatorial optimization problems. It can be modelled as a distance constrained vehicle routing problem (DCVRP or DVRP). The three index formulation by Laporte (1992) denoted VRP4 can be adapted to model 1D-CSP with SDCL, ignoring the capacity constraints on the vehicles. However, if the second objective function (sum of squared leftovers) is to be considered in the DCVRP model, it becomes nonlinear: since leftover sizes are variable depending on the patterns' sequence, the resulting objective function is quadratic. In this vehicle routing context, the stock material and the items of the 1D-CSP with SDCL are represented respectively by the vehicles and the cities (duplicated to cover the demand of each item) of the DVRP, while the dummy item 0 corresponds to the depot in vehicle routing. The cut losses matrix C is replaced by an analogous matrix C' indicating the distances between items (cities), increased by (half of) the corresponding item lengths (i.e. $c'_{ij} = \frac{l_i+l_j}{2} + c_{ij}$).

Moreover the problem can be modelled as a slight variation of the Multiple Travelling Salesman Problem (*mTSP*). Following the description by Bektas (2006), the variations relevant for this work are related to the number of salesmen and to the so called *special restrictions*. The presented problem also shares similarities with the parallel machine scheduling problem with sequence dependent setup times (see e.g. Lopes and de Carvalho (2007)). Sequence dependent setup times basically model the same idea as SDCL in a scheduling context. Main differences lie with the fact that for the parallel machine scheduling problem, the number of machines ($\sim$ corresponding to cutting patterns) is bounded, but no limit on the usage time for each machine is defined. Typically, the objective is to optimize some measure of throughput (e.g. total makespan), rather than minimizing the number of machines subject to a finite maximum makespan (i.e. corresponding to the stock material length).

2.3 An approximation by 1D-CSP

Under one specific assumption, the 1D-CSP with SDCL can also be solved as a standard 1D-CSP or 1D-BPP. Let $\overline{c_{ij}}$ denote the maximum cut loss in the cut loss matrix c_{ij}. If $l_i + \overline{c_{ij}} \leq L - \overline{c_{ij}}$ $\forall i \in I$ holds, then a 1D-CSP with SDCL instance can be converted to a standard 1D-CSP instance by setting $l_i^* := l_i + \overline{c_{ij}}$ $\forall i \in I$ and $L^* := L - \overline{c_{ij}}$. Any cutting pattern or packing found by a 1D-CSP (or 1D-BPP) approach on this converted instance is feasible in any sequence. Consequently, the solution is also feasible for the 1D-CSP with SDCL. An important question addressed in Section 4 considers the conditions where an SDCL approach is better than a standard 1D-CSP approach.

3 Algorithms

We present a heuristic approach to the 1D-CSP with SDCL. Starting from an exact enumerative pattern based approach, presented in Section 3.1, a heuristic approach (denoted HSD) is developed that overcomes the shortcomings of the exact approach. This heuristic approach is described in Section 3.2.

3.1 An exact enumerative pattern based approach

A general MILP solver was not able to efficiently solve the three index model (Laporte, 1992) due to the subtour elimination constraints (SEC) and the item duplication required to cover demand. The following approach is therefore based on the solution of the model (2) - (4) which is the classical set cover (SC) formulation where r_p are pre-computed constants. Solving this model requires the computation of the set P formed by all efficient patterns. Efficient patterns are those for which the sequence of items within a pattern is optimal, i.e. that the total material loss incurred by the items and the cut losses is minimized. We refer to this optimization as *inner optimization*. The set P includes all patterns that can be generated using the given set of items, having a total length smaller than L, and for which the inner optimization is optimal. Patterns for which the inner optimization of the pattern is not optimal, can be safely ignored due to consideration of the second objective. The pseudocode of this approach is presented in Algorithm 1.

Algorithm 1 An enumerative approach

```
Require: I, c_ij, L, d
  P ← ∅
  for all s ∈ S(I) do                              ▷ For all item subsets s
    if LengthLowerBound(s) < L then
      p ← TSPOPT(s,c_ij)                           ▷ Find optimal pattern for subset s
      if Length(p) < L then
        P ← P ∪ {p}
      end if
    end if
  end for
  Solution ← SC(I,P,d)
  return Solution
```

The algorithm consists of two phases: generating all efficient patterns and solving model (2) - (4) to obtain the final solution. The pattern generation phase iterates over all possible subsets $S(I)$ (with repetition of items i if $d_i > 1$) to check if they form feasible and efficient patterns; in which case they are added to the pattern set P.

Checking for feasibility and efficiency requires solving a TSP problem on the cut losses matrix, restricted to the items (with multiplicity) in the considered pattern. If the optimal tour length (the minimal losses for the considered items), increased by the items' lengths, is lower than L, then the pattern is feasible. Clearly not all subsets of items need to be checked. A simple lower bound on the pattern length of the best permutation (**LengthLowerBound**) can be obtained by summing only the items' lenghts, without considering SDCL (or by considering that each loss is equal to the lowest value in the matrix C). The TSP optimization is only necessary when this lower bound does not exceed L.

Obviously, the main drawbacks of this algorithm are the memory needed for representing all the patterns, and the high computation time required to solve the TSP. The following heuristic approach tries to overcome these shortcomings.

