OPERATIONS RESEARCH
Vol. 55, No. 5, September–October 2007, pp. 909–920
ISSN 0030-364X | EISSN 1526-5463 | 07 | 5505 | 0909

informs

DOI 10.1287/opre.1070.0404

A New Dantzig-Wolfe Reformulation and Branch-and-Price Algorithm for the Capacitated Lot-Sizing Problem with Setup Times

Zeger Degraeve
London Business School, Regent's Park, London NW1 4SA, United Kingdom, zdegraeve@london.edu

Raf Jans
RSM Erasmus University, 3000 DR Rotterdam, The Netherlands, rjans@rsm.nl

Although the textbook Dantzig-Wolfe decomposition reformulation for the capacitated lot-sizing problem, as already proposed by Manne [Manne, A. S. 1958. Programming of economic lot sizes. *Management Sci.* **4**(2) 115–135], provides a strong lower bound, it also has an important structural deficiency. Imposing integrality constraints on the columns in the master program will not necessarily give the optimal integer programming solution. Manne's model contains only production plans that satisfy the Wagner-Whitin property, and it is well known that the optimal solution to a capacitated lot-sizing problem will not necessarily satisfy this property. The first contribution of this paper answers the following question, unsolved for almost 50 years: If Manne's formulation is not equivalent to the original problem, what is then a correct reformulation? We develop an equivalent mixed-integer programming (MIP) formulation to the original problem and show how this results from applying the Dantzig-Wolfe decomposition to the original MIP formulation. The set of extreme points of the lot-size polytope that are needed for this MIP Dantzig-Wolfe reformulation is much larger than the set of dominant plans used by Manne. We further show how the integrality restrictions on the original setup variables translate into integrality restrictions on the new master variables by separating the setup and production decisions. Our new formulation gives the same lower bound as Manne's reformulation. Second, we develop a branch-and-price algorithm for the problem. Computational experiments are presented on data sets available from the literature. Column generation is accelerated by a combination of simplex and subgradient optimization for finding the dual prices. The results show that branch-and-price is computationally tractable and competitive with other state-of-the-art approaches found in the literature.

Subject classifications: integer programming, algorithms: decomposition, column generation, branch-and-price; inventory/production: lot sizing: setup times.
Area of review: Optimization.
History: Received December 2004; revisions received January 2006, June 2006; accepted June 2006. Published online in *Articles in Advance* August 20, 2007.

1. Introduction and Problem Description

We consider an extension of the basic dynamic lot-sizing problem. Before an item can be produced, a setup must be performed and the setup time decreases the available capacity. This problem is known as the capacitated lot-sizing problem with setup times (CLST) (Trigeiro et al. 1989). Define the following sets, parameters and variables:

Sets
$I = \{1, \ldots, n\}$: set of items,
$T = \{1, \ldots, m\}$: set of time periods.

Parameters
d_{it}: demand of item i in period t $\forall i \in I$, $\forall t \in T$;
sd_{itk}: sum of demand of item i, from period t until k $\forall i \in I$, $\forall t, k \in T$: $k \geqslant t$;
hc_{it}: unit holding cost for item i in period t $\forall i \in I$, $\forall t \in T$;
sc_{it}: setup cost for item i in period t $\forall i \in I$, $\forall t \in T$;
vc_{it}: variable production cost for item i in period t $\forall i \in I$, $\forall t \in T$;
fc_i: unit cost for initial inventory for item i $\forall i \in I$;
st_{it}: setup time for item i in period t $\forall i \in I$, $\forall t \in T$;
vt_{it}: variable production time for item i in period t $\forall i \in I$, $\forall t \in T$;
cap_t: time capacity in period t $\forall t \in T$.

Decision variables
x_{it}: production quantity of item i in period t $\forall i \in I$, $\forall t \in T$;
y_{it} := 1 if setup for item i in period t, = 0 otherwise $\forall i \in I$, $\forall t \in T$;
s_{it}: inventory for item i at the end of period t $\forall i \in I$, $\forall t \in T$;
s_{i0}: amount of initial inventory for item i $\forall i \in I$.

The mathematical formulation of the CLST is then as follows:

$$\text{Min} \quad \sum_{i \in I} fc_i s_{i0} + \sum_{i \in I} \sum_{t \in T} (sc_{it} y_{it} + vc_{it} x_{it} + hc_{it} s_{it}) \tag{1}$$

$$\text{s.t.} \quad s_{i,t-1} + x_{it} = d_{it} + s_{it} \quad \forall i \in I, \forall t \in T, \tag{2}$$

$$x_{it} \leqslant sd_{itm} y_{it} \quad \forall i \in I, \forall t \in T, \tag{3}$$

$$\sum_{i \in I} (st_{it} y_{it} + vt_{it} x_{it}) \leqslant cap_t \quad \forall t \in T, \tag{4}$$

$$y_{it} \in \{0,1\}, \quad x_{it} \geqslant 0, \quad s_{it} \geqslant 0, \quad s_{i0} \geqslant 0, \quad s_{im} = 0 \quad \forall i \in I, \forall t \in T. \tag{5}$$

The objective function (1) minimizes the total costs, consisting of the setup cost, the variable production cost, the inventory holding cost, and initial inventory cost. Constraints (2) are the demand constraints. To deal with infeasible problems, we allow for initial inventory, which is available in the first period at a large feasibility cost of fc_i (Vanderbeck 1998). There is no setup required for initial inventory. Next, we have the setup forcing (3) and capacity constraints (4). Finally, we have the nonnegativity and integrality constraints (5), and the inventory at the end of the final period is set to zero. Let v_{CLST} be the optimal objective value for problem (1)–(5) and $\overline{v}_{\text{CLST}}$ its LP relaxation. We introduce a more compact notation for the variables: $x_i = (x_{i1}, x_{i2}, \ldots, x_{im})$, $s_i = (s_{i0}, s_{i1}, s_{i2}, \ldots, s_{im})$, and $y_i = (y_{i1}, y_{i2}, \ldots, y_{im})$. Further, we define X^i, the single-item lot-size polytope for each product i, as follows:

$$X^i = \left\{ (x_i, s_i, y_i) \;\middle|\; \begin{array}{l} s_{i,t-1} + x_{it} = d_{it} + s_{it} \quad \forall t \in T; \\ x_{it} \leqslant sd_{itm} y_{it} \quad \forall t \in T; \\ y_{it} \in \{0,1\}, \;\; x_{it} \geqslant 0, \;\; s_{it} \geqslant 0 \quad \forall t \in T; \\ \qquad s_{i0} \geqslant 0; \;\; s_{im} = 0. \end{array} \right\}.$$

We first provide a brief literature review on the capacitated lot-sizing problem (§2). The starting point of this research is the well-known observation that the formulation provided by Manne (1958) has a structural deficiency because it is not equivalent to formulation (1)–(5) and provides only a lower bound. It is important to make a distinction between a formulation, in our case a Dantzig-Wolfe (DW) decomposition reformulation, and a solution method, i.e., column generation and branch-and-price (B&P). We have contributions on both accounts. To the best of our knowledge, this is the first time that a formulation has been proposed for the dynamic lot-sizing problem, which is the equivalent mixed-integer programming (MIP) Dantzig-Wolfe reformulation of the original problem (§3). We show how the integrality restrictions on the original setup variables translate into integrality restrictions on the new master variables by separating the setup and production decisions. The set of extreme points of the lot-size polytope that are needed for the MIP Dantzig-Wolfe reformulation is larger than the set of dominant plans used by Manne. Further, we are the first to implement a branch-and-price algorithm for this problem. In this algorithm, we combine Lagrange relaxation and Dantzig-Wolfe decomposition to speed up the column generation process (§4). Our computational experiments on data sets available from the literature indicate that the B&P algorithm yields high-quality solutions that are comparable to the current state-of-the-art results reported in the literature (§5). Finally, avenues for future research are discussed in the conclusion (§6).

