New Stabilization Procedures for the Cutting Stock Problem

François Clautiaux
Université des Sciences et Technologies de Lille, LIFL UMR CNRS 8022, INRIA, Bâtiment INRIA, Parc de la Haute Borne, 59655 Villeneuve d'Ascq, France, francois.clautiaux@univ-lille1.fr
Cláudio Alves, José Valério de Carvalho, and Jürgen Rietz
Departamento de Produção e Sistemas, Escola de Engenharia, Universidade do Minho, 4710-057 Braga, Portugal, {claudio,vc}@dps.uminho.pt, Juergen_Rietz@gmx.de

In this paper, we deal with a column generation based algorithm for the classical cutting stock problem. This algorithm is known to have convergence issues, which are addressed in this paper. Our methods are based on the fact that there are interesting characterizations of the structure of the dual problem, and that a large number of dual solutions are known. First we describe methods based on the concept of *dual cuts*, proposed by Valério de Carvalho (2005). We introduce a general framework for deriving cuts, and we describe a new type of dual cuts, which exclude solutions that are linear combinations of some other known solutions. We also explore new lower and upper bounds for the dual variables. Then we show how the prior knowledge of a good dual solution helps improving the results. It tightens the bounds around the dual values, and makes the search converge faster if a solution is sought in its neighborhood first. A set of computational experiments on very hard instances are reported at the end of the paper. They confirm the effectiveness of the methods proposed.

Key words: combinatorial optimization; integer programming; cutting stock

---

1. Introduction

The cutting stock problem (CSP) consists in finding the minimum number of identical stock rolls needed to cut a set of items of different sizes. It belongs to the class of *Cutting and Packing* problems as a *single stock-size cutting stock problem* (Wäscher et al., 2007). Each item has a given demand, which corresponds to the number of times it has to be cut from the stock rolls.

Many researchers have used Dantzig-Wolfe decomposition (Dantzig and Wolfe, 1960) and *column generation methods* for solving the CSP (see Gilmore and Gomory (1961, 1963) for the first attempt). The integer program is decomposed into a restricted master problem

initialized with a set of columns, and optimized to determine the value of the dual variables. The dual information is passed to a subproblem that evaluates if there is still any column that can be added to the master problem and improve the current solution. If there is such a column, the master problem is reoptimized, otherwise the process stops.

Column generation processes are known to have convergence issues. The dual variables may oscillate from one iteration to the next (Du Merle et al., 1999), and primal degeneracy may also arise (Luebbecke and Desrosiers, 2005). Convergence of column generation has been the topic of many contributions. One of the first approaches was the *Boxstep* method of Marsten et al. (1975). Their method tries to stabilize and accelerate the column generation process by guiding the dual variables, imposing box constraints on the solutions of the restricted master. Kallehauge et al. (2006) used a similar scheme to solve the vehicle routing problem with time windows, but instead of considering fixed-size boxes as in Marsten et al. (1975), they allowed the boxes to be dynamically updated. In both the papers of Marsten et al. (1975) and Kallehauge et al. (2006), the dual variables cannot lie outside the boundaries defined by the box constraints. Du Merle et al. (1999) relaxed this constraint. The dual variables may take values outside the boxes, but there is a proportional penalty if that situation occurs. In the primal, this approach consists in adding weighted and bounded slack and surplus variables to the original constraints of the problem. The authors report good results for transportation and location problems.

Other approaches that keep the solution process within the framework of linear programming have been described in the literature. Kim et al. (1995), for example, considered a linear penalty method that can be extended to column generation (Neame, 1999). As an alternative, many nonlinear methods can be used to stabilize column generation such as the bundle methods (Hiriart-Urruty and Lemaréchal, 1993) and the analytic centers cutting plane methods (Goffin and Vial, 1989), for example.

Degraeve and Peeters (2003) solved the CSP with column generation using an hybrid procedure that relies on a subgradient method. Initially, the dual values of the restricted master are updated using subgradient optimization, and columns are generated based on these values. Simplex is used at a second stage to optimize the master problem with the new columns. The subgradient method allows for a fast update of the dual values, and in practice, their approach accelerates the column generation process.

One of the most promising approaches to stabilized column generation has been proposed by Carvalho (2005). The method consists in adding a polynomial number of *dual cuts* (primal

columns) to the restricted master problem. Carvalho (2005) proposed a set of *weak* dual-optimal inequalities for the CSP, which do not cut any optimal solution of the dual problem (Ben Amor et al., 2006a). Since then, the method has been used successfully to accelerate other cutting problems (Alves and Carvalho, 2007; Puchinger and G., 2005). Alves and Carvalho (2008) showed how to solve the multiple length CSP with branch-and-price using these dual cuts in all the nodes of the branch-and-bound tree. In this paper, we explore the dual cuts of Valério de Carvalho from a different point of view. We propose new valid dual cuts for the CSP based on new theoretical results, and we devise new procedures to stabilize column generation processes.

The cuts of Carvalho (2005) can be seen as a restriction of the dual space to a subset of non-dominated solutions sharing a special structure. The method is based on the fact that there is always an optimal dual solution whose coefficients can be obtained by applying a non-decreasing and *superadditive* function to the sizes of the items. In this paper, we introduce new dual cuts, which take into account the fact that a maximal solution has to be *symmetric* in addition to increasing and superadditive.

As they are defined by Carvalho (2005), the dual cuts only exclude solutions that are dominated by another single solution. Another type of cuts may be applied. It is a classical result of linear programming that any solution that is the linear combination of two other solutions cannot lead to improved results, even if it is not dominated by any of these two solutions separately. Consequently, finding properties of non-dominated solutions may lead to new effective dual cuts, if they can be expressed by the mean of a small set of linear constraints. Our approach is the following. First we show general results, which allow one to derive cuts from a valid dual solution. Then we exhibit two such solutions π^0 and π^1, and we propose hand-tailored cuts to avoid exploring some solutions that can be expressed as the linear combination of π^0 or π^1 and another solution.

Another contribution of this paper is a method that aims at finding lower and upper bounds for the dual values. The additional piece of information needed by this method is the value of a lower bound, which is obtained using a set of precomputed dual solutions. This value permits one to update the lower bounds for several dual values. Then the method exploits the characterization of the maximal solutions to propagate these lower bounds to all dual variables. We also generalize a result that allows us to remove the small item sizes if they cannot change the value of an optimal solution.

Another way of taking into account an interesting (hopefully optimal) known dual solu-

tion is to restrict the search space to a small box around this solution, similarly with Ben Amor et al. (2006b). To compute these boxes, we use the values given by the known dual solution that yields the best lower bound among a subset of functions. Such a dual solution can be obtained by applying a so-called *dual-feasible function* to the item sizes. The boxes are increased if they are too small, and at the end they are removed and the search is resumed. The idea is to make the method converge faster to an *interesting* area by avoiding large and useless fluctuations of the dual variables.

All the methods we propose lead to dual constraints (additional columns) that are introduced in a column generation based method for the CSP. Computational experiments on a set of hard instances from the literature and randomly generated ones confirm that each method reduces the number of column generation iterations and the total computing time.

Section 2 reviews several results of the literature concerning dual solutions and dual cuts. In Section 3 we propose new dual cuts. Section 4 is about bounds for the dual values. Section 5 contains numerical experiments.

2. Definitions and previous results

In this section we briefly review several concepts used in this paper. First we describe the model of Gilmore and Gomory (1961, 1963) for the CSP. Then we deal with properties of *maximal* dual solutions of the CSP. Finally we discuss on how *dual cuts* can be applied to speed up column generation processes.

Throughout the remainder, we will use the following notation for the CSP. An instance $D = (C, I, b)$ of CSP is composed of a roll size $C \in \mathbb{N}^*$, a set $I \subseteq \{1, \ldots, C\}$ of item sizes, and a demand vector $b \in \mathbb{N}^C$, which associates with each $i \in I$ a demand $b_i \in \mathbb{N}^*$. For the sake of simplicity, we consider in the remainder that $I = \{1, \ldots, C\}$. Any item size is present, but does not contribute to the cost function (we set its demand b_i to 0).

2.1. Column generation based model of Gilmore and Gomory (1961, 1963)

A combination of items of I in a roll is called a pattern. Each possible cutting pattern is described by a column $p = (a_{1p}, \ldots, a_{ip}, \ldots, a_{|I|p})^T$, where a_{ip} is the number of items of width i in the pattern p. The model of Gilmore and Gomory (1961, 1963) is expressed as follows:

$$min \sum_{p \in P} x_p \tag{1}$$

$$\text{subj. to} \quad \sum_{p \in P} a_{ip} x_p \geq b_i, \quad i \in I \tag{2}$$

$$x_p \geq 0, \qquad \forall p \in P \tag{3}$$

$$x_p \text{ integer }, \qquad \forall p \in P \tag{4}$$

where P is the set of all valid patterns. A valid cutting pattern is such that

$$\sum_{i \in I} a_{ip} i \leq C, \qquad \forall p \in P$$
$$a_{ip} \geq 0 \text{ and integer}, \quad \forall p \in P, i \in I$$

As the number of possible cutting patterns may be large, the master program is initialized with only a subset of cutting patterns (columns). It finds the best solution that only uses the patterns available. Then an optimization algorithm is executed to find a pattern that could improve the quality of the current solution. Degeneration occurs when the new pattern brings no improvement. The optimization problem to solve is equivalent to a knapsack problem, which can be solved using dynamic programming or an enumerative algorithm. The dual of the linear relaxation (1)–(3) of the CSP above reads:

$$max \qquad \sum_{i \in I} b_i \pi_i \tag{5}$$

$$\text{subj. to} \quad \sum_{i \in I} a_{ip} \pi_i \leq 1, \forall p \in P \tag{6}$$

$$\pi_i \geq 0 \tag{7}$$

For all $i \in I$ the pattern consisting of $\lfloor C/i \rfloor$ times the piece of length i (and no other item) belongs to P. Therefore $\pi_i \leq 1/\lfloor C/i \rfloor \leq 1$ for all $i \in I$.

