A Math-heuristic Dantzig-Wolfe Algorithm for the Capacitated Lot Sizing Problem

Marco Caserta[1] and Stefan Voß[2]

[1] IE Business School, Maria de Molina 12, 28006 - Madrid - Spain
`mcaserta@faculty.ie.edu`
[2] Institute of Information Systems (IWI), University of Hamburg
Von-Melle-Park 5, 20146 Hamburg, Germany
`stefan.voss@uni-hamburg.de`

Abstract. The multi-item multi-period capacitated lot sizing problem with setups (CLST) is a well known optimization problem with wide applicability in real-world production planning problems. Based on a recently proposed Dantzig-Wolfe approach we present a novel math-heuristic algorithm for the CLST. The major contribution of this paper lies in the presentation of an algorithm that exploits exact techniques (Dantzig-Wolfe) in a metaheuristic fashion, in line with the novel trend of math-heuristic algorithms. To the best of the authors knowledge, it is the first time that such technique is employed within a metaheuristic framework, with the aim of tackling challenging instances in short computational time.

1 Introduction

The Multi-Item Multi-Period Capacitated Lot Sizing Problem with Setups (CLST) is a well known optimization problem that finds a wide variety of real-world applications. The CLST belongs to the class of $\mathcal{NP}$-hard problems [1, 15, 12]. A mixed-integer formulation of the CLST is:

$$
\begin{aligned}
\text{(CLST): min} \quad & z = \sum_{j=1}^{n}\sum_{t=1}^{T}\left(f_{jt}y_{jt} + c_{jt}x_{jt} + h_{jt}s_{jt}\right) + \sum_{j=1}^{n} h_{j0}s_{j0} \\
\text{s.t.} \quad & \sum_{j=1}^{n}\left(a_{jt}x_{jt} + m_{jt}y_{jt}\right) \le b_t \quad \forall t \\
& s_{jt-1} + x_{jt} = d_{jt} + s_{jt} \quad \forall j,t \\
& x_{jt} \le M y_{jt} \quad \forall j,t \\
& y_{jt} \in \{0,1\} \quad \forall j,t \\
& x_{jt}, s_{jt} \ge 0 \quad \forall j,t
\end{aligned}
$$

where items $j = 1, \ldots, n$ should be produced over time periods $t = 1, \ldots, T$. In the CLST formulation, f_{jt}, c_{jt}, and h_{jt} indicate the fixed cost, the unitary production cost and the unitary inventory holding cost for item j in period t, respectively. Parameters m_{jt} and a_{jt} indicate the setup time and the unitary production time, respectively, while b_t stands for the production capacity in period t. Parameter d_{jt} indicates the demand of item j in period t. Finally, in the model, three sets of decision variables are employed, *i.e.*, $y_{jt} \in \{0,1\}$, which takes value 1 if there is a setup for item j in period t; as well as $x_{jt} \geq 0$ and $s_{jt} \geq 0$, indicating the production volume and the inventory level for item j in period t, respectively. Note that s_{j0} is given as data indicating initial inventory.

Due to its vast industrial applicability, researchers have devoted special attention to the CLST (see, *e.g.*, [5], [13], [6], and [4]). Since the CLST is still difficult to solve to optimality, many researchers have tried to tackle the problem by working on relaxations of the same. A good description of some well studied relaxations of the CLST is provided by [11]. A recent discussion of solution approaches for the CLST can be found in [8].

In recent years, a lot of attention has been devoted to the integration, or hybridization, of metaheuristics with exact methods. This exposition also relates to the term math-heuristics (see, *e.g.*, [9]) which describes works which are, e.g., exploiting mathematical programming techniques in (meta)heuristic frameworks or on granting to mathematical programming approaches the cross-problem robustness and constrained-CPU-time effectiveness which characterize metaheuristics. Discriminating landmark is some form of exploitation of the mathematical formulation of the problems of interest. Here we follow the math-heuristic concept rather than a more general idea of hybridization, where the successful ingredients of various metaheuristics have been combined.