2. Literature Review

The regular CLST formulation, given by the model (1)–(5), usually has a large integrality gap. Much research is devoted to finding better formulations with a smaller gap. The model can be extended with valid inequalities. The formulation with (l, S) inequalities (Barany et al. 1984) describes the convex hull of the single-item uncapacitated lot-sizing polytope. Pochet (1988), Leung et al. (1989), and Miller et al. (2000) derive several other valid inequalities for the capacitated problem. Belvaux and Wolsey (2000, 2001) report on an efficient branch-and-cut system that includes preprocessing and inequalities for a variety of lot-sizing problems. Eppen and Martin (1987) propose a different approach for tightening the formulation using variable redefinition. Kleindorfer and Newson (1975), Thizy and Van Wassenhove (1985), and Trigeiro et al. (1989) propose to dualize the capacity constraint in a Lagrange relaxation approach. A detailed discussion of solution approaches for the CLST can be found in Jans and Degraeve (2007).

Manne (1958) proposes an innovative linear programming formulation for the CLST. He explicitly models all the possible setup schedules. Let $q \in Q^i$, $Q^i = \{(y_{i1}, \ldots, y_{im}) \mid y_{it} \in \{0,1\} \;\forall t \in T\}$ be a feasible setup schedule, and let y_{it}^q be one if there is a setup for item i in period t in setup schedule q, zero otherwise. There are 2^m different setup schedules possible for each product. Exactly one "dominant" production plan corresponds with each setup schedule. Manne considers only dominant production schedules, which have the property that for each period, demand will be met by production in that period if there is a setup, or otherwise from the nearest preceding period with a setup. Manne's dominant production schedules are in fact all the production schedules satisfying the Wagner and Whitin (1958) condition: $s_{i,t-1} x_{it} = 0 \;\forall t \in T, \forall i \in I$. The Wagner-Whitin production plan for item i according to setup schedule q is defined by the production quantities x_{it}^q and is further referred to as the Wagner-Whitin production plan q:

$$x_{it}^q = 0 \quad \text{if } y_{it}^q = 0, = sd_{it,k-1} \text{ otherwise,}$$
$$\text{with } k = \min_{t < l \leqslant m+1} (l\text{: } y_{il}^q = 1 \text{ or } l = m+1). \tag{6}$$

Note that for each schedule q, the inventory variables, s_{it}^q, and the initial inventory variable s_{i0}^q, are automatically determined by the production quantities, x_{it}^q, through the demand constraints (2). Next, the parameters for the total costs and capacity requirement for each schedule are defined as follows:

c_{iq}: total cost of initial inventory, setup, production, and inventory holding for the production of product i according to setup schedule $q, = fc_i s_{i0}^q + \sum_{t \in T} (sc_{it} y_{it}^q + vc_{it} x_{it}^q + hc_{it} s_{it}^q)$. (7)

r_{iqt}: capacity required for setup and variable production time to produce product i according to setup schedule q in period $t, = st_{it} y_{it}^q + vt_{it} x_{it}^q$. (8)

The decision variable is:

z_{iq}: fraction of schedule q for product i that will be produced.

Manne's formulation then, is as follows:

$$\text{Min} \quad \sum_{i\in I}\sum_{q\in Q^i} c_{iq}z_{iq}, \tag{9}$$

$$\text{s.t.} \quad \sum_{q\in Q^i} z_{iq} = 1 \quad \forall i \in I, \tag{10}$$

$$\sum_{i\in I}\sum_{q\in Q^i} r_{iqt}z_{iq} \leqslant cap_t \quad \forall t \in T, \tag{11}$$

$$z_{iq} \geqslant 0 \quad \forall i \in I, \forall q \in Q^i. \tag{12}$$

The objective function (9) minimizes the total cost. In the first constraint (10), we choose a convex combination of schedules for each item. The combination of the chosen schedules must also satisfy the capacity constraint (11). Note that the z_{iq} variables are defined to be continuous (12).

Dzielinski and Gomory (1965) note that Manne's model is actually the LP relaxation of the master for the Dantzig-Wolfe decomposition (Dantzig and Wolfe 1960) with the capacity constraints as the linking constraints and the subproblem being the single-item uncapacitated lot-size problem. It is interesting to note that Manne proposed his formulation in 1958, before the concept of Dantzig-Wolfe decomposition was developed in 1960. Dzielinski and Gomory (1965) propose using column generation to deal with the difficulty of the huge number of variables in Manne's formulation. Column generation starts with a feasible restricted master with only a few columns, and we add new columns iteratively as they are needed. At each iteration of the column generation procedure, we solve a single-item uncapacitated subproblem for each item i, where the objective function is to minimize the reduced cost. Columns with a negative reduced cost are added to the master. Next, we solve the master again as a linear program and try to find new columns that price out with the new dual prices in a new iteration. If we cannot find any new column with a negative reduced cost, we have solved the master's LP relaxation. The optimal value of formulation (9)–(12), $\overline{v}_M$, is equal to or better than the LP relaxation of the original formulation $\overline{v}_{\text{CLST}}$ because the subproblem does not have the integrality property. However, this decomposition formulation with only the dominant production plans has a major structural drawback. Imposing integrality constraints on the z_{iq} variables will not result in an equivalent formulation for the IP problem as formulated by (1)–(5). For the capacitated problem, the optimal solution will usually not consist of pure Wagner-Whitin schedules. This has already been observed by Florian and Klein (1971) for the single-item capacitated lot-sizing problem. Lambrecht and Vanderveken (1979) and Bitran and Matsuo (1986) also notice that the set of feasible solutions for Manne's model with integrality constraints is only a subset of the feasible solutions for the original integer problem. The main reason for this problem is that a solution for the subproblem, i.e., a new column, consists of both a setup and production quantity decision. The setup decision automatically determines the production decision according to the Wagner-Whitin property (6). Finally, note that many different Dantzig-Wolfe decomposition schemes are possible, depending on the selection of different subsystems (Jans and Degraeve 2004).

3. A MIP Dantzig-Wolfe Decomposition Approach for the CLST

Dantzig-Wolfe decomposition can be described as a special form of variable redefinition in which the original variables are replaced by a convex combination of the extreme points of a subsystem. We assume that the polyhedron is bounded, which is the case for the lot-sizing problem because the ending inventory must be zero. Otherwise, a linear combination of extreme rays is also needed. The LP relaxation of the Dantzig-Wolfe reformulation provides a tighter bound unless the subsystem has the integrality property. To ensure the equivalence between the Dantzig-Wolfe reformulation and the original formulation, requiring that no feasible solutions are lost or created (Martin 1987), the integrality restrictions are imposed on the original variables (Barnhart et al. 1998). Vanderbeck (2000) calls this the convexification approach to the Dantzig-Wolfe decomposition. He further points out that for an integer programming problem, there exists an alternative Dantzig-Wolfe reformulation based on the discretization of the integer subsystem, in which all the integer solutions are enumerated. The discretization approach also uses the interior points of the integer polyhedron and not just the extreme points. This allows imposing the integrality constraints directly on the new variables. For the special case where the variables are restricted to be binary, the convexification and discretization approaches are identical because there are no interior points in the integer subsystem.

The CLST is a MIP problem. Even though the integer variables are restricted to be binary, we cannot impose integrality conditions on the new decomposition variables, which is clearly illustrated by Manne's reformulation. The extreme points of the single-item lot-size polytope X^i will not necessarily provide an optimal solution to the overall problem. Whereas Manne (1958) proved that you only need the dominant schedules to obtain the LP relaxation of the Dantzig-Wolfe reformulation, we will prove in this section that an equivalent reformulation can be obtained by applying the Dantzig-Wolfe decomposition approach to formulation (1)–(5). We will show in several steps how this translates into an equivalent master problem with the integrality restriction on the new master variables. The key observation is that Manne's dominant plans are only a subset of the set of extreme points needed.