A dual solution π is a vector $(\pi_1, \pi_2, \ldots, \pi_{|C|})$ that obeys all the constraints (6) and (7). We denote the objective value (5) of the solution yielded by π for a given instance (C, I, b) as $v(\pi)$.

2.2. Superadditive functions, maximal solutions

Throughout the paper, we use the notions of *superadditivity* and *maximality*. We now discuss on their relation with dual solutions.

Definition 1. *A function f is superadditive if for all x, y it holds that $f(x)+f(y) \leq f(x+y)$.*

By abuse of language, we shall say increasing and superadditive dual solutions if their coefficients can be obtained by applying a monotonously increasing and superadditive function. Other attributes are used similarly. A function that leads to a dual solution is said dual-feasible (Johnson, 1973). Dual-feasible functions (DFF) are generally defined from $[0, 1]$ to $[0, 1]$. In the linear programming relaxation of the CSP, a solution maps values of item sizes in $\{1, \ldots, C\}$ into dual values in $[0, 1]$. We could also define a function from $[0, 1]$ to $[0, 1]$ by scaling the item sizes, *i.e.*, dividing the item sizes by the roll sizes, like in Fekete and Schepers (2001).

However, for the sake of simplicity, we will define dual-feasible functions from $\{1, \ldots, C\}$ to $[0, 1] \cap \mathbb{Q}$, and thus dual values and DFFs will be written in the same fashion. This restriction yields a discrete dual feasible function. In the remainder, we only consider discrete dual feasible functions.

A dual solution π is dominated by another solution π' if it cannot lead to a better solution for any data. If a dual solution is not dominated by any other, it is maximal. A corresponding concept has been defined for DFF by Carlier and Néron (2007). The following proposition is a slight rewriting of their result.

Proposition 1. *(Theorem 1. of Carlier and Néron (2007)). A dual-feasible function is* maximal *if and only if $f(C) = 1$, f is monotonously increasing, f is superadditive, and f is such that for all $i = 1, \ldots, C-1$ it holds that $f(i) + f(C-i) = 1$.*

If a solution π obeys for all $i = 1, \ldots, C-1$ the condition $f(i) + f(C-i) = 1$, and if $\pi_C = 1$, it is called symmetric.

A dual solution is *maximal* if and only if there is a maximal DFF (MDFF) f such that $f(i) = \pi_i$ for all i of I and all b_i. Nemhauser and Wolsey (1998) describe conditions that characterize extreme points of the knapsack polytope. Not all maximal solutions are extreme points of the polyhedron. This means that even when we ensure that the considered solutions are maximal, there are many solutions that are not useful. This is confirmed by the

experimentations of Carlier and Néron (2000, 2007) in the context of cumulative scheduling problems.

All item sizes i in $\{1, \ldots, C\}$ are considered, even if $b_i = 0$. Practically speaking, only the dual variables π_i related to sizes of positive demand are taken into account in the cost function. In order to remove symmetries, we will consider only solutions that are dominant with respect to the following dominance criterion: dual variables π_i such that $b_i = 0$ are chosen as small as possible if $i < C/2$, and as large as possible if $i > C/2$.

Proposition 2. *For a given demand vector b, there is a dominant set of optimal dual solutions such that, if $\widehat{i}$ is the smallest index such that $b_{\widehat{i}} > 0$*

1. $\pi_i = 0$ *if* $i < \widehat{i}$
2. $\pi_i = \max_{j=1,\ldots,i-1}\{\pi_j + \pi_{i-j}\}$ *if* $b_i = 0$ *and* $i \geq \widehat{i}$
3. $\pi_i = 1 - \pi_{C-i}$ *if* $b_i = 0$ *and* $i \geq C/2$

Proof. We have to show that for any maximal dual solution π, there is an equivalent dual solution π' such that π' is dominant with respect to the properties above, and $\sum_{i=1,\ldots,C} b_i \pi_i \leq \sum_{i=1,\ldots,C} b_i \pi'_i$.

We will show this result by constructing step by step a dominant solution from any initial maximal dual solution.

A first step consists in setting all dual values of indices i strictly less than $\widehat{i}$ to zero, and their symmetric π_{C-i} to one. This can be done safely: the solution obtained is clearly increasing and symmetric, and the superadditivity is conserved, since all the greatest values are increased, and all the smallest values are decreased.

Let π be a maximal solution (increasing, superadditive and symmetric), and let i^* be the smallest value such that $b_{i^*} = 0$ and $\pi_{i^*} > \max_{i+j=i^*}\{\pi_i + \pi_j\}$. If i^* does not exist, or if $i^* > C/2$, the solution is dominant.

If i^* exists and $i^* < C/2$, consider the following dual vector π': $\pi'_i = \pi_i$ if $i \notin \{i^*, C - i^*\}$, $\pi'_{i^*} = \max_{i+j=i^*}\{\pi_i + \pi_j\}$, and $\pi'_{C-i^*} = 1 - \pi'_{i^*}$. We show that π' is a maximal dual solution.

By construction, this solution is symmetric, and increasing. The only decreased value is π'_{i^*}. By construction, $\pi'_{i^*} \geq \pi'_i + \pi'_{i^*-i}$ for any $i = 1, \ldots, i^* - 1$. The only increased value is π'_{C-i^*}. We have to show that $\pi'_{C-i^*} + \pi_i \leq \pi_{C-i^*+i}$ for $i = 1, \ldots, i^*$. By construction, we have $\pi'_{i^*-i} + \pi'_i \leq \pi'_{i^*}$ for $i = 1, \ldots, i^*$. By symmetry, we have $\pi'_{i^*-i} = 1 - \pi'_{C-i^*+i}$ and $\pi'_{i^*} = 1 - \pi'_{C-i^*}$. Thus we obtain $\pi'_{C-i^*} + \pi'_i \leq \pi'_{C-i^*+i}$, which is the sought result.

By applying this process iteratively while there is such a value i^* in the current dual vector, a maximal dual solution with the dominance property is obtained.

Since only dual variables such that $b_i = 0$ are considered, the cost function is the same as the original (or is greater if some $b_{C-i} > 0$ for some modified value i). □

2.3. Dual cuts

In order to speed up the column generation procedure for solving the CSP, Carvalho (2005) added cuts that exclude solutions that are not superadditive or not increasing. Only a subset of such cuts were considered, because they are exponential in number. The dual cuts added were the following.

$$\begin{aligned} \pi_i &\leq \pi_j \quad \text{for } i < j \\ \pi_i + \pi_j &\leq \pi_{i+j} \text{ for } i, j \leq C, i + j \leq C \end{aligned}$$

Dual cuts can be seen as constraints that avoid oscillations of the dual variables. From a primal point of view, they can also be seen as *exchange vectors* (e.g. Perrot, 2005), which allow to implicitly generate a whole set of primal columns from the current set of columns available.

2.4. Bounds on the dual values (Caprara et al., 2005)

In the remainder, several results depend on the fact that we know initial lower and upper bounds for the dual values. The following bounds are valid for any maximal dual solution. This result is rewriting of a result stated by Caprara et al. (2005).

Proposition 3. *(originally stated by Caprara et al. (2005))*

The following bounds are valid for any maximal dual solution of (6)-(7). If C is odd, the value $\pi_{C/2}$ does not exist and the corresponding constraint has to be removed.

$$\begin{aligned} 0 \leq \pi_i \leq \frac{1}{\lfloor C/i \rfloor}, &\quad \textit{for } 0 < i < C/2 \\ \pi_{C/2} = 1/2 & \\ 1 - \frac{1}{\lfloor C/(C-i) \rfloor} \leq \pi_i \leq 1, &\quad \textit{for } C/2 < i < C \\ \pi_C = 1 & \end{aligned}$$

Proof. Special values. We have $\pi_C = 1$ by symmetry (since $\pi_0 = 0$). We also have $\pi_{C/2} = 1/2$ by symmetry ($\pi_{C/2} + \pi_{C/2} = 1$).

Values $1, \ldots, C/2 - 1$. The lower bound is due to constraint (7). The upper bound is due to the fact that the pattern with $\lfloor C/i \rfloor$ items of size i is valid. It follows that π_i cannot be larger than $\frac{1}{\lfloor C/i \rfloor}$ (otherwise constraint 6 would be violated).

Values $C/2 + 1, \ldots, C - 1$. The lower and upper bounds are deduced by symmetry from the lower and upper bounds of the values less than $C/2$. □

The upper bounds are obtained by the fact that at most $\lfloor C/i \rfloor$ items of size i can be cut from a roll of size C. The lower bounds are deduced from the upper bounds by symmetry of a maximal solution.

In the sequel we will use respectively l_i and u_i for a lower bound and an upper bound of the value of π_i.

3. New dual cuts

In this section, we introduce new dual cuts, which can be applied to a linear program for solving the CSP. The first cut we propose is similar to those of Carvalho (2005): it ensures that the dual solution will be *symmetric*.

The other cuts are based on a different idea, and define a new family of cuts. The motivation is to avoid solutions that are linear combinations of known solutions, even if they are maximal. Whereas the cuts of Carvalho (2005) leave at least one optimal solution, our procedures cut off all dual solutions with a given structure, of which only one needs to be checked in order to see whether the optimal solution is among these.

3.1. A first family of cuts

The dual cuts proposed by Carvalho (2005) exclude dual solutions that are not increasing and superadditive. Following the characterization of Nemhauser and Wolsey (1998), we can also cut solutions that are not *symmetric* (Cf. Proposition 1). This would lead to the following family of cuts.

Proposition 4. *A valid cut is given by*

$$\pi_i + \pi_{C-i} = 1.$$

Proof. If $\pi_i + \pi_{C-i} < 1$, then π is not maximal, since it is not symmetric. Consequently, this solution can be safely cut. □

When C is large, there is a small chance that there are two such items of size i and $C-i$ such that $b_i > 0$ and $b_{C-i} > 0$, even if a suitable preprocessing method is applied. Practically speaking, we use a slight generalization of the cuts above.