Based on a recently proposed Dantzig-Wolfe approach of [7], in this paper we present a novel math-heuristic algorithm for the CLST. The major contribution of this paper lies in the presentation of an algorithm that exploits exact techniques (Dantzig-Wolfe) in a metaheuristic fashion, in line with the novel trend of math-heuristic algorithms. To the best of the authors' knowledge, it is the first time that such technique is employed within a metaheuristic framework, with the aim of tackling challenging instances in short computational time. In addition, the proposed approach constitutes a clear example of the effectiveness of hybrid approaches, *e.g.*, approaches that intertwine classical mathematical programming techniques with novel metaheuristics.

To measure the effectiveness of the proposed approach, we have tested the algorithm to solve standard benchmark instances from [15], in line with what has been done by a number of authors, *e.g.*, [7], and [3].

The organization of the paper is as follows. In the next section, Section 2, we present a mathematical formulation and a Dantzig-Wolfe decomposition of the CLST; Section 3 illustrates the main ingredients of the proposed math-heuristic algorithm, while Section 4 summarizes the results of the algorithm when tested on a set of benchmark instances. Finally, Section 5 concludes with some final remarks.

2 Mathematical Formulation and Decomposition

Dantzig-Wolfe (DW) decomposition is a well known technique often employed to address mixed integer programs with substructures. This technique has been successfully applied in a number of contexts. (For more details on such technique see, *e.g.*, [16], and [2].) Recently, [7] have presented a DW approach for the CLST, addressing an important structural deficiency of the standard DW approach for the CLST proposed decades ago by [10].

In this section, borrowing ideas from [7], we present a DW reformulation and decomposition for the CLST that is especially suited for the math-heuristic algorithm presented in Section 3. In line with what proposed by [7], we employ a formulation in which setup variables and production variables are dealt with separately. Next, we illustrate how such reformulation leads to a natural implementation of a math-heuristic algorithm aimed at speeding up the column generation phase.

Let us consider the DW decomposition and its associated reformulation. We identify the capacity constraints as the "hard" constraints. Thus, if we eliminate such hard constraints, we obtain a problem that can easily be decomposed into smaller (easy) subproblems, one per item. Let us indicate with:

$$X_j^{\text{sub}} = \left\{ \begin{array}{lcl} s_{jt-1} + x_{jt} & = & d_{jt} + s_{jt} \\ \quad x_{jt} & \leq & My_{jt} \\ y_{jt} \in \{0,1\} & & \\ x_{jt}, s_{jt} \geq 0 & & \end{array} \right\}$$

the set of feasible solutions for the j^{th} subproblem. It is easy to observe that X_j^{sub} defines the feasible region for the single item uncapacitated lot sizing problem.

Let us now indicate with $\left\{\mathbf{x}_j^k\right\}$, where $\mathbf{x}_j^k = \left(y_{jt}^k, x_{jt}^k, s_{jt}^k\right)$, with $k = 1, \ldots, K_j$, the set of extreme points of $conv(X_j^{\text{sub}})$. That is, for each item j, the set of extreme points of the corresponding polytope is defined by all the possible feasible schedules and the corresponding dominant production schedules, *i.e.*, production schedules satisfying the Wagner-Whitin condition $s_{jt-1}x_{jt} = 0$ (also known as zero-inventory property). Thus, variables $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{s}$ of the CLST can be rewritten as convex combination of such extreme points. Therefore, we rewrite the CLST as:

$$\text{(M): } \min z = \sum_{j=1}^{n} \sum_{k=1}^{K_j} \left[\sum_{t=1}^{T} \left(f_{jt} y_{jt}^k + c_{jt} x_{jt}^k + h_{jt} s_{jt}^k \right) + \sum_{j=1}^{n} h_{j0} s_{j0} \right] \lambda_{jk} \quad (1)$$

$$\text{s.t. } \sum_{j=1}^{n} \sum_{k=1}^{K_j} \left(a_{jt} x_{jt}^k + m_{jt} y_{jt}^k \right) \lambda_{jk} \leq b_t, \quad t = 1, \ldots, T \quad (2)$$

$$\sum_{k=1}^{K_j} \lambda_{jk} = 1, \quad j = 1, \ldots, n \quad (3)$$

$$\lambda_{jt} \in \{0,1\}, \quad j = 1, \ldots, K, \ k = 1, \ldots, K_j \quad (4)$$