3.2 Heuristic approach

The main purpose of the heuristic is to modify the previous exact approach in order to overcome its drawbacks. The high memory required to represent the whole set of feasible patterns can be managed by reducing the number of patterns considered, at the expense of the optimality guarantee. The second critical point is the large time required to solve the TSP. A simple alternative is to solve the TSP heuristically, which leads to a sharp decrease in computational time required. In order to obtain a good quality TSP solution, we adopted an iterated version of dynasearch (Congram, 2000) followed by a 3-OPT (Lin, 1965) local search.

The pseudocode is presented in Algorithm 2. A solution of the 1D-CSP with SDCL is given

Algorithm 2 Heuristic approach

Require: I, c_{ij} L, **d**
  $U \leftarrow$ **GenerateUnitaryPatterns**(I,**d**,L)
  d' $\leftarrow$ **d** ▷ *Make a copy of the required demand*
  $k \leftarrow 1$
  while TotalAmount(**d'**) > 0 **do** ▷ *While demand not fully satisfied*
    $\hat{B}_{\mathbf{d'}} \leftarrow$ **GenerateLongPatterns**(I, c_{ij}, **d'**,U)
    $P_k \leftarrow U \cup \hat{B}_{\mathbf{d'}}$
    $S_k \leftarrow$ **SC**(I, **d'**, P_k)
    Update **d'** considering all patterns $p \in S_k$ fixed
    $k \leftarrow k + 1$
  end while
  Solution $\leftarrow$ **SC**(I, **d**, $\cup_k P_k$)
  Output: Solution

by the patterns and the corresponding number of occurrences required to completely satisfy the item demand. If a part of the solution is fixed by imposing some patterns, the remaining problem consists of finding other patterns to include in the solution in order to fulfil the demand.

Let U be the set of all the unitary patterns, formed by one item. If the set U is a proper subset of the pattern set considered as input of the SC step, it will always provide a feasible solution. In the worst case it will be given by adopting d_i times each unitary pattern $\{i\}$ $\forall i \in I$. Let $B_{\mathbf{d}}$ be the set of the longest feasible patterns considering item set I and demand vector **d**.

Each iteration k of the algorithm generates a solution to the problem by applying the SC model to a set of patterns P_k, with $U \subseteq P_k$ in order to guarantee feasibility. More precisely, P_k is determined as the union of U and a set $\hat{B}_{\mathbf{d'}} \subseteq B_{\mathbf{d'}}$, which includes the best patterns of $B_{\mathbf{d'}}$ according to the leftover criterion. Note that **d'** is the vector representing the current unsatisfied demand for each item. In the pseudocode, the set $\hat{B}_{\mathbf{d'}}$ is computed by the function **GenerateLongPatterns**.

GenerateLongPatterns starts from the unitary patterns U and iteratively generates good feasible patterns of increasing length. First, the unitary patterns are merged with themselves, generating all the possible patterns of length 2 (avoiding repetitions). Subsequently, the inner optimization algorithm is applied to the generated patterns and the best ones are kept according to the following criteria:

1. The ratio $\frac{LOAD}{WASTE}$, where $LOAD$ is the sum of the lengths of the items in the pattern and $WASTE$ is the sum of the cut losses. If it is high, it means that the stock material is filled properly.

2. The value of $WASTE$.

3. The level of item coverage, such that the final set contains patterns formed by a wide variety of items.

The best patterns are merged with unitary patterns again considering all the possible combinations, and the selection of the best patterns is performed. These steps are iterated as long as feasible patterns can be generated, i.e. while patterns do not violate the stock material length constraint. In general, the first criterion is more suitable for long patterns (for which it is more important to maximize raw material utilization), while the second works better for small or medium size patterns (for which large leftovers are acceptable, and the goal is to minimize cut losses).

When the function **GenerateLongPatterns** ends, the set $\hat{B}_{\mathbf{d'}}$ is defined and P_k can be computed as $U \cup \hat{B}_{\mathbf{d'}}$. At this point, the SC model is solved on P_k and a feasible solution of the problem is provided. This solution is formed by two types of patterns: patterns belonging to U and patterns belonging to $\hat{B}_{\mathbf{d'}}$. The latter are the best among the longest patterns that can be generated at iteration k. They are imposed as part of the final solution The remaining demand **d'** is updated considering this part of the solution as fixed.

These operations are iterated until the overall demand **d** is satisfied. Afterwards, the SC step is performed on all the generated patterns $\cup_k P_k$ in order to obtain a better solution.

The whole procedure can be seen as a column generation based primal heuristic, more precisely as a restricted master heuristic (see Joncour et al. (2010) for details). The restricted master problem is model (2) - (4), and the initial columns given by the unitary patterns ensure feasibility. Additional columns are generated heuristically by the **GenerateLongPatterns** procedure.

4 Computational study

The main research question in this work is to assess circumstances in which SDCL are relevant to consider, rather than disregarding them and applying a 1D-CSP or 1D-BPP approach. We therefore set up a computational study that compares the performance of the HSD heuristic and a 1D-BPP approach on a set of instances with varying characteristics. For the 1D-BPP approach, we opted for the MBS2+VNS heuristic from Fleszar and Hindi (2002). It performs well and considers the maximization of the sum of squared loads, thus aiming for very dense packings, and indirectly for reusable leftovers. This MBS2+VNS heuristic is applied to the 1D-BPP conversion of the problem described in Section 2.3.