Step 1. *Equivalent MIP Dantzig-Wolfe Reformulation*. Let $p = (x_i^p, s_i^p, y_i^p)$ be an extreme point of $\text{conv}(X^i)$ and

let P^i be the set of all the extreme points of conv(X^i). For each extreme point p of conv(X^i), we define a new variable z_p^i. The correct application of the Dantzig-Wolfe decomposition principle requires substituting the original variables by a convex combination of the extreme points of the substructure in the objective function (13) and the linking constraints (14). The relationship between the original production and setup variables and the convex combination is given in (16) and (17). The integrality constraints must be imposed on the original setup variables (18). For the production variables, we can omit this relationship (17). The convexity constraint (15) and nonnegativity constraint (19) enforce a convex combination. The Step 1 formulation is, then, as follows:

$$\text{Min} \quad \sum_{i\in I}\sum_{p\in P^i}\left(fc_i s_{i0}^p+\sum_{t\in T}(sc_{it}y_{it}^p+vc_{it}x_{it}^p+hc_{it}s_{it}^p)\right)z_p^i, \quad (13)$$

$$\text{s.t.} \quad \sum_{i\in I}\sum_{p\in P^i}(y_{it}^p st_{it}+x_{it}^p vt_{it})z_p^i \leqslant cap_t \quad \forall t\in T, \quad (14)$$

$$\sum_{p\in P^i} z_p^i = 1 \quad \forall i\in I, \quad (15)$$

$$y_{it}=\sum_{p\in P^i} y_{it}^p z_p^i \quad \forall i\in I, \forall t\in T, \quad (16)$$

$$x_{it}=\sum_{p\in P^i} x_{it}^p z_p^i \quad \forall i\in I, \forall t\in T, \quad (17)$$

$$y_{it}\in\{0,1\} \quad \forall i\in I, \forall t\in T, \quad (18)$$

$$z_p^i \geqslant 0 \quad \forall i\in I, \forall p\in P^i. \quad (19)$$

Step 2. *Alternative Description of the Set of Extreme Points.* In the first step, we defined P^i generally as the set of all the extreme points of conv(X^i). In this second step, we define this set more explicitly. Proposition 1 states that P^i consists of all the feasible solutions of X^i that also satisfy the Wagner-Whitin property. This set is much larger than the set of the 2^m dominant plans that Manne used because the set of extreme points also includes nondominant plans satisfying the Wagner-Whitin property, in which it is possible that there is a setup, but no production for some time periods. In Proposition 2, we establish the exact cardinality of this set.

PROPOSITION 1. $P^i=\{(x_i, s_i, y_i) \mid (x_i, s_i, y_i)\in X^i;\ s_{i,t-1}x_{it}=0\ \forall t\in T\}$.

PROOF. See Appendix A. □

PROPOSITION 2. $|P^i|=3^m$.

PROOF. For each of the m periods, there are exactly three possibilities for the setup and production variable: $y_{it}=0$ and $x_{it}=0$; $y_{it}=1$ and $x_{it}=0$; $y_{it}=1$ and $x_{it}>0$. Combining the possibilities over the m periods so that both the constraints of X^i and the Wagner-Whitin condition are satisfied gives 3^m extreme points in total. □

For each setup schedule $q\in Q^i$, we define the subset Q^{iq} of associated setup schedules as follows:

$$Q^{iq}=\{(y_{i1},\ldots,y_{im}) \mid y_{it}\leqslant y_{it}^q\ \forall t\in T;\ y_{it}\in\{0,1\}\ \forall t\in T\}.$$

Only if there is a setup in setup schedule q for a specific period, is a setup possible, but not required, for that period in a setup schedule in the subset. If there are s setups in schedule q, then Q^{iq} contains 2^s setup schedules. For each setup schedule $v\in Q^{iq}$, the associated dominant Wagner-Whitin production quantities x_{it}^v are uniquely defined (6). The setup schedule q can be combined with any of the Wagner-Whitin production plans $x_i^v \mid v\in Q^{iq}$ to form an extreme point. Because we have established a one-to-one relationship between extreme points of X^i and the feasible solutions satisfying the Wagner-Whitin property, we can now redefine P^i as follows:

$$P^i=\{(x_i^v, s_i^v, y_i^q) \mid q\in Q^i, v\in Q^{iq}\}.$$

Step 3. *Rewriting the Dantzig-Wolfe Reformulation.* We rewrite the model in terms of convex combinations of extreme points, using the fact that we can define the set P^i in terms of Q^i and Q^{iq}: $\sum_{p\in P^i} z_p^i=\sum_{q\in Q^i}\sum_{v\in Q^{iq}} z_{qv}^i$. The variable z_{qv}^i refers to the extreme point formed by taking setup schedule $q\in Q^i$ and the dominant Wagner-Whitin production plan associated with the setup schedule $v\in Q^{iq}$. The Step 3 formulation, then, is as follows:

$$\text{Min} \quad \sum_{i\in I}\sum_{q\in Q^i}\sum_{v\in Q^{iq}}\left(fc_i s_{i0}^v+\sum_{t\in T}(sc_{it}y_{it}^q+vc_{it}x_{it}^v+hc_{it}s_{it}^v)\right)z_{qv}^i, \quad (20)$$

$$\text{s.t.} \quad \sum_{i\in I}\sum_{q\in Q^i}\sum_{v\in Q^{iq}}(y_{it}^q st_{it}+x_{it}^v vt_{it})z_{qv}^i \leqslant cap_t \quad \forall t\in T, \quad (21)$$

$$\sum_{q\in Q^i}\sum_{v\in Q^{iq}} z_{qv}^i = 1 \quad \forall i\in I, \quad (22)$$

$$y_{it}=\sum_{q\in Q^i}\sum_{v\in Q^{iq}} y_{it}^q z_{qv}^i \quad \forall i\in I, \forall t\in T, \quad (23)$$

$$y_{it}\in\{0,1\} \quad \forall i\in I, \forall t\in T, \quad (24)$$

$$z_{qv}^i \geqslant 0 \quad \forall i\in I, q\in Q^i, v\in Q^{iq}. \quad (25)$$

Step 4. *Integrality Constraints.* Define a new variable zs_q^i as follows:

$$zs_q^i=\sum_{v\in Q^{iq}} z_{qv}^i \quad \forall q\in Q^i,\ \forall i\in I.$$

This enables us to rewrite the original setup variables (23):

$$y_{it}=\sum_{q\in Q^i}\sum_{v\in Q^{iq}} y_{it}^q z_{qv}^i=\sum_{q\in Q^i} y_{it}^q\left(\sum_{v\in Q^{iq}} z_{qv}^i\right)$$
$$=\sum_{q\in Q^i} y_{it}^q zs_q^i \quad \forall i\in I,\ \forall t\in T,$$

and the convexity constraint (22) as $\sum_{q\in Q^i} zs_q^i=1\ \forall i\in I$. We next prove the equivalence between the integrality constraints on the original variables y_{it} and the integrality constraint on the new master variables zs_q^i.

PROPOSITION 3. $y_{it} \in \{0,1\} \Leftrightarrow zs_q^i \in \{0,1\}$.

PROOF. (1) $y_{it} \in \{0,1\} \Leftarrow zs_q^i \in \{0,1\}$. The restrictions that $zs_q^i \in \{0,1\}$ and $\sum_{q\in Q^i} zs_q^i = 1\ \forall i \in I$ imply that exactly one zs_q^i will be equal to one in the sum $\sum_{q\in Q^i} zs_q^i$. Because $y_{it}^q \in \{0,1\}$, it follows that $\sum_{q\in Q^i} y_{it}^q zs_q^i = y_{it} \in \{0,1\}$.