Proposition 5. *The following family of cuts is valid.*

$$\pi_i + \pi_j \geq 1, \forall i, j \in I : i + j \geq C \tag{8}$$

Proof. Using Proposition 4, we know that $\pi_i + \pi_{C-i} = 1$. Since dominant dual values follow an increasing rule, $\pi_j \geq \pi_{C-i}$ and thus $\pi_i + \pi_j \geq 1$. □

This family of cuts is straightforward to apply. When combined with the cuts of Carvalho (2005), cuts (8) may lead to improved results.

3.2. A general result

In the remainder of this section, our goal is to exclude dual solutions that are linear combinations of a given dual solution π' and another one. First we state general results, which are independent of any chosen solution π'. This leads to a general framework for deriving cuts. Then two specific solutions π^0 and π^1 are studied, and more specific cuts are derived.

An issue is that such cuts may also cut π'. This means that the value of an optimal solution of the problem (5)–(7) found when these cuts are applied may be strictly less than the value of an actual optimal solution. If that happens, π' is an optimal solution. The following general proposition ensures that these cuts are safe if π' has been recorded.

Let π' be a dual solution of the CSP and E be a cut that excludes only π' and a set of dual solutions that can be expressed as a convex combination of π' and another dual solution, which is not cut by E.

Proposition 6. *If OPT is the optimal solution value of the LP (1)–(3) and $\widehat{OPT}$ the optimal solution value of the LP obtained when E is applied, then $OPT = max\{v(\pi'), \widehat{OPT}\}$.*

3.2.1. Characterization of dominated solutions

Using the general result of Proposition 6, powerful cuts can be derived if one is able to characterize solutions that are convex combinations of some other solutions. Before showing

how that can be done, we first state a result that helps us proving that a solution is the linear combination of two other solutions, and thus cannot lead to a better result.

Lemma 1. *Let π be a maximal solution of (5)–(7). Solution π is dominated if and only if there exists a maximal dual solution π' of (5)–(7) different from π such that*

$$\pi_i = 0 \implies \pi'_i = 0 \tag{9}$$

$$\pi_i + \pi_j = \pi_{i+j} \implies \pi'_i + \pi'_j = \pi'_{i+j} \tag{10}$$

Proof. $\Rightarrow$: If π is dominated, then there are two maximal dual solutions π' and $\widehat{\pi}$ such that $\lambda_1\pi' + \lambda_2\widehat{\pi} = \pi$ $(\lambda_1, \lambda_2 \in]0, 1[,\ \lambda_1 + \lambda_2 = 1)$. Because π' and $\widehat{\pi}$ are nonnegative, the implication (9) is obvious. Now suppose that $\pi_i + \pi_j = \pi_{i+j}$ and $\pi'_i + \pi'_j < \pi'_{i+j}$. In this case, since $\widehat{\pi}_i + \widehat{\pi}_j \leq \widehat{\pi}_{i+j}$, we obtain $\pi_i + \pi_j = \lambda_1\pi'_i + \lambda_1\pi'_j + \lambda_2\widehat{\pi}_i + \lambda_2\widehat{\pi}_j < \lambda_1\pi'_{i+j} + \lambda_2\widehat{\pi}_{i+j} = \pi_{i+j}$, which is in contradiction with our assumption.

$\Leftarrow$: We first show that if both properties above are satisfied, and for a small-enough positive value ϵ, we can construct $\widehat{\pi} := (\pi - \epsilon\pi')/(1 - \epsilon)$ a maximal dual solution, and thus a valid solution of (5)–(7).

It is sufficient to show that $\widehat{\pi}$ is superadditive, and that $\widehat{\pi}_i \geq 0$ for all i (if the two properties are true, the solution must be increasing, and symmetric). For the sake of simplicity, we make the proof with $\pi - \epsilon\pi'$, which is sufficient to show that $\widehat{\pi}$ is superadditive.

The solution $\pi - \epsilon\pi'$ is positive, since by assumption, there is no value i such that $\pi_i = 0$ and $\pi'_i \neq 0$. Since the dual values are discrete, one can always find a small enough ϵ to ensure that $\pi - \epsilon\pi'$ is always positive.

Now we prove that the $\pi - \epsilon\pi'$ is also superadditive. For all i, j such that $\pi_i + \pi_j < \pi_{i+j}$, since the dual values are discrete, one can always find a sufficient small ϵ to ensure that $\pi_i - \epsilon\pi'_i + \pi_j - \epsilon\pi'_j \leq \pi_{i+j} - \epsilon\pi'_{i+j}$. For i, j such that $\pi_i + \pi_j = \pi_{i+j}$, the relation is satisfied, since in this case $\pi'_i + \pi'_j = \pi'_{i+j}$.

We have shown that $\widehat{\pi}$ is maximal if the properties above are satisfied. By construction, $\widehat{\pi}_C = 1$. It follows that π can be rewritten as $\epsilon\pi' + (1 - \epsilon)\widehat{\pi}$, with π' and $\widehat{\pi}$ two maximal dual solutions. So π can be expressed as the linear combination of two other dual solutions, and thus is dominated.

□

3.2.2. Deriving the cuts

In the following we relax the conditions of Lemma 1 so they can be expressed as linear constraints. Constraints (11) and (12) below are respectively related to constraints (9) and (10).

For condition (12), we consider any pair of values such that $\pi_j + \pi_k < \pi_{j+k}$ in π and compute a lower bound for a weighted sum of the dual values if $\pi_j + \pi_k = \pi_{j+k}$ in a dual solution. This bound is obtained by improving the lower bounds of π_j and π_k, and then by propagating these lower bounds using the superadditivity.

Let π be a maximal dual solution defined for values $\{1, \ldots, C\}$ and i^* be the smallest value smaller than or equal to $C/2$ such that $\pi_{i^*} > 0$ in π. If i^* does not exist (which only happens in one specific case when C is odd), constraint (11) can be forgotten, since in this case, condition (9) is always false. Let also Δ be the set of pair of values (j, k) such that $\pi_j + \pi_k < \pi_{j+k}$ in π. In the following, we use the values l_i and u_i, the bounds defined in Subsection 2.4. Without loss of generality, we consider that $j \leq k$.

Condition (11) is necessary and sufficient to have (9) violated, since $\pi'_{i^*} = 0$ implies that there is a value such that $\pi'_{i^*} = 0$ and $\pi_{i^*} > 0$.

Note that in the following, the values $\widehat{t}_i^{(j,k)}$ have to be computed by increasing value of i. We also recall that $I = \{1, \ldots, C\}$.

In the special case where $l = k$, the cases $i = j$ and $i = k$ are equivalent, and the case $j < i < k$ never occurs.

Proposition 7. *If π does not yield the optimal solution of the problem (5)–(7), there exists an optimal solution π' such that either*

$$\pi'_{i^*} = 0 \tag{11}$$

or the following dual cut is valid, for any set of nonnegative parameters λ_i, $i \in I$:

$$\sum_{i \in I} \pi'_i \lambda_i \geq \min_{(j,k) \in \Delta} \left\{ l_{j+k} \lambda_{j+k} + \sum_{i \in I \setminus \{j,k\}} \widehat{t}_i^{(j,k)} \lambda_i \right\} \tag{12}$$

with

$$\widehat{t}_i^{(j,k)} = \begin{cases} l_i & \text{if } i < j \\ max\left\{l_i, l_{j+k} - u_j\right\} & \text{if } i = j \\ max\left\{l_i, \underset{l+m=i}{max}\{\widehat{t}_l^{(j,k)} + \widehat{t}_m^{(j,k)}\}\right\} & \text{if } j < i < k \\ max\left\{l_i, \underset{l+m=i}{max}\{\widehat{t}_l^{(j,k)} + \widehat{t}_m^{(j,k)}\}, l_{j+k} - u_k\right\} & \text{if } i = k \\ max\left\{l_i, \underset{l+m=i}{max}\{\widehat{t}_l^{(j,k)} + \widehat{t}_m^{(j,k)}\}\right\} & \text{if } i > k \end{cases}$$

Proof. We have to show that any solution that is cut is also a linear combination of π and another dual solution. It will follow that the cut is safe if π does not yield the optimal solution.

First we show that the value $\widehat{t}_i^{(j,k)}$ is a valid lower bound for the value that π'_i can take when $\pi'_j + \pi'_k = \pi'_{j+k}$. By the initial construction, $\widehat{t}_i^{(j,k)} \geq l_i$. Since $\pi'_j + \pi'_k = \pi'_{j+k}$, we know that $\pi'_k \geq l_{j+k} - u_j$. For $i = j$, a similar deduction can be made. For any value greater than j, the validity of $\widehat{t}_i^{(j,k)}$ is due to the superadditivity of a maximal dual solution.

For a given pair (j,k), $l_{j+k}\lambda_{j+k} + \sum_{i \in I \setminus \{j,k\}} \widehat{t}_i^{(j,k)} \lambda_i$ is a lower bound for $\sum_{i \in I} \pi'_i \lambda_i$ if $\pi'_j + \pi'_k = \pi'_{j+k}$. We know that at least one pair (j,k) of Δ is such that $\pi'_j + \pi'_k = \pi'_{j+k}$. We deduce that $\sum_{i \in I} \pi'_i \lambda_i$ must be at least equal to the smallest possible value related to a pair (j,k) of Δ.

This means that the right-hand term of Equation (12) is a lower bound for $\sum_{i \in I} \pi'_i \lambda_i$ when there is at least one pair (j,k) in Δ such that $\pi'_j + \pi'_k = \pi'_{j+k}$. Consequently, in any solution π' that does not respect constraint (12), there is no pair (j,k) of Δ such that $\pi'_j + \pi'_k = \pi'_{j+k}$.