The original CLST is rewritten in such a way that every extreme point of the polyhedron of subproblems X_j^{sub} is enumerated and the corresponding weight variable is either set to 1, if the extreme point is selected, or to 0 otherwise. However, it is a well-known fact that, if the capacity constraint is binding for at least one time period, the extreme points of the polytope of the single item uncapacitated lot sizing problem will not necessarily provide an optimal solution to the overall problem [7]. Thus, when imposing the binary constraints on the reformulation variables, *i.e.*, constraint (4), the optimal solution to the original CLST could be missed. In other words, there is not a strict correspondence between the original setup variables y_{jt} and the newly introduced reformulation variables λ_{jk}. As pointed out by [7], the key point here is that the dominant plans are only a subset of the set of extreme points needed to completely define the search space of the original CLST.

What we would need is a convex combination of the setup plans for each single item problem, *i.e.*, we should eliminate constraint (4) from the master problem (M) and, instead of it, we should impose the binary conditions on the original setup variables. This approach is called *convexification approach* to the DW decomposition [16]. Alternatively, one could use a *discretization approach*, in which *all* the integer solutions are enumerated (including interior points of the subsystem polyhedron, *i.e.*, including solutions that make use of non-dominant production plans and yet satisfy the Wagner-Whitin property). The discretization approach would allow to re-introduce constraint (4) into the master problem.

As pointed out by [7], it is also possible to use a third approach, *i.e.*, a discretization approach for the binary variables of the CLST and a convexification approach for the continuous variables. That is, given a feasible setup plan, a convex combination of production plans respecting that setup plan is generated. Therefore, the set of extreme points included in the master program includes non-dominant plans satisfying the Wagner-Whitin property, *i.e.*, there will be solutions for which, in some periods, it is possible that there exists a setup but no production.

Consequently, in the sequel, we will introduce two sets of reformulation variables, *i.e.*, (i) λ_{jk}, to select exactly one setup plan among the ones included in the master problem, and (ii) μ_{jkw}, to select a convex combination of production schedules (dominant and non-dominant) arising from a given setup schedule.

An Example. Let us consider a single-item lot sizing problem with five time periods. Let us fix $d_t = 10$, for $t = 1, \dots, 5$. Let us assume we are given a setup schedule, in which there exists a setup in the first, third, and fifth period (as presented in Table 1). In the sequel, for the sake of clarity, inventory levels s_t are omitted. In the table, the row associated to solution x_1 corresponds to the dominant production plan associated to the given setup plan. However, in the reformulation, we also want to consider non-dominant production plans that are still feasible with respect to the given setup plan, *e.g.*, production plans x_2 to x_4. Every production plan presented in Table 1 still satisfies the Wagner-Whitin property. However, except for the production schedule indicated by x_1, all the production schedules are non-dominant plans, *i.e.*, there exist some periods in which a setup is not accompanied by production.

t	1	2	3	4	5
d_t	10	10	10	10	10
y_t^k	1	0	1	0	1
x_1^k	20	0	20	0	10
x_2^k	50	0	0	0	0
x_3^k	20	0	30	0	0
x_4^k	40	0	0	0	10

Table 1. An example of the convexification-discretization approach.

In line with what is mentioned above, the *convexification-discretization* approach defines two sets of dual variables, one for the setup plan and one for the production plan. Therefore, with respect to Table 1, we will use:

– A binary variable $\lambda_k \in \{0, 1\}$ to determine whether the setup plan described by $\mathbf{y}^k$ should be selected; and
– A set of continuous variables μ_w^k, with $w = 1, \ldots, 4$, to define a convex combination of production plans arising from the same setup plan.