(2) $y_{it} \in \{0,1\} \Rightarrow zs_q^i \in \{0,1\}$. Given that $y_{it} \in \{0,1\}$, assume that there exists a feasible solution in which for some product i, we have a fractional solution in the zs_q^i variables, i.e., $\exists q_1^i, q_2^i \in Q^i \mid q_1^i \neq q_2^i; zs_{q1}^i + zs_{q2}^i = 1; zs_{q1}^i, zs_{q2}^i \notin \{0,1\}; zs_{q1}^i, zs_{q2}^i \geqslant 0$. Because the two setup schedules q_1^i, q_2^i are different, there is at least one period k for which $y_{ik}^{q1} \neq y_{ik}^{q2}$. Combined with the fact that $y_{ik}^{q1}, y_{ik}^{q2} \in \{0,1\}$, this implies that $y_{ik} = \sum_{q\in Q^i} y_{ik}^q zs_q^i = y_{ik}^{q1} zs_{q1}^i + y_{ik}^{q2} zs_{q2}^i \notin \{0,1\}$. This is in conflict with the fact that $y_{it} \in \{0,1\}$—hence, our assumption that a solution exists in which some of the zs_q^i variables are fractional is wrong. □

Due to Proposition 3, we can drop the integrality constraints on the original y_{it} variables and impose the integrality constraints on the new master variables zs_q^i. As a result, we can also drop the definition constraints for the y_{it} variables. Consequently, this new formulation does not contain any of the original variables x_{it} and y_{it} and is as follows:

$$\text{Min} \sum_{i\in I}\sum_{q\in Q^i}\sum_{v\in Q^{iq}}\left(fc_i s_{i0}^v + \sum_{t\in T}(vc_{it}x_{it}^v + hc_{it}s_{it}^v)\right)z_{qv}^i + \sum_{i\in I}\sum_{q\in Q^i}\left(\sum_{t\in T} sc_{it}y_{it}^q\right)zs_q^i, \tag{26}$$

$$\text{s.t.} \sum_{i\in I}\sum_{q\in Q^i} y_{it}^q st_{it} zs_q^i + \sum_{i\in I}\sum_{q\in Q^i}\sum_{v\in Q^{iq}} x_{it}^v vt_{it} z_{qv}^i \leqslant cap_t \quad \forall t \in T, \tag{27}$$

$$\sum_{q\in Q^i} zs_q^i = 1 \quad \forall i \in I, \tag{28}$$

$$zs_q^i = \sum_{v\in Q^{iq}} z_{qv}^i \quad \forall i \in I,\ q \in Q^i, \tag{29}$$

$$zs_q^i \in \{0,1\} \quad \forall i \in I,\ q \in Q^i, \tag{30}$$

$$z_{qv}^i \geqslant 0 \quad \forall i \in I,\ q \in Q^i,\ v \in Q^{iq}. \tag{31}$$

The variables have the following interpretation: $zs_q^i = 1$ if for product i setup schedule q is selected, 0 otherwise; and z_{qv}^i is the fraction used of the Wagner-Whitin production plan of setup schedule $v \in Q^{iq}$. The convexity constraint (28) together with the integrality constraints on the z_{qv}^i variables (30) enforces the selection of exactly one setup schedule for each item. To determine the production quantities, we make a convex combination of the Wagner-Whitin production plans associated with setup schedules from the subset of the chosen setup schedule (29). The intuitive idea behind the use of this subset is that all feasible production quantities for a given setup schedule can be obtained as a convex combination of the production quantities of the 2^m different Wagner-Whitin production plans. Obviously, the Wagner-Whitin production plans in which there is production in a time period for which the chosen setup schedule does not have a setup can be excluded because you cannot have production in the first place if there is no setup. Finally, constraint (27) is the capacity constraint, taking into account both the setup times and variable production times.

In this model, we have overcome the problem of Manne's formulation by separating the integer setup and the continuous production quantity decisions using the variables zs_q^i and z_{qv}^i. We have proven that the formulation (26)–(31) is equivalent to the MIP Dantzig-Wolfe decomposition reformulation. For the LP relaxation of this reformulation, we can substitute out the zs_q^i variable by constraint (29). The optimal value of the resulting LP, $\bar{v}_{\text{DWCL}}$, is the same as the optimal solution of Manne's model, $\bar{v}_M$. The proof is presented in Appendix B.

Step 5. *Formulation with Fewer Variables*. In a final step, we further simplify the formulation. The cost coefficient of the z_{qv}^i variable, i.e., $fc_i s_{i0}^v + \sum_{t\in T}(vc_{it}x_{it}^v + hc_{it}s_{it}^v)$, does not depend on q, but only on v. The same observation holds for the capacity coefficient, i.e., $x_{it}^v vt_{it}$. Hence, we can define a new variable: $zp_v^i = \sum_{q\in Q^i \mid v\in Q^{iq}} z_{qv}^i$. For the chosen setup schedule, constraint (35) enforces that the sum of the associated production plans must be larger than one. However, because this is a minimization problem with positive objective function coefficients (i.e., assuming positive production and holding cost coefficients), their sum will equal exactly one. The constraint is still valid for the other setup schedules because their value zs_q^i will be zero. This formulation has $2 * 2^m$ variables and is as follows:

$$\text{Min} \sum_{i\in I}\sum_{v\in Q^i}\left(fc_i s_{i0}^v + \sum_{t\in T}(vc_{it}x_{it}^v + hc_{it}s_{it}^v)\right)zp_v^i + \sum_{i\in I}\sum_{q\in Q^i}\left(\sum_{t\in T} sc_{it}y_{it}^q\right)zs_q^i, \tag{32}$$

$$\text{s.t.} \sum_{i\in I}\sum_{q\in Q^i} y_{it}^q st_{it} zs_q^i + \sum_{i\in I}\sum_{v\in Q^i} x_{it}^v vt_{it} zp_v^i \leqslant cap_t \quad \forall t \in T, \tag{33}$$

$$\sum_{q\in Q^i} zs_q^i = 1 \quad \forall i \in I, \tag{34}$$

$$zs_q^i \leqslant \sum_{v\in Q^{iq}} zp_v^i \quad \forall i \in I,\ q \in Q^i, \tag{35}$$

$$zs_q^i \in \{0,1\} \quad \forall i \in I,\ q \in Q^i, \tag{36}$$

$$zp_q^i \geqslant 0 \quad \forall i \in I,\ q \in Q^i. \tag{37}$$

The existence of such a huge number of variables in the various formulations makes the problem well suited for column generation.

4. The Branch-and-Price Algorithm

In this section, we describe the most important building blocks of the algorithm. We start with an initial heuristic

that provides a good upper bound. Next, we do column generation to find the LP optimum, which is a lower bound for the IP optimum. We speed up the column generation process by using a combination of simplex and subgradient optimization. At each iteration, we also try to construct feasible upper bounds. Finally, we combine column generation and branch-and-bound in a branch-and-price algorithm. Our algorithm is based on formulation (20)–(25).

4.1. Initial Heuristic

The first step in the algorithm is finding a heuristic upper bound. We use the efficient algorithm proposed by Trigeiro, Thomas, and McClain (1989) (TTM). This heuristic consists of Lagrange relaxation and a smoothing heuristic to create feasible production plans. We improve the upper bounds found by TTM in two ways. First, we fix all the setup variables y_{it} to one or zero according to the solution proposed by the TTM heuristic. If there is any production in period t for item i, we fix the setup variable y_{it} to one, otherwise to zero. Next, we solve the remaining problem to find the optimal production quantities. It is well known that the resulting problem after fixing the setup variables is a network problem (e.g., Thizy and Van Wassenhove 1985) for which efficient algorithms exist. This procedure is referred to as the network heuristic (NH). A second improvement is a lot elimination heuristic (LEH). Starting again from the setup solution given by TTM, we try to improve this solution by checking if we can eliminate a setup. We first check the items with the highest setup cost, starting with the setup at the end of the time horizon and moving to the beginning. If the setup variable equals one, we fix it to zero and solve a new network problem to find the optimal production quantities. If the upper bound improves, we keep that setup variable at zero; otherwise, we set it back to one and continue.

4.2. Column Generation at the Root Node

Next, we start the column generation procedure (CG). We first add some initial columns to the master. For each item, we add the uncapacitated Wagner-Whitin solution. Columns where all the demand is met from initial inventory are also added. The single-item uncapacitated subproblem is solved using an efficient implementation of the Wagner-Whitin algorithm. Let $\overline{v}^r_{\mathrm{DWCL}}$ be the objective value of the restricted master at pricing step r and let rc^r_i be the reduced costs of the columns that we generate at iteration r. If no column was added for an item, the reduced cost equals zero. At each iteration, we calculate a lower bound on $\overline{v}_{\mathrm{DWCL}}$, the optimal LP solution of the Dantzig-Wolfe reformulation (Lasdon 1970):

$$\mathrm{LB} = \overline{v}^r_{\mathrm{DWCL}} + \sum_{i \in P} rc^r_i. \tag{38}$$

If the current restricted master solution, $\overline{v}^r_{\mathrm{DWCL}}$, is equal to the lower bound (38), then no column prices out favorably and we stop the column generation process.