Using Lemma 1, we deduce that if a solution does not respect any of the two constraints (11) and (12), it is a linear combination of π and another solution. According to the initial assumption, any such solution can be safely cut.

□

Note that these cuts may exclude the dual solution π.

This leads to a general three-step method for deriving cuts when a dual solution π is known (take one of those surveyed in Clautiaux et al. (2008) for example). Since we know that at least one of the two conditions has to hold, it means that either both hold (step 3), or only (11) holds (step 5), or only (12) holds (step 8). The method is described by Algorithm 1, where i^* is defined as above.

Algorithm 1: A generic method for applying dual cuts that exclude a given dual solution

```
1 Create the original LP ;
2 Apply the cuts (11) and (12) to the LP ;
3 Solve the LP obtained ;
4 Remove the cuts (12) from the LP ;
5 Solve the LP obtained ;
6 Remove the cut (11) from the LP;
7 Add the cut π_{i*} > 0 and the cuts (12) to the LP ;
8 Solve the LP obtained ;
9 Return the maximum value obtained ;
```

The quality of the cuts depends on the choice of initial dual solution and on the values λ_i. Several methods can be used to compute a suitable set of coefficients. We describe in the computational experiments parts the technique we used.

3.3. The solution π^0

Any dual solution can be used in the framework described above. When specific dual solutions are considered, more powerful cuts can be derived. First we study a dual solution with a simple structure. Once again, if C is odd, the case $C/2$ just has to be forgotten.

- $\pi_i = 0$ for $i < C/2$
- $\pi_{C/2} = 1/2$
- $\pi_i = 1$ for $i > C/2$

The value related to this solution is equal to the number of items of size greater than $C/2$ plus the number of items of size $C/2$ divided by two. We name this dual solution π^0. We chose this solution for its simple structure, and for it would lead to the most unbalanced solutions, in terms of the values of the dual variables.

3.3.1. Characterization of solutions combined with π^0

Proposition 8. *If π is a maximal dual solution of (5)–(7) such that, for $i = 1, \ldots, \lceil C/2 \rceil - 1$, it holds that $\pi_i < i/C$, then π is a convex combination of π^0 and another dual solution.*

Proof. We use Lemma 1 to show this result. We have to show that conditions (9) and (10) are verified, where π^0 plays the role of π' and π is defined as above.

Note that when C is odd, some cases cannot happen, which does not change the validity of the proof.

Since $\pi^0_i = 0$ for all $0 < i < C/2$, and since for any maximal dual solution π, $\pi_i > 0$ if $i \geq C/2$, condition (9) is verified.

Now we show that condition (10) is also verified, *i.e.* for all pairs i, j such that $\pi_i + \pi_j = \pi_{i+j}$, we have $\pi^0_i + \pi^0_j = \pi^0_{i+j}$. Assume without loss of generality $x \leq y$. The proof consists in five cases, which cover all possibilities. Three are related to cases where $\pi^0_i + \pi^0_j$ is always equal to π^0_{i+j}, the others are related to cases where $\pi_i + \pi_j$ cannot be equal to π_{i+j}.

In the three following cases, $\pi^0_i + \pi^0_j$ is always equal to π^0_{i+j}.

1. If $i + j < C/2$ then $\pi^0_i + \pi^0_j = 0 + 0 = 0$ and $\pi^0_{i+j} = 0$.
2. If $i < C/2 < j$ and $i + j \leq C$ then $\pi^0_i + \pi^0_j = 0 + 1 = 1$ and $\pi^0_{i+j} = 1$.
3. If $i = j = C/2$ then $\pi^0_i + \pi^0_j = 1/2 + 1/2 = 1$ and $\pi^0_{i+j} = \pi^0_C = 1$.

In the two following cases, $\pi_i + \pi_j$ cannot be equal to π_{i+j}.

1. If $i < C/2$, $j < C/2$, and $i + j = C/2$, by assumption $\pi_i + \pi_j < i/C + j/C = 1/2$ and since π is maximal (and then symmetric), $\pi_{i+j} = \pi_{C/2} = 1/2$.
2. If $i < C/2$, $j \leq C/2$, and $i + j > C/2$, $\pi_i + \pi_j < i/C + j/C$. By symmetry, $\pi_{i+j} = 1 - \pi_{C-i-j}$. Since $C - i - j < C/2$, we have $\pi_{C-i-j} < 1 - \frac{i+j}{C}$. Consequently, $\pi_{i+j} > i/C + j/C$.

□

We use Proposition 8 to detect dual solutions that are dominated by π^0 and another solution.

3.3.2. Deriving the cuts

Cuts can be obtained using the general result of Proposition 7, but since π^0 has a specific structure, a hand-tailored method can be designed. The idea is to cut some solutions where there are no values $i < C/2$ such that $\pi_i \geq i/C$. This cannot be verified directly by the mean of a linear cut, so we study the different values that can be taken by the sum of the dual values.

There must be one value $j < C/2$ such that $\pi_j \geq j/C$. Our idea is to consider the weighted sum of the dual values for each possible j. For this purpose, we compute the minimum value that π_i can take when $\pi_j = j/C$. All values π_i with $i < j$ can be equal to 0, whereas the values larger than j have to follow the superadditivity rule.

Proposition 9. *Let $I_1 = \{1, \ldots, \lfloor C/2 \rfloor\}$. If $v(\pi^0) < OPT$ then for any nonnegative values $\lambda_i, i \in I_1$, the following cut is valid:*

$$\sum_{i \in I_1} \pi_i \lambda_i \geq \min_{j \in I_1, b_j > 0} \left\{ \sum_{i \in I_1} \lfloor \frac{i}{j} \rfloor * \frac{j \lambda_i}{C} \right\} \tag{13}$$

Proof. Let $\widehat{x}_i^j := \lfloor i/j \rfloor * j/C$. Proposition 8 tells us that there must be at least one item $j \in I_1$ such that $\pi_j \geq j/C$. For a given $j \in I$, $\widehat{x}_i^j$ is the minimum value that π_i can take in a maximal solution when $\pi_j = j/C$ (this bound is obtained by considering the superadditivity only). Consequently $\sum_{i \in I_1} \widehat{x}_i^j \lambda_i$ is a lower bound for the weighted sum of the dual values when $\pi_j = j/C$. If one tests all possibilities for j, and keeps the minimum sum, a lower bound for $\sum_{i \in I_1} \pi_i \lambda_i$ is obtained. Only values j such that $b_j > 0$ are considered. By dominance of π (see Proposition 2), $\pi_j \geq j/C$ with $b_j = 0$ will occur only if there is at least one i such that $i < j$ and $\pi_i \geq i/C$, a case that is already considered. □

For a given value of C, a given demand vector, and a given set of parameters, the right-hand part of the inequality can be computed only once for all executions. When such a cut is applied, one has to be aware that solution π^0 is also cut. For the method to remain exact, $v(\pi^0)$ has to be recorded.

An issue is to find an interesting set of parameters λ_i. We set $\lambda_j = 0$ for any j such that $b_j = 0$. Our strategies are described in the computational experiments section.

3.4. The solution π^1

We now consider the following dual solution, which we name π^1.

- $\pi_i = 0$ if $i \leq C/3$
- $\pi_i = 1/2$ if $i \in]C/3, 2C/3[$
- $\pi_i = 1$ if $i \geq 2C/3$

In the following, we study this dual solution, and derive dual cuts to exclude solutions that are obtained by linear combination of π^1 and another solution. This is more difficult than with π^0, since its structure is slightly more complex. A three-step method is used, whereas only one step was needed for π^0.

3.4.1. Characterization of solutions combined with π^1

Proposition 10. *Let π be a maximal dual solution. If $i \leq C/3$ implies $\pi_i < 3i/4C$, and $i \in]C/3, C/2[$ implies $\pi_i \geq 3i/4C$, then π is a convex combination of π^1 and another dual solution.*

Proof. We use Lemma 1 to show this result. We show that the two conditions of Lemma 1 are verified, where π^1 plays the role of π' and π is defined as above.

We have $\pi^1_i = 0$ for $i \leq C/3$. For i in $]C/3, C/2[$, π_i is strictly greater than 0 by assumption. Consequently the first condition of Lemma 1 is verified.

Now we show that for each pair i, j such that $\pi_i + \pi_j = \pi_{i+j}$ it holds that $\pi^1_i + \pi^1_j = \pi^1_{i+j}$. Without loss of generality, we assume that $i \leq j$ and $i + j \leq C$. Nine cases have to be considered.

In the four following cases, $\pi^1_i + \pi^1_j$ is always equal to π^1_{i+j}.

1. If $i \leq j \leq C/3$ and $i + j \leq C/3$ then $\pi^1_i + \pi^1_j = 0 + 0 = 0 = \pi^1_{i+j}$.
2. If $i \leq C/3 < j < 2C/3$ and $i + j \in]C/3, 2C/3[$ then $\pi^1_i + \pi^1_j = 0 + 1/2 = 1/2 = \pi^1_{i+j}$.
3. If $C/3 < i \leq j < 2C/3$ then $\pi^1_i + \pi^1_j = 1/2 + 1/2 = 1 = \pi^1_{i+j}$ because $i + j > 2C/3$.
4. If $i \leq C/3$ and $j \geq 2C/3$ then $i + j \geq 2C/3$, $\pi^1_i + \pi^1_j = 0 + 1 = 1$ and $\pi^1_{i+j} = 1$.

In the five following cases, $\pi_i + \pi_j$ cannot be equal to π_{i+j}.

1. If $i \leq j \leq C/3$ and $i + j \in]C/3, C/2]$ then $\pi_i + \pi_j < 3i/4C + 3j/4C$ and $\pi_{i+j} \geq 3(i + j)/4C$.
2. If $i \leq j \leq C/3$ and $i + j \in]C/2, 2C/3[$ then $\pi_i + \pi_j < 3(i + j)/4C$. Since $i + j \leq 2C/3$, this is smaller than $3(2C/3)/4C = 1/2$ and $\pi_{i+j} \geq 1/2$ since π is a maximal dual solution.
3. If $i = j = C/3$ then $i + j = 2C/3$. Since $\pi_{C/3} < 1/4$, $\pi_{C/3} + \pi_{C/3} < 1/2$ and by symmetry $\pi_{2C/3} = 1 - \pi_{C/3} > 3/4$.