To formalize, and in line with what is presented in [7], let us assume we are considering a single-item lot sizing problem, *e.g.*, item $j \in [1, n]$. Let us also assume we indicate with

$$\mathcal{Y} = \left\{\mathbf{y}^1, \ldots, \mathbf{y}^k, \ldots, \mathbf{y}^{K_j}\right\}$$

the set of all feasible setup plans for item j. Let us now consider a setup schedule $\mathbf{y}^k = \left(y_1^k, \ldots, y_T^k\right) \in \mathcal{Y}$ and let us define the set of induced setup schedules as:

$$\mathcal{Y}(k) = \left\{(y_1, \ldots, y_T) : y_t \leq y_t^k, y_t \in \{0, 1\}, t = 1, \ldots, T\right\}$$

That is, given a setup schedule $\mathbf{y}^k$, the set of induced setup schedules, indicated with $\mathcal{Y}(k)$, is built by taking all the possible schedules dominated by $\mathbf{y}^k$, *i.e.*, assuming that $\mathbf{y}^k$ defines s setups, the set $\mathcal{Y}(k)$ contains 2^s setup schedules. Obviously, for each setup schedule in $\mathcal{Y}(k)$ there exists a unique dominant Wagner-Whitin production plan. Thus, the set of extreme points defined by the setup schedule $\mathbf{y}^k$ is defined by all the combinations of the Wagner-Whitin production plans arising from the induced set $\mathcal{Y}(k)$ with the original setup plan $\mathbf{y}^k$.

With respect to the example provided in Table 1, we have that $\mathbf{y}^k = (1, 0, 1, 0, 1)$. Thus, the set of induced setup schedules is:

$$\mathcal{Y}(k) = \{(1, 0, 1, 0, 1), (1, 0, 0, 0, 0), (1, 0, 1, 0, 0), (1, 0, 0, 0, 1)\}.$$

As previously mentioned, since the setup schedule $\mathbf{y}^k$ defines three setups, and considering the setup in the first period as fixed, the setup schedule $\mathbf{y}^k$ gives rise to $2^{3-1} = 4$ induced setup plans. Each setup plan in $\mathcal{Y}(k)$, in turn, defines a unique Wagner-Whitin schedule, as indicated in Table 1. Thus, we now have four extreme points that could be added to the master plan, *i.e.*, $\left(\mathbf{y}^k, \mathbf{x}_1^k\right)$, $\left(\mathbf{y}^k, \mathbf{x}_2^k\right)$, $\left(\mathbf{y}^k, \mathbf{x}_3^k\right)$, and $\left(\mathbf{y}^k, \mathbf{x}_4^k\right)$.

Finally, with respect to the dual variables of the master problem, *i.e.*, the new columns of program (M), we generate the following 2^{s-1} new columns, *i.e.*, $\left(\lambda_k, \mu_1^k\right), \left(\lambda_k, \mu_2^k\right), \left(\lambda_k, \mu_3^k\right)$, and $\left(\lambda_k, \mu_4^k\right)$.

Finally, once we generated a set of (dominant and non-dominant) production plans associated to a given setup plan, the relation between these two is established using the following constraints (considering a single-item problem):

$$\sum_{k=1}^{K_j} \lambda_k = 1 \tag{5}$$

$$\sum_{w=1}^{|\mathcal{Y}(k)|} \mu_w^k = \lambda_k, \quad k = 1, \ldots, K_j \tag{6}$$

$$\lambda_k \in \{0, 1\}, \quad k = 1, \ldots, K_j \tag{7}$$

$$\mu_w^k \geq 0, \quad w = 1, \ldots, |\mathcal{Y}(k)|, \; k = 1, \ldots, K_j \tag{8}$$

Thus, via Equations (5) and (7) we enforce the discretization mechanism for the (binary) setup variables, *i.e.*, one setup plan must be selected; on the other hand, with Equations (6) and (8), we enforce the convexification mechanism, *i.e.*, once a setup plan is selected (*e.g.*, $\lambda_k = 1$ for any $k \in [1, K_j]$), we allow for a convex combination of (dominant and non-dominant) induced production plans to be selected.