4.3. Hybrid Simplex/Subgradient Optimization

To accelerate the column generation algorithm, we combine it with a Lagrange relaxation approach in a hybrid simplex/subgradient optimization algorithm. In the Lagrange problem, the capacity constraint is dualized into the objective function with a specific set of nonnegative multipliers u (Kleindorfer and Newson 1975). The Lagrange problem also decomposes into single-item uncapacitated lot-sizing problems. A strong relationship exists between Dantzig-Wolfe decomposition and Lagrange relaxation (Geoffrion 1974). The optimal values of the relaxed Dantzig-Wolfe reformulation and the Lagrange dual are the same. Because both methods have the same subproblem, we can use both to generate columns. Also, the optimal dual variables for the linking constraints in the Dantzig-Wolfe master correspond to optimal multipliers for the complicating constraint in the Lagrange objective function. The hybrid simplex/subgradient optimization procedure iterates between column generation and Lagrange relaxation, and exchanges information. Basically, it consists of a nested double loop. After solving the restricted master by the simplex method in the outer loop, we perform several iterations of Lagrange relaxation in the inner loop. Subgradient optimization is used to update the dual prices, starting from the latest simplex dual prices. This provides us with new columns because the Lagrange subproblem is identical to the column generation subproblem. Next, we add the new columns that are generated in the Lagrange iterations and we reoptimize the restricted master. This procedure speeds up the convergence of the column generation process. Further details on this procedure can be found in Degraeve and Jans (2003), Degraeve and Peeters (2003), and Huisman et al. (2005).

4.4. Calculating Upper Bounds

Calculating feasible upper bounds during column generation is done in two ways. In a first heuristic, we start from the optimal LP solution of the current master. We round the fractional setup variables to either one or zero according to some cutoff point. Next, a network problem is solved to determine the optimal production quantities for this fixed setting of the setup variables. This procedure is repeated for different values of cutoff points (repeated rounding heuristic (RRH)). We typically observe a U shape in the solution values when increasing the cutoff point. Therefore, we cut this heuristic short when the solution starts to increase. This heuristic is done at every pricing step of the column generation iteration, starting from the new primal solution given by the current master. Our preliminary experiments have indicated that this yields better results than when the heuristic is applied only once at termination. In a second heuristic, we take the solutions given by the subproblems and perform the smoothing subroutine proposed by Trigeiro et al. (1989).

4.5. Branch and Price

Branching on the new master variables zs_q^i in formulation (26)–(31) or (32)–(37) would lead to an unbalanced tree. Instead, we branch on the original variables $y_{it} = \sum_{q\in Q^i}\sum_{v\in Q^{iq}} y_{it}^q z_{qv}^i$ in formulation (20)–(25). By branching on the original setup variables, we actually branch on groups of columns, and this will lead to a more balanced enumeration tree (Vanderbeck 2000, Barnhart et al. 1998). Therefore, the implementation of our branch-and-price algorithm is based on formulation (20)–(25). The branch-and-price algorithm consists of three major subroutines: branch, twin, and backtrack. Our search strategy is depth first, where branching is done on the current node. The branching decisions are enforced by adapting or deleting old columns and generating new ones. We use insights from formulation (26)–(31) in the branching scheme. When we branch up—i.e., we force y_{it} to one—we do not delete the columns in the master that do not have a setup in period t, but we adjust them, enforcing a setup by adapting the column's cost and capacity requirement to reflect this setup. In this way, we ensure that the setup cost and setup time is properly accounted for in all the available columns according to the branching decision. The advantage of adjusting the columns is that we avoid the potential need for regenerating these columns later. In the subproblem, we enforce a setup by fixing the setup variable to one, but this does not necessarily imply that there must be some positive production. This also allows us to work with a master that contains only the convexity (22) and capacity (21) constraints in each node of the tree. The simplicity of this master directly determines the effectiveness of our solution approach. The columns consisting of initial inventory only are used to maintain feasibility at each node in the branch-and-price algorithm. Details on the branch, twin, and backtrack procedures can be found in Degraeve and Jans (2003). After we have imposed the branching adjustments, we solve the master. If the current objective value is larger than or equal to v_{UB}, our best upper bound, then we do column generation to see if the objective value can drop under this upper bound by generating more columns. We stop column generation at a node if no column prices out. We do not have to investigate a node further if during column generation the lower bound on the master (38) exceeds the current upper bound v_{UB}. At each step of the column generation process, we also do the network and smoothing heuristic to obtain better upper bounds.

5. Computational Results

In §§5.1 to 5.4, we report in detail on computational experiments with the set of 540 test instances used by TTM. The instances have a time horizon of 20 periods, and the total set consists of three subsets with 180 problems for each case of 10, 20, or 30 products. We report the averages for each class of 10, 20, and 30 products. In §5.5, we discuss results on other data sets. Our algorithms were coded in Fortran using the WATCOM Fortran compiler 10.6 and linked with the LINDO library version 5.3 (Schrage 1995). We use the network solver subroutine available in the LINDO library for solving the network problems. The tests were done on a Pentium III 750 MHz computer under the Windows 2000 operating system. CPU times are given in seconds, and the gap is calculated as the percentage difference between the best upper bound and lower bound, compared to the best lower bound.

5.1. Initial Heuristics

We tested the effectiveness of the various algorithmic steps in several experiments. The first heuristic (TTM) is the Trigeiro et al. (1989) algorithm. We used their original code and experimented with different Lagrange iteration limits. For further experiments, we use 300 iterations (TTM-300) because our initial experiments indicated that the improvement leveled off beyond this limit. TTM performed 150 steps of the Lagrange iteration (TTM-150). In Table 1, we report on the average gap between the Lagrange lower bound and best feasible solution and the CPU time. The remainder of Table 1 summarizes the gap and CPU time for the improvement heuristics performed after TTM. The network heuristic (NH) further reduces the gap with a relatively small amount of extra CPU time. After the network heuristic, we do the lot elimination heuristic (LEH), with a limited search over the first $0.3 * n * m$ setup variables that are at one.

5.2. Solving the Root Node

Up to this point, the gap is still calculated relative to the Lagrange lower bound from TTM, which is equal to or lower than the optimal column generation-based lower bound $\bar{v}_{\mathrm{DWCL}}$. In the next step, column generation is performed at the root node to obtain this exact lower bound $\bar{v}_{\mathrm{DWCL}}$. Comparing the heuristic solutions of LEH to the exact lower bound $\bar{v}_{\mathrm{DWCL}}$ obtained by column generation instead of the Lagrange lower bound results in an average gap of 3.13%, 1.22%, and 0.78% for 10, 20, and 30 products, respectively. During column generation, we also perform some primal heuristics. We implement the repeated rounding heuristic (RRH) and the smoothing heuristic (SH), as discussed in §4. We speed up the column

Table 1. Initial heuristics and solving the root node problem.

	10 products		20 products		30 products	
	Gap	Time	Gap	Time	Gap	Time
TTM-150	3.81	0.24	1.57	0.43	1.09	0.57
TTM-300	3.71	0.43	1.50	0.79	1.05	1.07
NH	3.58	0.48	1.44	0.84	1.02	1.13
LEH	3.33	0.55	1.35	1.11	0.95	1.72
CGH	3.10	0.91	1.20	1.83	0.72	2.90
CGS	3.07	1.43	1.20	2.71	0.71	4.09

generation process by using the hybrid simplex/subgradient optimization procedure. Within two simplex iterations, we do a maximum of 25 Lagrange iterations. We report the results of the best upper bound after column generation with the hybrid method in the row CGH in Table 1. We also report on the algorithm where we do column generation at the root node with only simplex optimization (CGS). CGS has a substantially larger CPU time and a slightly better gap compared to CGH. With CGH, we do fewer pricing iterations and hence perform the heuristics (RRH and SH) fewer times. Therefore, we have less chance of finding a good upper bound. The slightly better quality of the upper bound for CGS accounts for the better gap.

5.3. Branch-and-Price

In our branch-and-price algorithm, we tested five different branching strategies, depending on the selection of the branching variable:

B&P1: First fractional variable, order the items by input list,

B&P2: First fractional variable, order the items by decreasing setup cost,

B&P3: Fractional variable closest to 0.5,

B&P4: Fractional variable closest to zero or one,

B&P5: Fractional variable closest to one.