The tests were executed on an Intel Core i5-3550 3.30 GHz and 4GB RAM, under Windows 7. The algorithms were implemented in C++ (except MBS2 + VNS which was coded in Java). CPLEX 12.4 was used as ILP solver. A time limit of 200 seconds was set for the SC steps of the HSD heuristic.

4.1 Experimental setup

A set of instances was randomly generated in such a way that it includes instances with varying item size and cut losses. Firstly, all the instances have been generated such that the stock size is $L = 1000$. Two datasets have been generated. The main dataset covers a wide range of parameter values. The second, smaller, dataset consists of instances generated to study a specific feature.

Instances of the first dataset are uniquely identified by four parameters, denoted IS, N, CLV and ID. The first parameter IS establishes the probabilistic distribution of the item sizes, computed according to a truncated Poisson distribution with minimum and maximum values of respectively 50 and 200, and mean $\lambda = IS$. The second parameter $N = \sum_{i \in I} d_i$ denotes the total number of items to be cut. The demands d_i are generated according to a uniform distribution between

5 and 10, forcing that N is fixed. N can take values 25, 50, 100, 200 and 300. It defines a first approximation of the size of the instance. The third parameter CLV defines the variability of the cut losses. Both a high and low variability of the cut losses are studied. The cut losses are generated according to a uniform distribution; a setting of CLV = High denotes cut losses between 15 and 45, and CLV = Low denotes cut losses between 5 and 15. These distributions lead to an average global cut losses variability corresponding respectively to 3% and 1% of the whole stock size.

Finally, five instances (distinguished by ID) with the same settings of the first three parameters have been generated. In what follows, we present the averaged results over these five instances, per parameter setting. The overall number of instances in this dataset is equal to 150.

The main dataset contains no instances with a very low cut loss variability. Consequently, the second set of instances was constructed with very small cut losses in order to compare the performances of HSD and 1D-CSP approaches under these circumstances. N is fixed and equal to 100 and item sizes have been generated according to a uniform distribution between 100 and 150. Five instances are generated for five different values of absolute cut loss variability ($CL =$ 0,1,2,3,4). The cut losses vary uniformly between $[1, 1 + CL]$.

4.2 Experimental results and discussion

The averaged results of the tests performed on the main dataset are reported in Tables 1, 2 and 3. HSD indicates the heuristic based on sequence dependencies introduced in Section 3.2, while MBS2+VNS refers to the heuristic by Fleszar and Hindi (2002). The tables report the following results for both HSD and MBS2+VNS: the averaged values of the first objective ($O1$ = number of stock material used), the second objective ($O2$ = sum of squared leftovers) and the execution time of the algorithm (T, in seconds) per parameter setting, as well as the averaged lower bound (LB) for those instances. This lower bound has been determined with the column generation procedure of Vancroonenburg et al. (2014) that solves the linear relaxation of 1D-CSP with SDCL, considering only $O1$. The lower bound LB_{BP} is the lower bound for the bin packing conversion of the problem. LB_{BP} enables assessing the quality of the heuristic results (with respect to $O1$) for the bin packing version. For both HSD and MBS2+VNS, the average computation time required for finding these bounds is reported, as well as the number of optimal solutions found ($\#OPT$, maximum = 5). The time limit is set to 12 hours.

The computational results reveal that HSD is the best performing heuristic in terms of number of cutting patterns required. Considering SDCL is therefore relevant to find cutting patterns that make better use of raw material. Even a cut losses variability as small as 1% of the stock size and an item size as large as 17% of the stock size, should not be overlooked when considering SDCL. However, the consideration of SDCL comes at a computational cost, as the HSD heuristic requires significantly more computational time. It is possible to conclude that the computational effort is higher when item sizes are smaller (as in the case of $IS = 80$) and cut losses are smaller (as in the case of $CLV = Low$). In these cases, the inner optimization is more computationally intensive since patterns are formed by a larger number of items. Smaller cut losses make HSD less effective and increase its execution time.

Table 4 summarizes the results on the second set of instances that have very small cut losses. As expected, MBS2+VNS performs better than HSD if the cut losses are very small. HSD can only outperform MBS2+VNS if the absolute cut losses are larger and vary more (i.e. when cut losses are between $[1, 3]$ and larger). In the cases where the results on $O1$ are identical, HSD performs better than MBS2+VNS.

Main dataset		HSD						MBS2+VNS					
N	CLV	LB	$T_{LB}(s)$	$O1$	$O2$	$T(s)$	# OPT.	LB_{BP}	$T_{LB}(s)$	$O1$	$O2$	$T(s)$	# OPT.
25	Low	3.0	1.2	**3.0**	**526620.0**	1.6	5	3.0	0.1	**3.0**	417392.6	0.0	5
25	High	3.0	0.3	**3.0**	124866.4	1.0	5	4.0	0.1	4.0	454319.4	0.0	5
50	Low	5.0	13.3	**5.0**	**449547.2**	14.5	5	5.0	0.1	**5.0**	111841.2	0.2	5
50	High	5.8	5.2	**5.8**	352242.6	7.8	5	7.2	0.1	7.2	320836.6	0.2	5
100	Low	9.4	104.1	**9.6**	557879.0	63.4	4	10.2	0.2	10.2	374432.0	0.9	5
100	High	10.2	65.8	**10.4**	218870.4	33.6	4	13.2	0.2	13.2	134662.4	0.1	5
200	Low	17.8	831.6	**17.8**	433030.2	996.3	5	19.6	0.3	19.6	488221.4	2.1	5
200	High	20.2	3151.3	**20.6**	255675.6	149.8	3	26.8	0.6	26.8	122297.6	0.2	5
300	Low	26.4	2327.7	**26.6**	147403.6	13710.4	4	29.4	0.8	29.4	213160.4	0.3	5
300	High	29.8	4965.2	**30.6**	533039.4	416.2	1	39.6	0.9	40.0	300060.4	0.4	3