Let us now present the Dantzig-Wolfe master reformulation and the derived subproblems.

$$\min z = \sum_{j=1}^{n} \sum_{k=1}^{K_j} \left[\sum_{t=1}^{T} \left(f_{jt} y_{jt}^k \lambda_{jk} + \sum_{w=1}^{|\mathcal{Y}(k)|} \left(c_{jt} x_{jt}^w + h_{jt} s_{jt}^w \right) \mu_{jk}^w \right) \right] \tag{9}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} \sum_{k=1}^{K_j} \left(m_{jt} y_{jt}^k \lambda_{jk} + \sum_{w=1}^{|\mathcal{Y}(k)|} a_{jt} x_{jt}^w \mu_{jk}^w \right) \leq b_t, \quad t = 1, \ldots, T \tag{10}$$

$$\sum_{k=1}^{K_j} \lambda_{jk} = 1, \quad j = 1, \ldots, n \tag{11}$$

$$\sum_{w=1}^{|\mathcal{Y}(k)|} \mu_{jk}^w = \lambda_{jk}, \quad j = 1, \ldots, n, \; k = 1, \ldots, K_j \tag{12}$$

$$\lambda_{jt} \in \{0, 1\}, \quad j = 1, \ldots, K, \; k = 1, \ldots, K_j \tag{13}$$

$$\mu_{jk}^w \geq 0, \quad j = 1, \ldots, K, \; k = 1, \ldots, K_j, \; w = 1, \ldots, |\mathcal{Y}(k)| \tag{14}$$

Due to the large number of variables of the master problem, the linear programming relaxation of the master is solved using column generation. The subproblem used to price in new columns is (separable over single items j):

$$\min\ r_j = \sum_{t=1}^{T} \left(\left(f_{jt} - u_t m_{jt} \right) y_{jt} + \left(c_{jt} - u_t a_{jt} \right) x_{jt} + h_{jt} s_{jt} \right) - \alpha_j \tag{15}$$

$$\text{s.t. } x_{jt} + s_{jt-1} = d_{jt} + s_{jt}, \quad t = 1, \ldots, T \tag{16}$$

$$x_{jt} \leq M y_{jt}, \quad t = 1, \ldots, T \tag{17}$$

$$y_{jt} \in \{0,1\}, \quad t = 1, \ldots, T \tag{18}$$

$$x_{jt}, s_{jt} \geq 0, \quad t = 1, \ldots, T \tag{19}$$

where $\mathbf{u}$ are the dual variables associated to the constraint (10) and α are the dual variables associated to constraint (11). As long as the pricing problem returns negative reduced costs for at least one item j, with $j = 1, \ldots, n$, these columns are added to the master, and the linear programming relaxation of the master is re-optimized. However, it is worth noting that, due to the discretization-convexification approach, the number of new columns generated every time the subproblem is solved can become large. Thus, in the next section, we propose a metaheuristic approach, based on the corridor method, aimed at finding a "good" set of new columns to be added to the master problem within a prespecified amount of computational time.

3 A math-heuristic approach

Let us now present in detail the math-heuristic column generation approach used to solve the subproblems.

The proposed approach belongs to the class of math-heuristic algorithms, since we use mathematical programming techniques in a heuristic fashion [9]. The basic steps of a column generation approach are here briefly highlighted:

1. Populate the master problem with an initial set of columns.
2. Solve the LP relaxation of the master.
3. Solve the subproblems defined using the current dual values obtained from the master.
4. Price in new columns: If there exists at least one new column with negative reduced cost, add such column(s) to the master. Otherwise, stop.