In a heuristic implementation of the branch-and-price algorithm, we fix some of the y_{it} variables depending on their value in the primal solution at the end of CG at the root node. We fix all the variables below 0.05 to zero and above 0.95 to one. Next, we solve the smaller problem with the remaining variables optimally, using the branch-and-price algorithm. We call this reduced branch-and-price (RB&P). For all the implementations reported, we used a limit of 2,000 nodes. The summary results for the various branching strategies are given in Table 2.

Table 2. Results for the branch-and-price algorithm (maximum of 2,000 nodes).

	10 products		20 products		30 products	
	Gap	Time	Gap	Time	Gap	Time
B&P1	2.83	45.80	1.14	65.32	0.69	99.00
B&P2	2.75	51.49	1.08	75.85	0.65	111.71
B&P3	2.99	52.68	1.17	74.30	0.71	110.76
B&P4	2.74	32.86	1.09	51.63	0.65	83.36
B&P5	2.58	48.10	1.05	70.87	0.62	116.05
RB&P1	2.79	20.96	1.11	26.44	0.65	31.79
RB&P2	2.69	21.72	1.07	25.50	0.63	37.32
RB&P3	2.84	22.46	1.10	28.72	0.64	38.11
RB&P4	2.76	18.96	1.08	23.84	0.67	31.35
RB&P5	2.74	24.53	1.06	27.87	0.64	35.50

On average, we obtain the smallest gaps for the fifth branching strategy, where we branch on the fractional variable closest to one. B&P4, where we branch on the variable closest to zero or one takes, on average, the least CPU time, and is almost always better than B&P1, B&P2, and B&P3. The reduced branch-and-price is roughly two to three times faster compared to the optimal implementation. The gap can be either better or worse. RB&P2 seems to give the best gaps on average. With TTM, 10.7% of the instances could be solved to optimality. This percentage increases to 28.5% with B&P5.

The data set from TTM is constructed according to a full factorial experiment with five factors: capacity utilization (low, medium, or high), number of items (10, 20, or 30), time between orders (TBO) (low, medium, or high), demand variability (medium or high), and setup time (low or medium). In Table 3, we compare the effect on the gap of the different factors, as calculated with three different procedures: the TTM heuristic, the hybrid column generation at the root node, and the branch-and-price algorithm. We

Table 3. Gap results for the full factorial experiment.

	10 products			20 products			30 products		
	TTM	Root	B&P5	TTM	Root	B&P5	TTM	Root	B&P5
Capacity usage									
Low	0.79	0.76	0.58	0.17	0.17	0.13	0.13	0.08	0.05
Medium	2.23	2.01	1.48	0.65	0.62	0.49	0.29	0.29	0.24
High	8.10	6.53	5.66	3.68	2.80	2.52	2.72	1.81	1.55
TBO									
Low	1.91	1.59	1.51	0.94	0.67	0.64	0.75	0.42	0.39
Medium	2.54	2.28	1.77	0.99	0.89	0.76	0.62	0.50	0.43
High	6.67	5.45	4.45	2.58	2.02	1.74	1.77	1.25	1.03
Demand variability									
Medium	3.99	3.34	2.61	1.57	1.23	1.07	1.17	0.80	0.66
High	3.42	2.87	2.54	1.44	1.16	1.03	0.92	0.65	0.57
Setup time									
Low	3.85	3.18	2.60	1.73	1.29	1.12	1.30	0.85	0.68
High	3.56	3.03	2.55	1.27	1.10	0.98	0.80	0.60	0.55
Average	**3.71**	**3.10**	**2.58**	**1.50**	**1.20**	**1.05**	**1.05**	**0.72**	**0.62**

Table 4. CPU times for the full factorial experiment.

	10 product			20 product			30 product		
	TTM	Root	B&P5	TTM	Root	B&P5	TTM	Root	B&P5
Capacity usage									
Low	0.37	0.62	24.16	0.65	1.19	38.34	0.76	1.66	58.13
Medium	0.44	0.83	44.44	0.79	1.63	73.92	1.09	2.37	118.56
High	0.49	1.29	75.70	0.92	2.68	100.35	1.36	4.68	171.46
TBO									
Low	0.34	0.85	36.82	0.58	1.68	46.96	0.71	2.46	83.31
Medium	0.45	0.86	50.44	0.83	1.83	77.13	1.14	2.84	118.77
High	0.51	1.04	57.04	0.94	1.98	88.53	1.37	3.41	146.08
Demand variability									
Medium	0.41	0.90	47.90	0.76	1.80	70.29	1.04	2.88	117.76
High	0.45	0.93	48.30	0.81	1.86	71.46	1.11	2.93	114.34
Setup time									
Low	0.43	0.94	50.03	0.80	1.92	73.64	1.08	3.02	120.05
High	0.43	0.89	46.17	0.77	1.74	68.11	1.06	2.79	112.05
Average	**0.43**	**0.91**	**48.10**	**0.79**	**1.83**	**70.87**	**1.07**	**2.90**	**116.05**

give separate results for the problems with 10, 20, and 30 items. In Table 4, we report the CPU times. The results confirm the conclusions of TTM. Demand variability and setup time have a minor effect on the gap and CPU times. The relative differences in gap between medium and high demand variability and low and high setup times seem to decrease for the solutions at the root node and at the end of the branch-and-price algorithm, compared to the TTM heuristic. If the capacity is more constrained, the problems become more difficult to solve with respect to both the gap and CPU time. Problems with a higher TBO are also more difficult to solve. The effect on the gap of a low and medium TBO seems only minor, whereas the effect of a high TBO is more apparent.

5.4. Comparison with Other Approaches

Compared to the Lagrange heuristic by TTM, our algorithm is able to reduce the gaps substantially further. Other approaches have been proposed in the literature to solve the CLST. Belvaux and Wolsey (2000) describe a branch-and-cut algorithm that is specifically developed to solve lot-sizing problems. They report on six instances taken from a test set used by TTM. Belvaux and Wolsey (2000) used unit variable production times $vt_{it} = 1$ for all their data sets, whereas for one of them, specifically G30, the original data set has fractional variable production times. In Table 5, we report the results for the test problems. G30b refers to the G30 data set with unit variable production times. Belvaux and Wolsey use a 200 MHz computer under Windows NT and set a time limit of 900 seconds. We have a limit of 2,000 nodes. Although we cannot make any robust conclusion based on such a limited comparison, the branch-and-cut system seems to perform better on the smaller problems and our algorithm seems to perform slightly better on the larger problems. For the smaller problems, their branch-and-cut algorithm gives a better lower bound at the root node, whereas for the larger problems, both procedures yield the same lower bound. The results indicate that these CLST problems are indeed hard to solve.

The problem can also be solved using the network reformulation (Eppen and Martin 1987). We used the MIP solver of LINDO. The row EMLP in Table 6 indicates the time needed to compute the LP relaxation. This is much higher than the time it takes with the column generation approach

Table 5. Comparison of branch-and-cut and branch-and-price.

	Branch-and-cut Belvaux and Wolsey (2000)				Branch-and-price (B&P5)			
	LP	IP	Time	Gap	LP	IP	Time	Gap
Tr6-15 (G30)	—	—	—	—	37,103.1	37,809	33	1.90
Tr6-15 (G30b)	37,213.3	37,721*	38.4	1.09	37,201.2	38,162	29	2.51
Tr6-30 (G62)	60,979.4	61,806	900	1.36	60,946.2	62,644	359	2.79
Tr12-15 (G53)	73,858.2	74,799	900	1.27	73,847.9	75,035	66	1.61
Tr12-30 (G69)	130, 177	132,650	900	1.90	130,177.2	131,234	215	0.81
Tr24-15 (G57)	136, 366	136,872	900	0.37	136,365.7	136,860	44	0.36
Tr24-30 (G72)	287, 753	288,424	900	0.23	287,753.4	288,383	306	0.22

*Indicates a proven optimal solution.

Table 6. Variable redefinition results.