Table 1: Results on instances with $IS = 80$. Missing values for the lower bounds denote that the column generation model did not produce a bound with a time limit of 12 hours for at least one instance. Bold indicates best results.

Main dataset		HSD						MBS2+VNS					
N	CLV	LB	$T_{LB}(s)$	$O1$	$O2$	$T(s)$	# OPT.	LB_{BP}	$T_{LB}(s)$	$O1$	$O2$	$T(s)$	# OPT.
25	Low	4.0	1.1	**4.0**	**398896.6**	0.4	5	4.0	0.1	**4.0**	248966.6	0.0	5
25	High	4.8	0.3	**4.8**	579865.4	0.2	5	5.0	< 0.1	5.0	136972.0	0.0	5
50	Low	7.0	48.9	**7.0**	116578.0	1.4	5	7.6	0.1	7.6	461525.6	0.1	5
50	High	8.0	45.7	**8.2**	311025.8	1.4	4	9.8	< 0.1	9.8	263596.0	0.1	5
100	Low	13.8	2164.4	**14.0**	213577.4	7.0	4	14.8	0.2	14.8	388055.6	0.3	5
100	High	15.0	1090.2	**15.4**	379002.4	4.7	3	18.6	0.2	18.6	266930.8	0.1	5
200	Low	-	31528.9	**27.2**	471917.8	76.7	-	28.8	0.5	28.8	478105.2	0.2	5
200	High	29.2	5925.7	**30.0**	281166.0	24.0	1	36.8	0.3	37.6	506611.2	0.3	1
300	Low	-	40269.6	**40.8**	310140.0	121.4	-	43.0	1.1	43.4	432623.6	0.3	3
300	High	-	21508.1	**44.8**	305606.8	62.0	-	55.8	0.7	56.4	495685.2	0.6	2

Table 2: Results on instances with $IS = 125$. Missing values for the lower bounds denote that the column generation model did not produce a bound with a time limit of 12 hours for at least one instance. Bold indicates best results.

Main dataset		HSD						MBS2+VNS					
N	CLV	LB	$T_{LB}(s)$	$O1$	$O2$	$T(s)$	# OPT.	LB_{BP}	$T_{LB}(s)$	$O1$	$O2$	$T(s)$	# OPT.
25	Low	5.0	0.1	**5.0**	**204486.8**	0.1	5	5.0	0.1	**5.0**	70145.6	0.0	5
25	High	5.4	0.1	**5.6**	277806.2	0.1	4	7.0	< 0.1	7.0	637881.2	0.0	5
50	Low	9.6	11.6	**9.8**	183052.4	0.3	4	10.0	< 0.1	10.0	190604.4	0.1	5
50	High	10.2	0.9	**10.4**	222870.8	0.3	4	12.6	< 0.1	12.6	300492.6	0.2	5
100	Low	19.2	671.4	**19.6**	257374.6	1.5	3	19.8	0.1	19.8	117312.6	0.2	5
100	High	20.0	510.2	**20.2**	144705.6	1.0	4	24.6	0.1	24.6	467421.2	0.4	5
200	Low	-	36825.5	**37.6**	439263.2	9.6	-	38.8	0.3	39.2	474066.6	0.4	3
200	High	-	23508.4	**40.0**	135964.6	5.6	-	49.2	0.1	49.2	624402.4	2.8	5
300	Low	-	22080.9	**57.0**	716090.4	35.4	-	58.6	0.6	59.0	522072.4	0.7	3
300	High	-	42738.6	**60.0**	214577.6	17.4	-	74.6	0.2	74.6	1025290.8	4.1	5

Table 3: Results on instances with $IS = 170$. Missing values for the lower bounds denote that the column generation model did not produce a bound with a time limit of 12 hours for at least one instance. Bold indicates best results.