The main contribution of this paper is related to the use of a corridor method-inspired scheme [14] to address step 3. As presented in Section 2, we use a discretization-convexification approach for the master problem, *i.e.*, we employ discretization for the selection of the setup variables and convexification for the selection of a combination of induced production plans. Therefore, in phase 3 of the column generation approach, we need to devise an efficient method aimed at obtaining a "good" setup plan and a sufficiently large set of induced production plans. Given a current setup plan $\mathbf{y}^k$, there exist 2^{s-1} induced production plans, *i.e.*, $|\mathcal{Y}(k)| = 2^{s-1}$, where s is the total number of setups in $\mathbf{y}^k$, *i.e.* $\sum_{t=1}^{T} y_t^k = s$. Therefore, a complete enumeration of all the induced production plans for a given setup plan $\mathbf{y}^k$ is infeasible for practical, real-world size instances.

Due to the aforementioned drawback, we propose here a math-heuristic approach to generate a sufficiently large set $\mathcal{Y}(k)$ of induced production plans in a controlled amount of computational time. Let us present here the steps of the corridor method-inspired scheme used to generate new columns (the scheme is presented for a single item). In Figure 1, we present in details the steps of the third phase of the algorithm. Basically, we first generate a Wagner-Whitin solution for the current subproblem. Then, we apply a corridor method algorithm to collect a set of induced solutions. Such solutions are stored in a pool of solutions Ω. The corridor method phase stops when either the optimal solution to the constraint problem or a maximum running time have been reached. Subsequently, the solutions in the pool Ω are priced in and, whenever a solution with negative reduced cost is found, such solution is added to the master.

The overall algorithm begins by creating an initial set of columns for the master problem. We first add to the master a set of Wagner-Whitin solutions obtained fixing all the dual values equal to zero, along with the solutions in which all the demand is satisfied using initial inventory s_{j0}. Once these columns are added to the master, the linear programming relaxation of problem (9)-(14) is solved to optimality using a standard LP solver. The dual values obtained after solving problem (M) are then used to define the subproblems (15)-(19) (one per item). Each subproblem is then solved with the proposed algorithm of Figure 1 and new columns are priced into the master problem (9)-(14). The master-subproblem cycle is repeated until there exist no new columns with negative reduced cost. In that case, the algorithm stops and problem (9)-(14) is finally solved to optimality and the best feasible solution for the original CLST is returned.

4 Computational Results

In this section we present the computational results of the algorithm on a set of well-known benchmark instances. We report results on six instances taken from the test set used by [15], as reported by [3]. The same instances have been tackled by [7] and, therefore, constitute an interesting test bed for the preliminary evaluation of the proposed algorithm. As reported by [7], these instances are hard to solve to optimality. However, as presented in Table 2, we could solve all these instances to optimality in a reasonable amount of computational time using IBM CPLEX. Thus, we could also measure how far our heuristic solution was from an optimal solution. Optimal values for these instances were also reported by [11].

The algorithm proposed in this paper was coded in C++ and compiled using the GNU g++ 4.5.2 compiler on a dual core Pentium 1.8GHz Linux workstation with 4Gb of RAM. Throughout the computational experiment phase, the maximum running time for each "constrained" subproblem $r_j(\mathbf{u}, \alpha, \mathbf{y}^k)$ was kept fixed to one second, while the number of columns generated in each iteration δ was fixed to 100.

In Table 2, the first column reports the instance name, columns two and three report the optimal value and the running time obtained using IBM CPLEX 12.1 as MIP solver. Columns four and five provide the best value and the running time of [3], while columns six and seven provide the same information as presented in [7]. Finally, the last two columns provide the best result and the running time of the proposed algorithm. With respect to running times, [3] has a limit of 900 seconds, while [7] stopped the algorithm after vising 2000 nodes.