	10 products		20 products		30 products	
	Gap	Time	Gap	Time	Gap	Time
EMLP		1.23		5.08		12.52
EMIP	4.49	2,949	2.94	6,077	2.35	4,904

(see Table 1). The LP relaxation of the network reformulation gives the same lower bound, but does not provide an upper bound. The row EMIP reports on the gap and computation time for the IP solution. For 10 and 20 products, we set a pivot limit of 10 million; for 30 products, we set a pivot limit of five million. We observe that the network reformulation is much slower and does not give the same good-quality solutions as our procedure.

Finally, Gopalakrishnan et al. (2001) develop a customized tabu search algorithm for the CLST. They test their procedure on the data set from TTM with the 540 problem instances and report an average gap of 4.01% and an average CPU time of 97 seconds on a Pentium 550 MHz processor. With our algorithm, we obtain an average gap of 1.67% and an average CPU time of 1.9 seconds at the root node (CGH). For our branch-and-price implementation (B&P5), we have an average gap of 1.42% and average time of 79 seconds. Our algorithm is clearly superior to the tabu search.

5.5. More Results and Comparisons

We also tested our procedure on other data sets from TTM. These are 70 instances from the F-set and 71 from the G-set. All the instances in the F-set are 6 products and 15 period problems. The G-set consists of 46 instances with 6 products and 15 periods and 5 instances for each of the cases with 12 products and 15 periods, 24 products and 15 periods, 6 products and 30 periods, 12 products and 30 periods, and 24 products and 30 periods. In Table 7, we give the gap and CPU time for the initial heuristic, which includes TTM, NH, and LEH, for the column generation at the root node; and for the branch-and-price algorithm using B&P5 with a 2,000 node limit. We compare this to the Eppen and Martin formulation solved by LINDO with a maximum of five million pivots. Our branch-and-price algorithm performs better than the network reformulation.

Finally, we tested our algorithm on the capacitated lot-sizing problem without setup times. We did our computational experiments on the data sets from Cattrysse et al. (1990). They have three data sets with 40 instances each. The first one has instances with 50 items and 8 periods, the second has 20 items and 20 periods, and the third has 8 items and 50 periods. In Table 8, we report the average gap and time for the initial heuristic, the column generation at the root node, and the branch-and-price algorithm using B&P5 with a 2,000 node limit. We compare this with the results from Cattrysse et al. (1990) for their best implementation, which is called Heur4 in their paper. The gaps are calculated compared to our lower bounds. We also give the average time that they reported, using an Olivetti M24 with an 8086/8087 processor and 8 MHz. Our gaps are substantially better compared to the solutions obtained by Cattrysse et al. For the first set, the column generation and branch-and-price algorithm are very effective in further closing the

Table 7. Results for F and G data sets from Trigeiro et al. (1989).

	Degraeve and Jans (2003)						Eppen and Martin (1987)	
	Initial heuristics		CG root		B&P		B&B	
	Gap	Time	Gap	Time	Gap	Time	Gap	Time
F	3.69	0.17	3.55	0.28	2.87	28.45	8.99	721.63
G6-15	4.96	0.18	4.73	0.33	3.82	29.30	5.82	770.96
G12-15	1.11	0.34	1.07	0.56	1.00	45.21	3.69	1,994.30
G24-15	0.36	0.75	0.36	1.10	0.33	62.11	0.45	3,754.06
G6-30	3.22	0.68	3.22	1.06	2.86	317.13	2.16	2,871.99
G12-30	1.15	1.49	1.15	2.12	0.87	240.32	0.86	5,690.63
G24-30	0.24	3.43	0.24	4.66	0.20	383.49	1.00	11,596.28

Table 8. Results for data sets from Cattrysse et al. (1990).

	Degraeve and Jans (2003)						Cattrysse et al. (1990)	
	Initial heuristics		CG root		B&P		Best heuristics	
	Gap	Time	Gap	Time	Gap	Time	Gap	Time
Set 1	8.06	0.63	0.81	1.37	0.70	44.36	1.34	373
Set 2	1.80	1.09	1.16	1.56	0.99	114.67	3.02	1,352
Set 3	6.66	3.46	5.46	5.71	4.85	1,527.86	9.28	3,854

gap compared to the initial heuristic. For the second and third set, there is less improvement, but the gaps are still much smaller compared to the Cattrysse et al. algorithm.

6. Conclusion and Future Research

In this paper, we present a MIP Dantzig-Wolfe reformulation for the capacitated lot-sizing problem with setup times. In this new formulation, the integer setup and the continuous production quantity decisions are separated. As such, we overcome the deficiency of the Dantzig-Wolfe reformulation proposed by Manne (1958), which can only be used to calculate a lower bound. We also describe an implementation of a branch-and-price algorithm. A combination of simplex optimization and subgradient updating is used to speed up the column generation process. Computational results show that B&P provides good results for the CLST. A limited comparison suggests that it is competitive with a state-of-the-art branch-and-cut system. Further, it is superior to other optimal procedures such as the network reformulation approach and to heuristics such as a customized tabu search algorithm.

Extensions of this research can focus on both the formulation and the algorithm. First, many extensions of the lot-sizing problem have been proposed, such as the backlog case and multilevel production. It would be interesting to adapt our approach for these extensions. For multilevel lot sizing, reformulations with echelon stock have been used to decompose the problem per level. Second, the combination of simplex optimization and subgradient updating, which is used to speed up the column generation process, could be useful for other formulations such as the generalized assignment problem or the simple plant location problem. Further computational improvements could be obtained by combined column and row generation (Vanderbeck 1998). Third, in the CLST, the continuous variable appears in both the subproblem and master. As a result, an extreme point of the subproblem is not necessarily an extreme point of the overall problem. This is precisely the deficiency in Manne's formulation. Vanderbeck and Savelsbergh (2006) discuss a general theoretical framework for Dantzig-Wolfe decomposition for MIPs. Our paper complements this general framework with an analysis of a specific MIP problem, namely, the capacitated lot-sizing problem. It would be interesting to investigate whether the ideas presented here can be transferred to the decomposition of other MIP problems.

Appendix A. Proof of Proposition 1

The proof consists of two parts:

(1) If (x_i^1, s_i^1, y_i^1) is an extreme point of $\text{conv}(X^i)$, then (x_i^1, s_i^1, y_i^1) has the Wagner-Whitin property and $(x_i^1, s_i^1, y_i^1) \in \{(x_i, s_i, y_i) \mid (x_i, s_i, y_i) \in X^i;\ \ s_{i,t-1}x_{it} = 0\ \forall t \in T\}$. See Eppen and Martin (1987), Proposition 2.4, for the general lot-sizing case.

(2) If

$$(x_i^1, s_i^1, y_i^1) \in \{(x_i, s_i, y_i) \mid (x_i, s_i, y_i) \in X^i; s_{i,t-1}x_{it} = 0\ \forall t \in T\},$$

i.e., if it has the Wagner-Whitin property, then it is an extreme point of $\text{conv}(X^i)$.

Following Definition 4.1 (Nemhauser and Wolsey 1988, p. 93), (x_i^1, s_i^1, y_i^1) is an extreme point of $\text{conv}(X^i)$ if there do not exist two points $(x_i^2, s_i^2, y_i^2), (x_i^3, s_i^3, y_i^3) \in \text{conv}(X^i)$, $(x_i^2, s_i^2, y_i^2) \neq (x_i^3, s_i^3, y_i^3)$, such that $(x_i^1, s_i^1, y_i^1) = (1/2) \cdot (x_i^2, s_i^2, y_i^2) + (1/2)(x_i^3, s_i^3, y_i^3)$.

Given a point $(x_i^1, s_i^1, y_i^1) \in \{(x_i, s_i, y_i) \mid (x_i, s_i, y_i) \in X^i; s_{i,t-1}x_{it} = 0\ \forall t \in T\}$, suppose that there do exist two different points $(x_i^2, s_i^2, y_i^2), (x_i^3, s_i^3, y_i^3) \in \text{conv}(X^i)$, such that $(x_i^1, s_i^1, y_i^1) = (1/2)(x_i^2, s_i^2, y_i^2) + (1/2)(x_i^3, s_i^3, y_i^3)$. We will prove that then $(x_i^2, s_i^2, y_i^2) = (x_i^3, s_i^3, y_i^3)$.