	HSD						MBS2+VNS					
CL	LB	$T_{LB}(s)$	$O1$	$O2$	$T(s)$	# OPT.	LB_{BP}	$T_{LB}(s)$	$O1$	$O2$	$T(s)$	# OPT.
0	13.4	0.4	13.6	526327.8	19.1	4	13.4	0.3	**13.4**	464630.0	1.3	5
1	13.4	1224.4	13.6	422938.8	16.4	4	13.4	0.3	**13.4**	311685.0	1.0	5
2	13.4	1297.3	**13.6**	**491371.8**	17.2	4	13.6	0.3	**13.6**	370179.0	0.7	5
3	13.4	766.7	**13.4**	290571.6	15.9	5	13.6	0.3	13.6	272759.4	1.4	5
4	13.4	1145.9	**13.6**	**386685.6**	14.1	4	13.6	0.2	**13.6**	177527.4	0.8	5

Table 4: Summary of results on the second set of generated instances, with low cut loss variability. Cut losses are drawn uniformly from $[1, 1 + CL]$

5 Two case studies

Having shown experimentally when the specific nature of SDCL becomes interesting, we conclude this paper with two industrial cases. The first case study considers a cutting stock problem at joineries, where SDCL are rather small. The second case study investigates the truss cutting problem (Lewis et al., 2011) while omitting item orientations. In this problem, SDCL exist due to the conversion of a 2D problem to a 1D problem. Because of this conversion, the SDCL are rather large. Following from our computational study in Section 5.2, we can expect that the consideration of SDCL will be most significant for the TCP.

5.1 Case study 1: joinery stock cutting with angle specific cutting losses

Cutting stock is an important daily activity in joineries: wooden, aluminium and plastic profiles of standard length are cut to specific order lengths for assembling door and window frames. These frame assemblies require the profiles to be cut under specific angles, either 45 (oblique) or 90 (right) degrees. Cutting profiles always destroys a small part of the material, proportional to the width of the saw blade. This loss depends on the angle under which the material is cut: cutting a plastic profile under a right angle may cause a material loss of 5 mm, whereas cutting the same profile under an oblique angle may cause an 8 mm loss.

To illustrate how this can make a difference, consider the example in Fig. 3. It shows three profiles A, B and C that need to be cut from a standard length. Fig. 4 shows two possible ways. It is clear that solution (a) requires 4 oblique cuts and 2 orthogonal cuts, while solution (b) requires 4 oblique cuts and only 1 orthogonal cut. This results from the fact that profile C is cut after profile A in solution (b). This allows the orthogonal cut at the end of profile C, and the orthogonal cut at the start of profile A, to be cut in one saw blade motion, which reduces cut loss. Although this loss might seem negligible, it does not hold when the cut angle diversity is high and when the stock length to average profile length is high. Inefficient cutting patterns may lead to unnecessary cut losses and more material waste.

Real life data was obtained from a company developing joinery stock cutting software. Converting the data instances to 1D-CSP with SDCL instances can be done as follows:

- Each of the trapezoidal assembly parts are modelled by an item. The length of the item l_i is set to the longest base of the trapezoid.

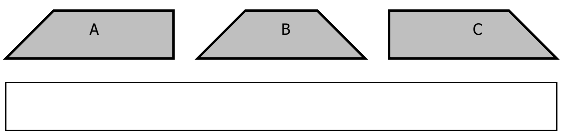
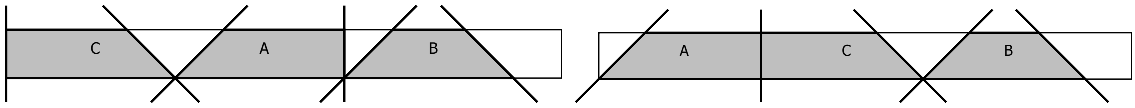


Figure 3: Simple joinery stock cutting example showing three profiles A,B and C that need to be cut from a standard length profile.



(a) (b)

Figure 4: Two possible solutions of the simple problem in Fig. 3.

Instance		I		c_{ij}			
Profile	L (mm)	μ_{l_i} (mm)	std (mm)	$\mu_{c_{ij}}$ (mm)	std (mm)	$\frac{\mu_{l_i}}{L}$	$\frac{\mu_{c_{ij}}}{\mu_{l_i}}$
3001	5400	1156.05	442.18	15.41	1.91	0.214	0.013
3001	6000	1156.05	442.18	15.41	1.91	0.193	0.013
3001	6500	1156.05	442.18	15.41	1.91	0.178	0.013
3001	7000	1156.05	442.18	15.41	1.91	0.165	0.013
3002	5400	1690.08	514.76	13.15	3.54	0.313	0.008
3002	6000	1690.08	514.76	13.15	3.54	0.282	0.008
3002	6500	1690.08	514.76	13.15	3.54	0.260	0.008
3002	7000	1690.08	514.76	13.15	3.54	0.241	0.008
3082	5400	607.48	136.89	10.88	4.26	0.112	0.018
3082	6000	607.48	136.89	10.88	4.26	0.101	0.018
3082	6500	607.48	136.89	10.88	4.26	0.093	0.018
3082	7000	607.48	136.89	10.88	4.26	0.087	0.018
5182	5400	557.56	224.17	12.08	3.98	0.103	0.022
5182	6000	557.56	224.17	12.08	3.98	0.093	0.022
5182	6500	557.56	224.17	12.08	3.98	0.086	0.022
5182	7000	557.56	224.17	12.08	3.98	0.080	0.022

Table 5: Instance characteristics of joinery stock cutting data. Four different instances were created with a stock length L of respectively 5400 mm, 6000 mm, 6500 mm and 7000 mm, for each profile type.