4. If $i \leq C/3 < j \leq C/2$ and $i + j \geq 2C/3$, we have $\pi_j \leq 1/2$ since $j \leq C/2$ and π is nondecreasing and symmetric. Thus $\pi_i + \pi_j < 3i/4C + 1/2 \leq \frac{3}{4C}\frac{C}{3} + 1/2 = 3/4$ and by symmetry $\pi_{i+j} = 1 - \pi_{C-i-j} > 1 - \frac{3(C-i-j)}{4C} = \frac{C+3(i+j)}{4C} = 1/4 + \frac{3(i+j)}{4C} \geq 1/4 + \frac{3(2C/3)}{4C} = 1/4 + 1/2 = 3/4$.

5. If $i \leq C/3$, $j \in]C/2, 2C/3[$ and $i + j \geq 2C/3$, $\pi_i < \frac{3i}{4C}$ and by symmetry $\pi_j = 1 - \pi_{C-j}$. Since $C - j \in]C/3, C/2[$, $\pi_{C-j} \geq 3(C-j)/4C$ and thus $\pi_j \leq 1 - \frac{3(C-j)}{4C}$. Consequently, $\pi_i + \pi_j < \frac{3i}{4C} + 1 - \frac{3(C-j)}{4C} = \frac{C+3i+3j}{4C}$ and by symmetry $\pi_{i+j} > 1 - \frac{3(C-i-j)}{4C} = \frac{C+3i+3j}{4C}$.

□

3.4.2. Deriving the cuts

There is no straightforward way of deriving a cut from this property, since the condition to be verified is $\exists i \leq C/3, \pi_i \geq 3i/4C$ **or** $\exists i \in]C/3, C/2[, \pi_i < 3i/4C$. This means that three non-dominated cases are possible: the first condition is verified, the second condition is verified, or both are verified.

This leads to a three-step method, where each step is related to a possible case. First we assume that there are no values $i \leq C/3$ such that $\pi_i \geq 3i/4C$ (only condition 2 has to be verified). We add the corresponding cuts. When the optimal solution under this first assumption is found, cuts are added to ensure that the first condition is also respected, and the program rerun (conditions 1 and 2 have to be verified). Then the cuts added in the first step are removed, and replaced by a cut that can be derived from the fact that there is at least one value $i \leq C/3$ such that $\pi_i \geq 3i/4C$ (now only condition 1 has to be verified). The method is summed up in Algorithm 2.

The cuts we derived are based on the following propositions, whose proofs are omitted, since they are similar to Proposition 9.

Note that $\widehat{y}_i^j$ and $\widehat{z}_i^j$ have to be computed by increasing values of i and for all values in $1, \ldots, C/2$.

Proposition 11. *If the optimal solution is such that there are no items $i \leq C/3$ such that $\pi_i \geq 3i/4C$, and if $v(\pi^1) < OPT$, then for any nonnegative values $\lambda_1, \ldots, \lambda_k$, and I_1 defined as above,*

$$\sum_{i \in I_1} \pi_i \lambda_i \leq \max_{j \in]C/3, C/2[, b_j > 0} \left\{ \sum_{i \in I_1} \widehat{y}_i^j \lambda_i \right\} \tag{14}$$

is a valid dual cut, where $\widehat{y}_i^j$ *is defined as follows:*

$$\widehat{y}_i^j = \begin{cases} \frac{3j/4C}{\lfloor j/i \rfloor} & \text{if } i < j \\ 3j/4C & \text{if } i = j \\ 1/\lfloor \frac{C}{i} \rfloor & \text{if } i > j \end{cases}$$

Proposition 12. *If the optimal solution is such that there is an item* i *of* $[1, C/3]$ *such that* $\pi_i \geq 3i/4C$*, then for any positive values* $\lambda_1, \ldots, \lambda_k$*, and* I_1 *defined as above,*

$$\sum_{i \in I_1} \pi_i \lambda_i \geq \min_{j \in [1, C/3], b_j > 0} \left\{ \sum_{i \in I_1} \widehat{z}_i^j \lambda_i \right\} \tag{15}$$

is a valid dual cut, where $\widehat{z}_i^j$ *is defined as follows:*

$$\widehat{z}_i^j = \begin{cases} 0 & \text{if } i < j \\ 3j/4C & \text{if } i = j \\ \lfloor i/j \rfloor * 3j/4C & \text{if } i > j \end{cases}$$

Algorithm 2: Applying the cuts (14) and (15)

Compute $v(\pi^1)$;
Add the cut $\sum_{i \in I} \pi_i b_i > v(\pi^1)$;
// Assume that for all $i \leq C/3$, $\pi_i < 3i/4C$ *and there is a* $C/2 > j > C/3$ *such that* $\pi_j < 3j/4C$
Add the cuts $\pi_i < 3i/4C$, $\forall i \leq C/3$;
Add the cuts (14) ;
Run the column-generation procedure ;
// Assume that there is an $i \leq C/3$*, such that* $\pi_i \geq 3i/4C$ *and a* $C/2 > j > C/3$ *such that* $\pi_j < 3j/4C$
Remove the cuts on the $i \leq C/3$;
Add the cuts (15) ;
Resume the column-generation method ;
// Assume that there is an $i \leq C/3$*, such that* $\pi_i \geq 3i/4C$ *and for all* $C/2 > i > C/3$, $\pi_i \geq 3i/4C$
Remove the cuts (14) ;
Add the cut $\pi_i \geq 3i/4C$, $\forall C/3 < i < C/2$;
Resume the column-generation method ;
// Let OPT *be the maximum value obtained*
return $max\{v(\pi^1), OPT\}$;

4. Bounding the dual variables

Another way of taking into account the data is to consider an estimation for the number of stock rolls needed. In this section, we propose to use this information to tighten the bounds u_i and l_i for each dual variable π_i. We also describe how variables related to small values can be set to zero. Finally we show that limiting the search space to the neighborhood of an initial *good* dual solution improves previous results.

We recall that we consider in this paper that $I = \{1, \ldots, C\}$ (with some b_i equal to 0). In the sequel, $OPT(I)$ is the optimal value of a solution for the set I, LB is a lower bound for $OPT(I)$ and UB an upper bound for this value.

4.1. New bounds

The first method is a lifting procedure for the lower bounds l_i associated with the dual values π_i. It is based on a trial-and-error procedure. A solution is chosen by setting the dual variable π_i to its minimum l_i and all other dual variables to their maximum u_i. If the value of (possibly invalid) solution is smaller than a known lower bound, l_i is not valid, and it can be updated. This reasoning may lead to better results if the superadditivity constraints are considered.

For a real number α in $[l_i, u_i]$, let $\widehat{l}_i^{(j,\alpha)}$ be a lower bound for the value π_i when $\pi_j = \alpha$. It is defined as follows:

$$\widehat{l}_i^{(j,\alpha)} = \begin{cases} l_i & \text{if } i < j \\ \alpha & \text{if } i = j \\ max\Big\{l_i, \underset{k+m=i}{max}\{\widehat{l}_k^{(j,\alpha)} + \widehat{l}_m^{(j,\alpha)}\}\Big\} & \text{if } j < i < C - j \\ 1 - \alpha & \text{if } i = C - j \\ max\Big\{l_i, 1 - \frac{\alpha}{\lfloor \frac{j}{C-i} \rfloor}, \underset{k+m=i}{max}\{\widehat{l}_k^{(j,\alpha)} + \widehat{l}_m^{(j,\alpha)}\}\Big\} & \text{if } C - j < i \end{cases}$$

For the sake of simplicity, we also introduce an additional notation $\widehat{u}_i^{(j,\alpha)} = 1 - \widehat{l}_{C-i}^{(j,\alpha)}$, which is the equivalent upper bound obtained by symmetry. Now we define $\theta^j(\alpha)$, a function that associates with the value α the corresponding upper bound on the value of a solution obtained when $\pi_j = \alpha$. This bound is based on $\widehat{l}_i^{(j,\alpha)}$ described above, and on the symmetry of a MDFF.

$$\theta^j(\alpha) = b_C + \sum_{i=1}^{\lfloor C/2 \rfloor} max\{b_i \widehat{u}_i^{(j,\alpha)} + b_{C-i} \widehat{l}_{C-i}^{(j,\alpha)}, b_i \widehat{l}_i^{(j,\alpha)} + b_{C-i} \widehat{u}_{C-i}^{(j,\alpha)}\}$$

Lemma 2. *For $j < C/2$, $\pi_j = \alpha$ implies $\sum_{i \in I} \pi_i b_i \leq \theta^j(\alpha)$.*

Proof. We prove this result in two steps. First we prove that $\widehat{l}_i^{(j,\alpha)}$ is a lower bound for π_i when $\pi_j = \alpha$. Then we will deduce that the inequality is valid.

For any value $i \leq j$, $\widehat{l}_i^{(j,\alpha)}$ is a valid lower bound by construction. If $i = C - j$, we have $\pi_i = 1 - \alpha$ by symmetry. For any value $i > j$, by superadditivity, we have $\widehat{l}_i^{(j,\alpha)} \geq \underset{k+m=i}{max}\{\widehat{l}_k^{(j,\alpha)} + \widehat{l}_m^{(j,\alpha)}\}$. For $C - j < i$, we have $\pi_{C-i} \leq \frac{\alpha}{\lfloor \frac{j}{C-i} \rfloor}$, otherwise π_j could not be equal to α if the coefficients follow a superadditive rule. By symmetry, $\pi_i \geq 1 - \frac{\alpha}{\lfloor \frac{j}{C-i} \rfloor}$.