Instance	Optimal		BW		DJ		CM	
	z^*	$T^\natural$	z	$T^\dagger$	z	$T^\ddagger$	z	$T^\natural$
Tr6-15	37,721	1.24	37,721	38.4	38,162	29	37,721	1.62
Tr6-30	61,746	126.2	61,806	900	62,644	359	62,885	6.45
Tr12-15	74,634	2.67	74,799	900	75,035	66	74,727	29.65
Tr12-30	130,596	154.41	132,650	900	131,234	215	131,185	19.91
Tr24-15	136,509	23.70	136,872	900	136,860	44	136,556	9.34
Tr24-30	287,929	110.99	288,424	900	288,383	306	287,974	40.73

$\natural$: time on a dual core pentium 1.8GHz Linux workstation.
$\dagger$: time on a 200MHz Windows workstation.
$\ddagger$: time on a pentium III 750MHz Windows workstation.

Table 2. Results on six instances from [15]. The optimal values and corresponding running times reported here have been obtained using CPLEX 12.1.

From the table, we can observe that the proposed algorithm is competitive in terms of both solution quality and running time, especially for larger instances. Running times among the three algorithms cannot be meaningfully compared, due to the huge differences in machines speed. However, we can conclude that the proposed approach is faster than CPLEX alone in solving these instances, of course at a price of delivering a near-optimal solution.

While it is true that no robust conclusions can be drawn on a such a limited benchmark test, the results presented in this paper do show that the proposed approach is promising.

5 Conclusions

In this paper, we have presented a novel math-heuristic for a well known optimization problem, the lot sizing problem with setup times and setup costs. The main contribution of the work lies in the introduction of a math-heuristic approach for the column generation phase of a Dantzig-Wolfe algorithm.

Starting from an observation of [7], we presented a Dantzig-Wolfe reformulation in which the setup variables and the production variables are dealt with separately. More specifically, for any given setup plan, we generate columns in which the production plan needs not be a dominant plan, *i.e.*, we introduce into the master problem columns corresponding to solutions in which, for certain periods, there might be a setup without having a production. Thus, a single setup plan induces a number of non-dominant production plans. However, due to the large size of the set of non-dominant plans induced by each setup plan, we designed a mechanism inspired in the corridor method to bound the search of non-dominant production plans in the neighborhood of the Wagner-Whitin dominant plan associated to the current setup plan. By adding an exogenous constraint to the pricing problem, we collect a set of new columns and, subsequently, we price into the master problem those columns with negative reduced costs. Finally, the column generation approach is repeated until no new columns with negative reduced costs are found.

The proposed algorithm has been tested on a well-known set of benchmark instances and the results obtained have been compared with two approaches from the literature, as well as with the optimal solutions obtained by IBM CPLEX 12.1. While additional results on other problems are still needed, the results presented in the computational section allow to conclude that the proposed approach is promising, both in terms of solution quality and running time, and leaves various options for future applicability in other types of problems.

References

1. O. A. Aras. *A Heuristic Solution Procedure for a Single-Facility Multi-Item Dynamic Lot Sizing and Sequencing Problem.* PhD thesis, Syracuse University, 1981.
2. C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch and Price: Column Generation for Huge Integer Programs. *Operations Research*, 46(3):316–329, 1998.
3. G. Belvaux and L. A. Wolsey. BC-PROD: A Specialized Branch-and-Cut System for Lot-Sizing Problems. *Management Science*, 46(5):724–738, 2000.
4. M. Constantino. A Cutting Plane Approach to Capacitated Lot-Sizing with Start-up Costs. *Mathematical Programming B*, 75(3):353–376, 1996.
5. M.A.H. Dzielinski and R.E. Gomory. Optimal Programming of Lot-Sizing Inventory and Labor Allocation. *Management Science*, 11(9):874–890, 1965.
6. G.D. Eppen and R. K. Martin. Solving Multi-Item Lot-Sizing Problems Using Variable Redefinition. *Operations Research*, 35(6):832–848, 1987.
7. R. Jans and Z. Degraeve. A New Dantzig-Wolfe Reformulation and Branch-and-Price Algorithm for the Capacitated Lot Sizing Problem with Setup Times. *Operations Research*, 55(5):909–920, 2007.
8. R. Jans and Z. Degraeve. Meta-heuristics for Dynamic Lot Sizing: A Review and Comparison of Solution Approaches. *European Journal of Operational Research*, 177(3):1855–1875, 2007.
9. V. Maniezzo, T. Stützle, and S. Voß, editors. *Matheuristics – Hybridizing Metaheuristics and Mathematical Programming.* Annals of Information Systems, Vol. 10. Springer, 2010.
10. A. S. Manne. Programming of Economic Lot Sizes. *Management Science*, 4(2):115–135, 1958.