- The sequence dependent cut losses c_{ij} are set as follows:

$$c_{ij} = \begin{cases} loss_{90} & \text{if endcut}(i) = \text{startcut}(j) = 90°, \\ loss_{90} + loss_{45} & \text{if endcut}(i) = 45° \text{ \& startcut}(j) = 90° \text{ or vice-versa}, \\ 2 \times loss_{45} & \text{if endcut}(i) = \text{startcut}(j) = 45°. \end{cases} \tag{5}$$

Start and end losses c_{0i} and c_{i0} are set as follows:

$$c_{0i} = \begin{cases} loss_{90} & \text{if startcut}(i) = 90°, \\ loss_{45} & \text{otherwise.} \end{cases} \tag{6}$$

$$c_{i0} = \begin{cases} loss_{90} & \text{if endcut}(i) = 90°, \\ loss_{45} & \text{otherwise.} \end{cases} \tag{7}$$

Four instances contained SDCL and were made available. The instances were tested with four different values of L: 5400 mm, 6000 mm, 6500 mm, and 7000 mm, which are the most common, commercially available, stock lengths for this case study. Table 5 reports the following characteristics of the instances: the mean item length (μ_{l_i}) and the item length standard deviation, the mean cut loss ($\mu_{c_{ij}}$) and the cut loss standard deviation, and the mean item length to stock size ratio and the mean cut loss to average item length ratio.

Table 6 presents the performance of both HSD and MBS2+VNS on the real life data set. It shows that for instances with profile 5182, savings can be made on $O1$ when using HSD instead of MBS2+VNS. When considering the instance characteristics (Table 5), it becomes clear that these particular instances have a small average item length (denoted μ_{l_i}) to large object size L ratio,

		HSD				MBS2+VNS			
Profile	L (mm)	LB	$T_{LB}(s)$	$O1$	$T(s)$	LB_{BP}	$T_{LB}(s)$	$O1$	$T(s)$
3001	5400	-	-	49	3.2	49	3.4	49	8.4
3001	6000	-	-	44	7.1	44	4.6	44	4.6
3001	6500	41	28869.6	41	7.4	41	5.2	41	8.9
3001	7000	38	19780.9	38	13.1	38	5.2	38	2.8
3002	5400	8	0.1	8	0.2	8	0.1	8	0.1
3002	6000	8	45.9	8	0.2	8	0.1	8	< 0.1
3002	6500	7	0.4	7	0.3	7	0.1	7	0.1
3002	7000	7	3115.8	7	0.3	7	0.2	7	< 0.1
3082	5400	6	1779.4	6	2.3	6	0.1	6	0.3
3082	6000	5	10449.8	6	3.9	6	0.3	6	0.3
3082	6500	-	-	5	4.8	5	0.3	5	0.1
3082	7000	5	2033.6	5	7.3	5	0.3	5	0.2
5182	5400	-	-	**188**	264.2	189	1296.7	190	148.8
5182	6000	-	-	**170**	324.1	171	6757.9	171	107.8
5182	6500	-	-	**157**	581.4	157	37649.2	158	200.3
5182	7000	-	-	**145**	776.3	-	-	146	129.6

Table 6: Summary of results on 12 real life instances of the joinery stock cutting problem. Note that missing lowerbounds indicate that the lowerbound could not be calculated withing 12 hours of computation time. Bold indicates best results.

and a large average cut loss (denoted $\mu_{c_{ij}}$) to average item length ratio. Consequently, many items fit in a single stock length L, and thus incur more cut losses. In this case, the total cut losses are larger than average. Furthermore, the standard deviation (denoted std.) of the cut losses is higher, resulting in a higher maximum c_{ij}. The bin packing approach MSB2+VNS is therefore less suited.

5.2 Case study 2: the truss cutting problem

The truss cutting problem (TCP) is a problem originating from the roofing industry (Lewis et al., 2011). Profiles (having trapezoidal shapes) of equal width have to be cut from wooden boards with the aim of minimizing area waste. Lewis et al. show that the TCP is a special case of 2D-CSP and can be solved with a 1D packing approach. Sequence dependent cut losses (inter-item wastage) are due to the shape of the items that may fit better together in certain arrangements.

The TCP can be converted to the 1D-CSP with SDCL as follows. Each trapezoidal piece is modelled as an item with length l_i equal to the central width of the parallel part, while the left and right sloped edges are projected vertically (denoted p_i and q_i) and contribute as losses. The sequence dependent losses c_{ij} between items i, j are then set as follows:

$$c_{ij} = \max(q_i, p_j) \tag{8}$$

As shown by Lewis et al. , adjacent trapezoids can always be flipped along their horizontal axis, such that their adjacent sloped parts overlap. Therefore, Equation 8 only considers the maximum projection between the central parallel parts of the two adjacent items. Finally, Lewis et al. consider a bin packing formulation. However, the TCP in a practical setting actually considers many identical trapezoids to be cut. In our cutting stock setting, identical trapezoidal pieces can easily be modelled as a single item with demand > 1.

Figure 5 shows an example of how this conversion works. Notice that the loss between any two adjacent items, e.g. (i, j) is equal to the maximum projection between i, j. In this case, $c_{ij} = \max(q_i, p_j) = p_j$. Also notice that item k has been flipped along its horizontal axis.