Now we prove that the inequality is valid. Consider each pair of values i and $C-i$: these values are such that $\pi_i + \pi_{C-i} = 1$. This means that for all i, $\pi_i b_i + \pi_{C-i} b_{C-i} \leq max\{b_i \widehat{u}_i^{(j,\alpha)} + b_{C-i} \widehat{l}_{C-i}^{(j,\alpha)}, b_i \widehat{l}_i^{(j,\alpha)} + b_{C-i} \widehat{u}_{C-i}^{(j,\alpha)}\}$. Indeed, either b_i is greater than b_{C-i} and maximizing the expression consists in setting π_i to its upper bound, or in the other case, π_i has to be set to its lower bound.

The validity of the lemma follows.

□

Lemma 2 leads to the following cut, which can be applied when lower bounds l_i and upper bounds u_i are known for the values π_i.

Proposition 13. *Let LB be a valid lower bound for the CSP. The two following constraints are valid dual cuts for the CSP for each $j < C/2$.*

$$\pi_j \leq max\{\alpha : \alpha \in [l_i, u_i], \theta^j(\alpha) \geq LB\}$$

$$\pi_j \geq min\{\alpha : \alpha \in [l_i, u_i], \theta^j(\alpha) \geq LB\}$$

Proof. Since $\theta^j(\alpha)$ is an upper bound for $\sum_{i \in I} \pi_i b_i$ when $\pi_j = \alpha$, all values α such that $\theta^j(\alpha) < LB$ can be safely excluded from the domain of π_j (since they would lead to a solution value less than a known lower bound). This is the case for all values excluded by the cuts. Therefore, the two cuts are valid. □

4.2. Computational issues

We now discuss computational issues related to Proposition 13. Function θ depends on the demand for each item type: it may not be convex, and thus the dichotomy cannot be used to determine the extrema. We describe two different approaches. One is based on a relaxation, the other is based on a small linear program.

A way of computing a lower bound for α is to solve the following linear program (16)-(22). In this program, the lower bound is LB, l_i and u_i are respectively a constant lower and a constant upper bound for π_i, and b_i is the demand for item i.

$$min\ \pi_j \tag{16}$$

subj.to

$$b_C + \sum_{i \in I \setminus \{C\}} \pi_i b_i \geq LB \tag{17}$$

$$\pi_{i+k} - \pi_i - \pi_k \geq 0 \text{ for } i, k, i+k \in I \tag{18}$$

$$\pi_i - \pi_k \geq 0, \text{ for } i > k \tag{19}$$

$$\pi_i + \pi_{C-i} = 1 \text{ for } i \in I \setminus \{C\} \tag{20}$$

$$\pi_i \geq l_i \text{ for } i \in I \tag{21}$$

$$\pi_i \leq u_i \text{ for } i \in I \tag{22}$$

An upper bound is obtained by replacing the min by a max.

Discussion 1. *If all item sizes are used, the problem solved is close to the model of Dyckhoff (1981) for solving the CSP. This means that the computational effort could be even larger than the one needed for the column-generation. When we only consider item sizes of demand strictly greater than 0, it is slightly greater than the initial solution in the column-generation based method of Carvalho (2005). One way of improving the bound is to add additional rows related to patterns. If too many rows are added, the method could be too much time consuming, so a tradeoff has to be found. For example, in our computational experiments, we already get good results using only a subset of the constraints of (16)-(22).*

Thus a lower bound for the problem can lead to tighter lower and upper bounds for some π_i. Note that increasing a value l_i may help tightening other bounds when it is used in a

new definition of $\widehat{l}_i^{(j,\alpha)}$. So the process can be repeated while a value has been modified in the previous step.

This method can also be used with an upper bound UB. In this case, we assume that UB is equal to the optimal value, and we apply the method. If the program leads to a value less than UB, the assumption is false, and UB can be decremented (dichotomy can also be used to fasten the search).

4.3. Original trust region

In Ben Amor et al. (2006b,a), the authors show that enforcing the dual values to be *near a priori* known optimal dual values fasten tremendously the convergence of the column generation algorithm. Here, we propose to extend this method to the case where a *good* (possibly non optimal) dual solution is known.

Several families of DFF have been proposed in the literature (see Clautiaux et al. (2008)). There is a chance that some optimal dual values are near the values given by the best DFF used. We propose the following algorithm: we apply a set of DFF, and keep the one which leads to the best result, then we assume that the dual values are optimal, and add the corresponding boxes around the dual values and solve the LP. The sizes of the boxes are increased iteratively until they become redundant. At the end, these constraints are removed and the search is resumed. Algorithm 3 describes our method.

Algorithm 3: Forcing an original trust region

1 consider a given CSP instance D ;
2 find the best DFF f available for D ;
3 build a classical Gilmore-Gomory LP for the CSP, with additional box constraints forcing the dual variables to be near to the values given by f ;
4 solve the LP ;
5 for each dual variable lying in the boundary of its corresponding box constraint, increase the size of the box. Goto 4;
6 remove the box constraints;
7 resume the resolution of the LP ;

In our implementation, we start with very small boxes centered around the values given by the best DFF. Whenever a box is found to be too small (after solving the LP, when the corresponding dual variable lies in the boundary of the box), its size is increased by a certain value. This value becomes larger with the number of iterations until these constraints become

redundant. The details of our implementation are given in Section 5. Our computational experiments show that the convergence of this approach depends heavily on the quality of the DFF that is used.

5. Computational Experiments

To evaluate the strength of the new cuts and methods proposed above, we conducted a set of computational experiments on randomly generated cutting stock instances and on hard instances from the literature (Scholl et al., 1997). Our focus is on hard instances, *i.e.* those which are solved in a large amount of time using the standard column generation algorithm. We used instances with a large number of different items, and such that the sizes of the items are small compared to the length of the rolls.

Table 1 illustrates the main characteristics of the instances. Column SET identifies the instance set, C stands for the length of the rolls, MIN and MAX correspond to the size of the smallest and the largest item, respectively, NIT is the average number of different item sizes in the instance and B is the average demand per item size. Each set is composed of 10 different instances. The last set is taken from the literature (Scholl et al., 1997), and it is also composed of 10 instances.

SET	C	MIN	MAX	NIT	B
1	100000	1	50000	200	20
2	100000	1	50000	500	20
3	100000	1	35000	200	20
4	100000	1	35000	500	20
5	100000	1	20000	200	20
6	100000	1	20000	500	20
7	100000	1	10000	200	10
8	100000	1	10000	500	10
9	100000	35000	50000	1000	20
HARD	100000	20000	35000	200	20

Table 1: Set of instances

The algorithms were coded in C++, and the CPLEX 10.2 Callable Library (Ilog, 2006) was used for some of the optimization subroutines. The tests were performed on a PC with an 2.20 GHz Intel Core Duo processor, and 2GB of RAM.

Clearly, given the length of the rolls and the relative size of the items, solving the knapsack pricing subproblems using dynamic programming is not computationally viable. Hence, we solved them using the branch-and-bound algorithm MT1 for knapsack problems, which is due to Martello and Toth (1990).

In the following tables, we compare the main stabilization strategies described in this paper with the standard column generation algorithm with and without the dual cuts proposed by Carvalho (2005). We show how our strategies make column generation converge faster when they are used alone, and together with the dual cuts of Carvalho (2005). The cuts are used according to the method of Carvalho (2005). The first restricted master problem is initialized with a single (artificial) column, and the dual cuts are added prior the resolution of the first restricted master problem. Whenever we considered the specific dual cuts of Valério de Carvalho, we only applied cuts of Type I and II (Carvalho, 2005).

Five new methods were compared to the standard column generation algorithm, and to the stabilization procedure proposed in Carvalho (2005), namely the dual cuts (13), the algorithms 1 and 2, and the methods based on the computation of dual bounds and on a trust region.

The parameters λ_i in the dual cuts (13), and in the algorithms 1 and 2 were chosen in the following way: the first cut was derived with $\lambda_0 = 1$ and $\lambda_i = 0$ for the other items, the second with $\lambda_0 = \lambda_1 = 1$ and $\lambda_i = 0$ for the remaining items, and so on, until all the items are finally considered. Thus, the number of cuts that were generated this way is equal to the cardinality of the items set considered in the respective cut. Other experiments were conducted with other sets of parameters, but none gave better results.

Concerning the implementation of the trust region method, in these experiments, we limited the number of iterations to 10, and we used two dual-feasible functions to define the initial trust region, namely u^k from Fekete and Schepers (2001) and f_2^k from Carlier et al. (2007). Initially, the dual variables are forced to be within a box of size proportional to i/C and centered around the values given by the best dual-feasible function. Whenever a dual variable lies in the boundary of its box, the size of the box is simply increased. Typically, after the 10 iterations, these box constraints become redundant.

The results of the different experiments are given in Tables 2 to 6. These results correspond to the resolution of the linear programming relaxation of the column generation model for the CSP. The execution was stopped after 900 seconds. An instance was considered as *solved* if the optimum was reached within this time limit. The entries in the tables have the

following meaning:

. Strategy: solution method used, for example

  . *standard* denotes the standard column generation algorithm,

  . *with dual cuts (Carvalho, 2005)* means that we added a polynomial set of dual cuts as in Carvalho (2005) before solving the first restricted master,

  . *with dual cuts (13)* means that we used only the described cuts,

  . *Algorithm 1* means that we used this algorithm with no other dual cuts.

. *solved*: number of instances solved within the time limit,

. *itr*: average number of pricing subproblems solved (column generation iterations);

. $\%red.\ itr$: relative reduction in the number of column generation iterations (compared to the standard algorithm);

. t_{tot}: average solution time (in seconds);

. $\%red.\ t_{tot}$: relative reduction in the total solution time (compared to the standard algorithm).

When a method fails to solve a single instance in a set, the respective rows are filled with entries $-$. It is important to note that the averages are computed based on the results of the 10 instances in the set. Therefore, the first criterium to consider when comparing the different approaches should be the number of solved instances.