11. A. J. Miller, G. L. Nemhauser, and M. W. Savelsbergh. Solving Multi-Item Capacitated Lot-Sizing Problems with Setup Times by Branch and Cut. Core dp 2000/39, Université Catholique de Louvain, Louvain-la-Neuve, Belgium, August 2000.
12. G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley, 1999.
13. Y. Pochet and L. A. Wolsey. Polyhedra for Lot Sizing with Wagner-Whitin Costs. *Mathematical Programming B*, 67(1-3):297–323, 1994.
14. M. Sniedovich and S. Voß. The Corridor Method: A Dynamic Programming Inspired Metaheuristic. *Control and Cybernetics*, 35(3):551–578, 2006.
15. W.W. Trigeiro, L.J. Thomas, and J.O. McClain. Capacitated Lot-sizing With Setup Times. *Management Science*, 35(3):353–366, 1989.
16. F. Vanderbeck and M. W. P. Savelsbergh. A Generic View of Dantzig-Wolfe Decomposition in Mixed Integer Programming. *Operations Research Letters*, 34(3):296–306, 2006.

S1. Initialization

1. Solve problem $r_j(\mathbf{u},\alpha)$ using Wagner-Whitin $\Rightarrow \left(\mathbf{y}^k, \mathbf{x}_1^k\right)$
2. If $r_j(\mathbf{u},\alpha) \geq 0$, STOP.

S2. Corridor Method$\left(\mathbf{y}^k, \delta\right)$

1. Define a neighborhood around the Wagner-Whitin solution as:
$$\mathcal{N}(\mathbf{y}^k) = \left\{ y \in \{0,1\}^T : y_{jt} \leq y_{jt}^k \right\} \tag{20}$$
2. Add the following *corridor constraint* to the subproblem $r_j(\mathbf{u},\alpha)$:
$$y_{jt} \leq y_{jt}^k, \quad t = 1, \ldots, T \tag{21}$$
and solve the resulting "constrained" subproblem $r_j(\mathbf{u},\alpha,\mathbf{y}^k)$.
3. While solving to optimality $r_j(\mathbf{u},\alpha,\mathbf{y}^k)$, collect the best $\delta \geq 1$ feasible solutions and store them in a pool:
$$\Omega = \left\{ (\mathbf{y}^{kw}, \mathbf{x}^{kw}) : \mathbf{y}^{kw} \in \mathcal{N}(\mathbf{y}^k) \right\} \tag{22}$$
where $\mathbf{y}^{kw}$ is a setup plan satisfying constraint (21), and $\mathbf{x}^{kw}$ is the dominant Wagner-Whitin solution associated to the setup plan $\mathbf{y}^{kw}$.
4. Stop the corridor method when one of the two criteria has been reached:
   a. maximum running time, or
   b. optimal solution.

S3. Pricing$\left(\mathbf{y}^k, \Omega\right)$

1. Price in new columns until all the solutions in Ω have been examined and Ω is empty:
   (a) Select a solution from the pool Ω and compute the reduced cost of the composed solution $\left(\mathbf{y}^k, \mathbf{x}^{kw}\right)$, with $w = 1, \ldots, |\Omega|$
   (b) If the reduced cost of the current solution is negative, *i.e.*, $r_j < 0$, add the column $\left(\mathbf{y}^k, \mathbf{x}^{kw}\right)$ to the master. Otherwise, discard the column, eliminate it from Ω, and go back to step 1a.

Fig. 1. Outline of the proposed corridor method-inspired algorithm for the generation of new columns.