Table 7 summarizes the results of both HSD and MBS2+VNS on the 'r-dataset' from (Lewis et al., 2011). As opposed to the 'a-dataset', which features instances with all items different

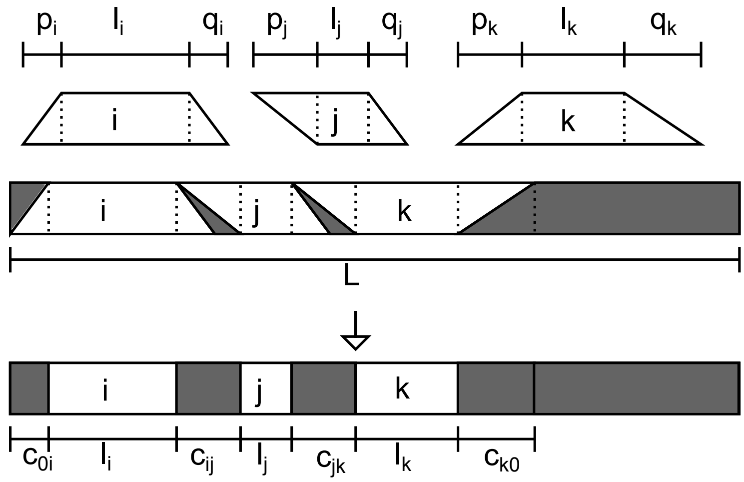


Figure 5: An example of converting the geometric sequence dependencies in the TCP to the more general 1D-CSP SDCL. (top) Three trapezoidal items, annotated with their central width and left and right projections, need to be cut from stock material of length L. (middle) An example arrangement of the three trapezoidal figures. Notice that trapzoid k is flipped for minimal waste. (bottom) Conversion of the three trapezoids to abstract items with SDCL, and a feasible arrangement.

($d_i = 1 \forall i \in I$), the r-dataset is more realistic since it features instances with fewer items having a demand higher than one. Given that our approach is focussed on cutting stock problems where demand is an important factor, we only consider the r-dataset. Table 7 also shows results produced by the Algorithm A heuristic from Lewis et al. (2011). Lewis et al. kindly provided their code, which we modified to not consider orientations[1]. As their algorithmic framework is local search based with a time-limit as stopping criterion, the executable was configured with a time-limit of 60 s. This was deemed sufficient by the authors (confirmed by Lewis et al. after personal correspondence), and experimentation does indicate that the algorithm converges much sooner. We report both the initial solution found by this code, the time required to find it, as well as the final solution found. These results can serve as a reference point for determining the quality of HSD when SDCL are really important. Finally, the code also generates a lower bound for the instances, which is also reported as LB_{TR}.

The r-dataset contained 1200 instances, with a fixed stock size of $L = 4200$ (mm) and item sizes ranging from small to large with respect to the stock size. The instances can be grouped according to the number of items N that need to be cut ($N = \sum_{i \in I} d_i$), ranging from 100 to 500, with each group containing 240 instances.

When considering the performance of HSD and MBS2+VNS, it is clear that HSD outperforms MBS2+VNS for $O1$. This was expected, as the SDCL contribute a lot in these instances. Another interesting observation is that more than 87% of the instances were solved to optimality by the HSD heuristic, as determined by the column generation lower bound.

When comparing HSD with Algorithm A, HSD overall performs slightly better. HSD finds more optimal results and thus has an overall better average $O1$. However, it must be noted that Algorithm B described in Lewis et al. (2011) has slightly better performance (with respect to Algorithm A), at additional computational cost. Algorithm B uses an exact approach for solving the inner optimization problem[2], enabling it to find better packings. However, we did not test Algorithm B in this work.

[1]Lewis et al. also allow items to be flipped along their vertical axis, thus allowing two orientations: left-to-right (not flipped), and right-to-left (flipped).

[2]Referred to as the TCP sub problem by the Lewis et al.

			Lewis et al. (2011) Alg. A init.*			Lewis et al. (2011) Alg. A final*		
N	# instances	LB_{Tr}	$O1$	$T(s)$	# OPT.	$O1$	$T(s)$	# OPT.
100	239[b]	38.6	43.3	< 0.1	125	42.9	60	201
200	240	75.5	84.4	< 0.1	66	83.5	60	152
300	239[b]	115.0	128.9	< 0.1	75	127.6	60	139
400	238[b]	154.5	173.2	< 0.1	67	171.3	60	118
500	239[ab]	193.0	217.4	0.3	73	215.1	60	115

			HSD[a]					MBS2+VNS[b]				
N	# instances	LB_{Tr}	LB	$T_{LB}(s)$	$O1$	$T(s)$	# OPT.	LB_{BP}	$T_{LB}(s)$	$O1$	$T(s)$	# OPT.
100	239[b]	38.6	42.7	45.7	**42.8**	1.9	220	47.2	0.2	47.3	0.8	228
200	240	75.5	83.0	70.8	**83.0**	2.0	223	92.7	0.3	92.9	1.3	205
300	239[b]	115.0	126.8	71.1	**127.0**	774.9	217	140.7	0.2	141.1	2.1	180
400	238[b]	154.5	170.2	76.7	**170.4**	2.0	194	188.1	0.3	188.8	2.6	154
500	239[ab]	193.0	213.9	93.8	**214.1**	2.0	198	237.0	0.2	237.8	3.8	149

Table 7: Summary of the results on the TCP instances. Bold indicates best results.
*Code from Lewis et al. (2011), adapted to ignore orientation changes.
[a]Note that one instance (with $N = 500$) could not be solved by the HSD heuristic, due to an out-of-memory error in the SC step. This instance was excluded for all algorithms, and is not considered for the averages.
[b]Note that 5 instances were infeasible for the bin packing approximation due to the maximum item length + maximum cut loss exceeding L. These instances were excluded for all algorithms, and are not considered for the averages.