Our experiments show a clear improvement on convergence when compared to standard column generation and to the method proposed in Carvalho (2005). In fact, the standard cuts of Carvalho (2005) are ineffective for these hard instances. In many cases, the convergence of column generation greatly deteriorates when these cuts are used. The number of instances solved to optimality decreases in many cases, and the average computing time is always worse than standard column generation, except for the HARD instances.

In general, the number of instances solved up to optimality within the time limit increases significantly when the stabilization strategies described in this paper are considered. In terms of the computing time required to solve the instances, the improvement is also impressive. In some cases, it goes up to almost 98%. Furthermore, the reduction is regular. For all the

instance sets, there is always more than one strategy that improve significantly the results. In the other cases, a better parametrization of the methods would certainly give better results. For example, in the instance set 9, the method based on the computation of dual bounds reduced drastically the number of column generation iterations, but the total computing time increased. This is due to the fact that we derived these bounds for each item size. Only for very demanding instances should such an approach be used.

From Tables 2 to 6, it seems that the trust region method provides the best results among all the approaches discussed in this paper. In fact, this happens because the DFF f_2^k of Carlier et al. (2007) gives near optimal values for these instances. These results just confirm the observation made in Ben Amor et al. (2006b) and Ben Amor et al. (2006a). In Table 7, we report on the results obtained with trust region when only u^k of Fekete and Schepers (2001) is used. This DFF gives results that are worse than f_2^k for these instances. In most of the cases, the method converges slower than the standard column generation algorithm. Hence, we can conclude that the trust region method depends heavily on the quality of the solutions given by the DFF.

The performance of the dual cuts (13), Algorithm 1 and dual bounds tends to be quite similar. These approaches give the best and most regular results. In fact, among these three approaches, no one clearly dominates the others. They always outperform the standard column generation algorithm, which is not the case for Algorithm 2. The quality of Algorithm 2 tends to be worse as the item sizes become smaller. However, all the approaches are consistently better than the standard dual cuts proposed in Carvalho (2005) which proved to be ineffective on large and hard cutting stock instances. Our approaches appear therefore as an alternative to these cuts.

6. Conclusion

It is well known that the linear bounds provided by column generation models for the CSP are strong, and hence any method that improves the efficiency of these methods is of great interest. Among the strategies proposed in the literature, dual cuts are among the most promising. In this paper, we proposed new procedures to derive dual cuts for the CSP. With the cuts generated using our approaches, we were able to reduce the number of column generation iterations necessary to solve this problem. We also described new methods for bounding the dual variables, permanently, or during an initialization phase. These methods

were tested on a broad range of instances from the literature, and they proved to be effective in many cases.

Many families of dual-feasible functions have been proposed in the literature, each is related to a family of dual solutions. Finding properties of functions that are linear combination of these known dual functions would be a way of improving our results. Applications of these ideas to enumerative methods is a work in progress.

Set	Strategy	*solved*	*itr*	%*red. itr*	t_{tot}	%*red.* t_{tot}
1	standard	10	1112.50		74.71	
	with dual cuts (Carvalho, 2005)	10	969.00	12.90	90.49	-21.12
	with dual cuts (13)	10	1059.60	4.76	68.21	8.69
	with dual cuts (13) and (Carvalho, 2005)	10	946.10	14.96	78.79	-5.47
	Algorithm 1	10	1064.80	4.29	73.18	2.04
	Algorithm 1 and dual cuts (Carvalho, 2005)	10	968.00	12.99	73.31	1.87
	Algorithm 2	10	858.30	22.85	71.02	4.93
	Algorithm 2 and dual cuts (Carvalho, 2005)	10	768.40	30.93	71.66	4.08
	with dual bounds	10	1059.90	4.73	69.37	7.14
	with dual bounds and cuts (Carvalho, 2005)	10	946.90	14.89	67.73	9.34
	trust region	10	651.40	41.45	59.95	19.75
	trust region with dual cuts (Carvalho, 2005)	10	610.90	45.09	96.13	-28.68
2	standard	8	2431.50		654.73	
	with dual cuts (Carvalho, 2005)	0	–	–	–	–
	with dual cuts (13)	10	2410.00	0.88	522.19	20.24
	with dual cuts (13) and (Carvalho, 2005)	8	1925.30	20.82	647.05	1.17
	Algorithm 1	10	2367.00	2.65	518.57	20.80
	Algorithm 1 and dual cuts (Carvalho, 2005)	10	1965.20	19.18	539.92	17.54
	Algorithm 2	10	1759.30	27.65	364.79	44.28
	Algorithm 2 and dual cuts (Carvalho, 2005)	10	1464.70	39.76	419.07	35.99
	with dual bounds	9	2336.10	3.92	554.66	15.28
	with dual bounds and cuts (Carvalho, 2005)	8	1913.40	21.31	503.08	23.16
	trust region	10	1494.70	38.53	291.10	55.54
	trust region with dual cuts (Carvalho, 2005)	10	1260.80	48.15	416.35	36.41

Table 2: Comparing the stabilization strategies on instance sets 1 and 2

7. Acknowledgements

This work was supported by the Portuguese Science and Technology Foundation through the postdoctoral grant SFRH/BPD/24139/2005 for François Clautiaux and the research project POS_C/57203/EIA/2004 for Cláudio Alves and José Valério de Carvalho.

Set	Strategy	*solved*	*itr*	%*red. itr*	t_{tot}	%*red.* t_{tot}
3	standard	9	861.20		218.40	
	with dual cuts (Carvalho, 2005)	8	701.80	18.51	354.31	-62.23
	with dual cuts (13)	10	791.70	8.07	119.37	45.35
	with dual cuts (13) and (Carvalho, 2005)	10	703.90	18.27	129.87	40.54
	Algorithm 1	10	797.00	7.45	106.41	51.28
	Algorithm 1 and dual cuts (Carvalho, 2005)	10	710.60	17.49	137.00	37.27
	Algorithm 2	10	828.20	3.83	193.46	11.42
	Algorithm 2 and dual cuts (Carvalho, 2005)	9	678.60	21.20	243.57	-11.52
	with dual bounds	10	785.90	8.74	108.35	50.39
	with dual bounds and cuts (Carvalho, 2005)	10	708.30	17.75	169.02	22.61
	trust region	10	567.60	34.09	82.74	62.12
	trust region with dual cuts (Carvalho, 2005)	10	499.90	41.95	80.44	63.17
4	standard	5	1545.60		766.15	
	with dual cuts (Carvalho, 2005)	0	–	–	–	–
	with dual cuts (13)	8	1762.00	-14.00	652.35	14.85
	with dual cuts (13) and (Carvalho, 2005)	8	1499.20	3.00	712.66	6.98
	Algorithm 1	9	1809.50	-17.07	690.94	9.82
	Algorithm 1 and dual cuts (Carvalho, 2005)	8	1476.80	4.45	603.15	21.28
	Algorithm 2	8	1857.70	-20.19	705.34	7.94
	Algorithm 2 and dual cuts (Carvalho, 2005)	6	1247.60	19.28	772.65	-0.85
	with dual bounds	8	1784.20	-15.44	654.16	14.62
	with dual bounds and cuts (Carvalho, 2005)	6	1227.60	20.57	713.16	6.92
	trust region	10	1297.30	16.06	536.25	30.01
	trust region with dual cuts (Carvalho, 2005)	9	1107.40	28.35	531.12	30.68

Table 3: Comparing the stabilization strategies on instance sets 3 and 4

The authors would like to thank the anonymous referees for their constructive comments, which helped improving the quality of the paper.

References

Alves, C., J.V. de Carvalho. 2007. Accelerating column generation for variable sized bin-packing problems. *European Journal of Operational Research* **183** 1333–1352.

Alves, C., J.V. de Carvalho. 2008. A stabilized branch-and-price-and-cut algorithm for the multiple length cutting stock problem. *Computers and Operations Research* **35** 1315–1328.

Ben Amor, H., J. Desrosiers, J.V. de Carvalho. 2006a. Dual-optimal inequalities for stabilized column generation. *Operations Research* **54** 454–463.

Set	Strategy	*solved*	*itr*	%*red. itr*	t_{tot}	%*red.* t_{tot}
5	standard	10	658.70		318.76	
	with dual cuts (Carvalho, 2005)	4	300.50	54.38	803.11	-151.95
	with dual cuts (13)	10	543.80	17.44	168.24	47.22
	with dual cuts (13) and (Carvalho, 2005)	10	498.70	24.29	154.96	51.39
	Algorithm 1	10	541.90	17.73	154.22	51.62
	Algorithm 1 and dual cuts (Carvalho, 2005)	10	501.80	23.82	185.80	41.71
	Algorithm 2	9	638.90	3.01	259.33	18.64
	Algorithm 2 and dual cuts (Carvalho, 2005)	4	325.70	50.55	752.45	-136.05
	with dual bounds	10	552.60	16.11	156.76	50.82
	with dual bounds and cuts (Carvalho, 2005)	10	498.80	24.28	185.14	41.92
	trust region	9	463.20	29.68	216.51	32.08
	trust region with dual cuts (Carvalho, 2005)	10	423.70	35.68	110.08	65.47
6	standard	1	591.20		893.11	
	with dual cuts (Carvalho, 2005)	0	–	–	–	–
	with dual cuts (13)	4	1049.70	-77.55	835.22	6.48
	with dual cuts (13) and (Carvalho, 2005)	6	978.20	-65.46	769.49	13.84
	Algorithm 1	7	1215.50	-105.60	779.52	12.72
	Algorithm 1 and dual cuts (Carvalho, 2005)	7	983.00	-66.27	776.13	13.10
	Algorithm 2	0	–	–	–	–
	Algorithm 2 and dual cuts (Carvalho, 2005)	0	–	–	–	–
	with dual bounds	4	1092.70	-84.83	843.75	5.53
	with dual bounds and cuts (Carvalho, 2005)	6	1061.20	-79.50	780.78	12.58
	trust region	8	1076.40	-82.07	631.65	29.27
	trust region with dual cuts (Carvalho, 2005)	4	768.10	-29.92	759.27	14.99

Table 4: Comparing the stabilization strategies on instance sets 5 and 6

Ben Amor, H., J. Desrosiers, F. Soumis. 2006b. Recovering an optimal lp basis from an optimal dual solution. *Operations Research Letters* **34** 569–576.