6 Conclusion

The present paper introduced a one dimensional cutting stock problem with sequence dependent cut losses (1D-CSP with SDCL) that considers minimization of the number of raw materials required for cutting a set of items. As a secondary objective, the reusability of leftover material is considered.

It was shown that the problem can be approximately solved using a standard one dimensional cutting stock problem or one dimensional bin packing approach. Therefore, the main research question was to identify beneficial conditions to specifically consider SDCL. To this end, a heuristic pattern based approach specifically taking the SDCL into account was developed.

A computational study on a set of generated instances with varying characteristics showed that it is clearly beneficial to consider SDCL, whenever the item size is small, i.e. up to 17% of the stock size, and when cut losses are not too small and have some variability, i.e. larger than 0.2% of the stock size).

Finally, two practical applications of the 1D-CSP with SDCL model show the relevance of considering SDCL. In a joinery stock cutting case, it is shown that raw material waste can be avoided by considering SDCL, even though the SDCL are very small (< 0.3% of the stock size). In a roofing industry case, it is shown that the 1D-CSP with SDCL model is flexible enough to tackle a 2D cutting problem. In this setting, the consideration of SDCL is indispensable since the SDCL are rather large.

Acknowledgements: We would like to thank TECHWIN Software bvba for introduction of the joinery stock cutting problem and for providing data. Furthermore, we are very thankful to Lewis et al. (2011) for providing their code and instances to setup a comparison, and for the fruitful discussions that followed from that. This research has been conducted in the context of project IWT 090294.

This work is supported by the Belgian Science Policy Office (BELSPO) in the Interuniversity Attraction Pole COMEX (http://comex.ulb.ac.be) and is partially funded by a Ph.D. grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders. (IWT-Vlaanderen)

References

M. Armbruster. A solution procedure for a pattern sequencing problem as part of a one-dimensional cutting stock problem in the steel industry. *European Journal of Operational Research*, 141(2): 328–340, Sept. 2002. ISSN 03772217.

T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega: The International Journal of Management Science*, 34(3):209–219, 2006.

R. K. Congram. *Polynomially searchable exponential neighbourhoods for sequencing problems in combinatorial optimisation*. PhD thesis, University of Southampton, 2000.

Y. Cui and Y. Yang. A heuristic for the one-dimensional cutting stock problem with usable leftover. *European Journal of Operational Research*, 204(2):245–250, July 2010. ISSN 03772217.

H. Dyckhoff. A typology of cutting and packing problems. *European Journal of Operational Research*, 44:145–159, 1990.

K. Fleszar and K. S. Hindi. New heuristics for one-dimensional bin-packing. *Computers & Operations Research*, 29:821–839, 2002.

P. C. Gilmore and R. E. Gomory. A linear programming approach to the cutting-stock problem-part ii. *Operations Research*, 11:864–888, 1963.

O. Holthaus. Decomposition approaches for solving the integer one-dimensional cutting stock problem with different types of standard lengths. *European Journal of Operational Research*, 141(2):295–312, Sept. 2002. ISSN 03772217.

C. Joncour, S. Michel, R. Sadykov, D. Sverdlov, and F. Vanderbeck. Column generation based primal heuristics. *Electronic Notes in Discrete Mathematics*, 36(0):695–702, 2010. ISSN 1571-0653.

G. Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59:345–358, 1992.

R. Lewis, X. Song, K. Dowsland, and J. Thompson. An investigation into two bin packing problems with ordering and orientation implications. *European Journal of Operational Research*, 213(1): 52 – 65, 2011. ISSN 0377-2217.

S. Lin. Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, 44: 2245–2269, 1965.

M. J. P. Lopes and J. V. de Carvalho. A branch-and-price algorithm for scheduling parallel machines with sequence dependent setup times. *European Journal of Operational Research*, 176 (3):1508–1527, 2007. ISSN 0377-2217.

A. Mobasher and A. Ekici. Solution approaches for the cutting stock problem with setup cost. *Computers & Operations Research*, 40(1):225–235, Jan. 2013. ISSN 03050548.

P. Trkman and M. Gradisar. One-dimensional cutting stock optimization in consecutive time periods. *European Journal of Operational Research*, 179(2):291–301, June 2007. ISSN 03772217.

W. Vancroonenburg, P. De Causmaecker, F. Spieksma, and G. Vanden Berghe. Column generation for the 1D-CSP with SDCL, and the orienteering problem with multiplicity. Technical report, KU Leuven, Department of Computer Science, CODeS & iMinds-ITEC, 2014. URL `http://allserv.kahosl.be/~wimvc/cg-1D-csp-with-sdcl.pdf`.

G. Wäscher, H. Hauner, and H. Schumann. An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183:1109–1130, 2007.

H. H. Yanasse and M. S. Limeira. A hybrid heuristic to reduce the number of different patterns in cutting stock problems. *Computers & Operations Research*, 33(9):2744–2756, Sept. 2006. ISSN 03050548.