Caprara, A., M. Locatelli, M. Monaci. 2005. Bilinear packing by bilinear programming. Michael Jünger, Volker Kaibel, eds., *Integer Programming and Combinatorial Optimization, 11th International IPCO Conference, Berlin, Germany, June 8-10, 2005*, *Lecture Notes in Computer Science*, vol. 3509. Springer, 377–391.

Carlier, J., F. Clautiaux, A. Moukrim. 2007. New reduction procedures and lower bounds for the two-dimensional bin-packing problem with fixed orientation. *Computers and Operations Research* **34** 2223–2250.

Carlier, J., E. Néron. 2000. A new LP-based lower bound for the cumulative scheduling problem. *European Journal of Operational Research* **127** 363–382.

Set	Strategy	*solved*	*itr*	%*red. itr*	t_{tot}	%*red.* t_{tot}
7	standard	3	786.10		798.85	
	with dual cuts (Carvalho, 2005)	0	–	–	–	–
	with dual cuts (13)	9	1107.20	-40.85	427.94	46.43
	with dual cuts (13) and (Carvalho, 2005)	5	1052.50	-33.89	620.80	22.29
	Algorithm 1	9	1127.10	-43.38	362.28	54.65
	Algorithm 1 and dual cuts (Carvalho, 2005)	9	1044.80	-32.91	352.89	55.83
	Algorithm 2	3	798.40	-1.56	787.24	1.45
	Algorithm 2 and dual cuts (Carvalho, 2005)	0	–	–	–	–
	with dual bounds	10	1153.60	-46.75	332.40	58.39
	with dual bounds and cuts (Carvalho, 2005)	4	1023.80	-30.24	652.26	18.35
	trust region	8	1006.90	-28.09	355.22	55.53
	trust region with dual cuts (Carvalho, 2005)	0	–	–	–	–
8	standard	2	239.60		731.76	
	with dual cuts (Carvalho, 2005)	3	172.40	28.05	733.71	-0.27
	with dual cuts (13)	10	487.20	-103.34	30.16	95.88
	with dual cuts (13) and (Carvalho, 2005)	9	433.20	-80.80	137.16	81.26
	Algorithm 1	10	484.50	-102.21	27.74	96.21
	Algorithm 1 and dual cuts (Carvalho, 2005)	9	431.40	-80.05	113.71	84.46
	Algorithm 2	3	300.50	-25.42	657.27	10.18
	Algorithm 2 and dual cuts (Carvalho, 2005)	2	170.50	28.84	770.90	-5.35
	with dual bounds	10	484.50	-102.21	24.81	96.61
	with dual bounds and cuts (Carvalho, 2005)	9	438.10	-82.85	120.59	83.52
	trust region	9	452.90	-89.02	107.07	85.37
	trust region with dual cuts (Carvalho, 2005)	4	337.00	-40.65	557.15	23.86

Table 5: Comparing the stabilization strategies on instance sets 7 and 8

Carlier, J., E. Néron. 2007. Computing redundant resources for cumulative scheduling problems. *European Journal of Operational Research* **176** 1452–1463.

Carvalho, J. V. de. 2005. Using extra dual cuts to accelerate column generation. *INFORMS Journal on Computing* **17** 175–182.

Clautiaux, F., C. Alves, J.V. de Carvalho. 2008. A survey of dual-feasible and superadditive functions. *Annals of Operations Research (in press - doi:10.1007/s10479-008-0453-8)* .

Dantzig, G. B., P. Wolfe. 1960. Decomposition principle for linear programs. *Operations Research* **8** 101–111.

Degraeve, Z., M. Peeters. 2003. Optimal integer solutions to industrial cutting-stock problems: Part 2, benchmark results. *INFORMS Journal on Computing* **15** 58–81.

Set	Strategy	*solved*	*itr*	%*red. itr*	t_{tot}	%*red.* t_{tot}
9	standard	10	989.50		22.59	
	with dual cuts (Carvalho, 2005)	10	921.10	6.91	74.33	-229.11
	with dual cuts (13)	10	2.00	99.80	0.36	98.41
	with dual cuts (13) and (Carvalho, 2005)	10	2.00	99.80	0.31	98.64
	Algorithm 1	10	2.00	99.80	0.47	97.91
	Algorithm 1 and dual cuts (Carvalho, 2005)	10	2.00	99.80	0.39	98.26
	Algorithm 2	10	979.80	0.98	25.07	-11.01
	Algorithm 2 and dual cuts (Carvalho, 2005)	10	923.90	6.63	64.58	-185.94
	with dual bounds	10	1.00	99.90	31.16	-37.95
	with dual bounds and cuts (Carvalho, 2005)	10	1.00	99.90	32.76	-45.02
	trust region	10	9.00	99.09	0.70	96.91
	trust region with dual cuts (Carvalho, 2005)	10	9.00	99.09	5.04	77.69
HARD	standard	10	751.10		32.28	
	with dual cuts (Carvalho, 2005)	10	438.10	41.67	30.55	5.37
	with dual cuts (13)	10	699.60	6.86	25.91	19.74
	with dual cuts (13) and (Carvalho, 2005)	10	433.70	42.26	25.15	22.09
	Algorithm 1	10	705.40	6.08	25.38	21.38
	Algorithm 1 and dual cuts (Carvalho, 2005)	10	433.40	42.30	24.55	23.93
	Algorithm 2	10	760.60	-1.26	25.51	20.98
	Algorithm 2 and dual cuts (Carvalho, 2005)	10	476.90	36.51	25.33	21.52
	with dual bounds	10	664.70	11.50	25.21	21.91
	with dual bounds and cuts (Carvalho, 2005)	10	435.90	41.97	23.65	26.74
	trust region	10	369.10	50.86	10.99	65.96
	trust region with dual cuts (Carvalho, 2005)	10	309.20	58.83	11.12	65.56

Table 6: Comparing the stabilization strategies on instance sets 9 and HARD

	standard			trust region		
Set	*solved*	*itr*	t_{tot}	*solved*	*itr*	t_{tot}
1	10	1112.50	74.71	7	596.70	573.92
2	8	2431.50	654.73	0	211.70	900.00
3	9	861.20	218.40	0	138.10	900.00
4	4	1545.60	766.15	0	353.40	900.01
5	10	658.70	318.76	3	247.27	686.28
6	1	591.20	893.11	0	342.50	900.00
7	3	786.10	798.85	0	182.90	900.00
8	2	239.60	731.76	0	93.10	900.00
9	10	989.50	22.59	10	1093.20	87.26
HARD	10	751.10	32.28	10	535.60	18.84

Table 7: Comparing standard column generation and trust region with the DFF u^k of Fekete and Schepers (2001)

Du Merle, O., D. Villeneuve, J. Desrosiers, P. Hansen. 1999. Stabilized column generation. *Discrete Mathematics* **194** 229–237.

Dyckhoff, H. 1981. A new linear programming approach to the cutting stock problem. *Operations Research* **29** 145–159.

Fekete, S., J. Schepers. 2001. New classes of fast lower bounds for bin packing problems. *Mathematical Programming* **91** 11–31.

Gilmore, P., R. Gomory. 1961. A linear programming approach to the cutting stock problem. *Operations Research* **9** 849–859.

Gilmore, P., R. Gomory. 1963. A linear programming approach to the cutting stock problem - part II. *Operations Research* **11** 863–888.

Goffin, J., J. Vial. 1989. Cutting planes and column generation techniques with the projective algorithm. *Journal of Optimization Theory and Applications* **65** 409–429.

Hiriart-Urruty, J., C. Lemaréchal. 1993. *Convex Analysis and Minimization Algorithms II: Advanced Theory and Bundle Methods*. A Series of Comprehensive Studies in Mathematics, Springer-Verlag.

Ilog. 2006. *Ilog CPLEX 10.0 Reference Manual*. 9, rue de Verdun, BP 85, F-92453, Gentilly, France.

Johnson, D. S. 1973. Near optimal bin packing algorithms. Dissertation, Massachussetts Institute of Technology, Cambridge, Massachussetts.

Kallehauge, B., J. Larsen, O. Madsen. 2006. Lagrangean duality applied on vehicle routing with time windows. *Computers and Operations Research* **33** 1464–1487.

Kim, S., K. Chang, J. Lee. 1995. A descent method with linear programming subproblems for nondifferentiable convex optimization. *Mathematical Programming* **71** 17–28.

Luebbecke, M., J. Desrosiers. 2005. Selected topics in column generation. *Operations Research* **53** 1007–1023.

Marsten, R., W. Hogan, J. Blankenship. 1975. The BOXSTEP method for large-scale optimization. *Operations Research* **23** 389–405.

Martello, S., P. Toth. 1990. *Knapsack problems - Algorithms and Computer Implementation*. Wiley, Chichester.

Neame, P. 1999. Nonsmooth dual methods in integer programming. Ph.D. thesis, University of Melbourne, Australia.

Nemhauser, G. L., L.A. Wolsey. 1998. *Integer and Combinatorial Optimization*. Wiley, New York.

Perrot, N. 2005. Integer programming column generation strategies for the cutting stock problem and its variants. Ph.D. thesis, Université Bordeaux 1.

Puchinger, J., Raidl G. 2005. Models and algorithms for three-stage two-dimensional bin packing. *European Journal of Operational Research* **183** 1304–1327.

Scholl, A., R. Klein, C. Jurgens. 1997. BISON: a fast hybrid procedure for exactly solving the one-dimensional bin-packing problem. *Computers and Operations Research* **24** 627–645.

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