Because $y_i^1 \in \{0, 1\}$, $y_i^2, y_i^3 \leqslant 1$, $y_i^2, y_i^3 \geqslant 0$, and $y_{it}^1 = (1/2)y_{it}^2 + (1/2)y_{it}^3$, it follows that $y_i^1 = y_i^2 = y_i^3$.

Assume that there are exactly k periods with a strictly positive production ($0 \leqslant k \leqslant m$) in (x_i^1, s_i^1, y_i^1). Define l_v as the index of the vth period with strictly positive production. By definition, $x_{it}^1 = 0\ \forall t \in T\backslash\{l_1, \ldots, l_k\}$, and therefore $x_{it}^2 = x_{it}^3 = 0\ \forall t \in T\backslash\{l_1, \ldots, l_k\}$ because $x_{it}^1 = (1/2)x_{it}^2 + (1/2)x_{it}^3$ and $x_{it}^2, x_{it}^3 \geqslant 0$.

Because (x_i^1, s_i^1, y_i^1) satisfies the Wagner-Whitin property, it follows that $x_{il_v}^1 = \sum_{t=l_v}^{l_{v+1}-1} d_{it}$, $v \in \{1, \ldots, k-1\}$, and $x_{il_k}^1 = \sum_{t=l_k}^{m} d_{it}$. Further, because of the Wagner-Whitin property, we have $s_{i,l_v-1}^1 = 0$, $v \in \{1, \ldots, k\}$, which implies that $s_{i,l_v-1}^2 = s_{i,l_v-1}^3 = 0$, $v \in \{1, \ldots, k\}$, as $s_{it}^2, s_{it}^3 \geqslant 0$. Combined with the observation that $x_{it}^2 = x_{it}^3 = 0\ \forall t \in T\backslash\{l_1, \ldots, l_k\}$, this results in the conclusion that $x_{i,l_v}^2 = x_{i,l_v}^3 = \sum_{t=l_v}^{l_{v+1}-1} d_{it}$ for $v \in \{1, \ldots, k-1\}$ and $x_{ik}^2 = x_{ik}^3 = \sum_{t=l_k}^{m} d_{it}$.

We conclude that $(x_i^2, s_i^2, y_i^2) = (x_i^3, s_i^3, y_i^3)$, and hence we have proven that $(x_i^1, s_i^1, y_i^1) \in \{(x_i, s_i, y_i) \mid (x_i, s_i, y_i) \in X^i;\ \ s_{i,t-1}x_{it} = 0\ \forall t \in T\}$ cannot be written as the sum of two different feasible points, and consequently, it is an extreme point. □

Appendix B. Proof of Proposition $\bar{v}_{\text{DWCL}} = \bar{v}_M$

PROPOSITION. $\bar{v}_{\text{DWCL}} = \bar{v}_M$.

PROOF. A variable z_{qv}^i: $q \neq v$ is dominated by z_{vv}^i in the LP relaxation of (20)–(25). Remember that the variable z_{qv}^i refers to the extreme point formed by taking setup schedule $q \in Q^i$ and the Wagner-Whitin production plan according to setup schedule $v \in Q^{iq}$. Both variables z_{qv}^i and z_{vv}^i have the same production quantities x_{it}^v, but a different setup schedule: y_i^q for z_{qv}^i and y_i^v for z_{vv}^i. According to the definition of Q_{iq}, schedule $v \in Q_{iq}$ has strictly fewer setups than schedule q if $q \neq v$. As a consequence, the total setup cost for z_{vv}^i ($\sum_{t\in T} sc_{it}y_{it}^v$) is lower than or equal to the total setup cost for z_{qv}^i ($\sum_{t\in T} sc_{it}y_{it}^q$), while the production and inventory costs are equal. Further, the variable z_{vv}^i has an equal or lower capacity utilization ($st_{it}y_{it}^v + vt_{it}x_{it}^v\ \forall t \in T$) compared to any other z_{qv}^i ($st_{it}y_{it}^q + vt_{it}x_{it}^v\ \forall t \in T$). Consequently, there

exists an optimal LP solution with $z^i_{qv} = 0 \ \forall i \in P, \forall q \in Q_i, \forall v \in Q_{iq}: q \neq v$. The only variables left are the z^i_{vv} variables, which are equivalent to the z_{iv} variables in formulation (9)–(12). Consequently, the LP relaxation of (20)–(25) and Manne's formulation (9)–(12) are equivalent and have the same optimal objective value. □

Acknowledgments

The authors thank John O. McClain for making his code and data sets available and Dirk Cattrysse for making his data test sets and results available. This research was partially supported by the Fund for Scientific Research (F.W.O.)–Flanders, Belgium, and the Erasmus Research Institute of Management (ERIM), The Netherlands. The authors also thank the anonymous referees for their comments, and one of them specifically for providing guidelines to better explain the authors' approach and for suggesting a short proof for Proposition 2.

References

Barany, I., T. J. Van Roy, L. A. Wolsey. 1984. Strong formulations for multi-item capacitated lot sizing. *Management Sci.* **30**(10) 1255–1261.

Barnhart, C., E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, P. H. Vance. 1998. Branch-and-price: Column generation for solving huge integer programs. *Oper. Res.* **46**(3) 316–329.

Belvaux, G., L. A. Wolsey. 2000. BC-PROD: A specialized branch-and-cut system for lot-sizing problems. *Management Sci.* **46**(5) 724–738.

Belvaux, G., L. A. Wolsey. 2001. Modelling practical lot-sizing problems as mixed integer programs. *Management Sci.* **47**(7) 993–1007.

Bitran, G. R., H. Matsuo. 1986. The multi-item capacitated lot size problem: Error bounds of Manne's formulations. *Management Sci.* **32**(3) 350–359.

Cattrysse, D., J. Maes, L. N. Van Wassenhove. 1990. Set partitioning and column generation heuristics for capacitated dynamic lotsizing. *Eur. J. Oper. Res.* **46** 38–47.

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

Degraeve, Z., R. Jans. 2003. A new Dantzig-Wolfe reformulation and branch-and-price algorithm for the capacitated lot sizing problem with set up times. ERIM Report Series ERS-2003-010-LIS, Erasmus University, Rotterdam, The Netherlands.

Degraeve, Z., M. Peeters. 2003. Optimal integer solutions to industrial cutting stock problems: Part 2, Benchmark results. *INFORMS J. Comput.* **15**(1) 58–81.

Dzielinski, B. P., R. E. Gomory. 1965. Optimal programming of lot sizes, inventory, and labor allocations. *Management Sci.* **11**(9) 874–890.

Eppen, G. D., R. K. Martin. 1987. Solving multi-item capacitated lot-sizing problems using variable redefinition. *Oper. Res.* **35**(6) 832–848.

Florian, M., M. Klein. 1971. Deterministic production planning with concave costs and capacity constraints. *Management Sci.* **18**(1) 12–20.

Geoffrion, A. M. 1974. Lagrangean relaxation for integer programming. *Math. Programming Stud.* **2** 82–113.

Gopalakrishnan, M., K. Ding, J.-M. Bourjolly, S. Mohan. 2001. A tabu-search heuristic for the capacitated lot-sizing problem with set-up carryover. *Management Sci.* **47**(6) 851–863.

Huisman, D., R. Jans, M. Peeters, A. P. M. Wagelmans. 2005. Combining column generation and Lagrangian relexation. G. Desaulniers, J. Desrosiers, M. Solomon, eds. *Column Generation*. Springer, New York, 247–270.

Jans, R., Z. Degraeve. 2004. Improved lower bounds for the capacitated lot sizing problem with setup times. *Oper. Res. Lett.* **32** 185–195.

Jans, R., Z. Degraeve. 2007. Meta-heuristics for dynamic lot-sizing: A review and comparison of solution approaches. *Eur. J. Oper. Res.* **177**(3) 1855-1875.

Kleindorfer, P. R., E. F. P. Newson. 1975. A lower bounding structure for lot-size scheduling problems. *Oper. Res.* **23**(2) 299–311.

Lambrecht, M., H. Vanderveken. 1979. Heuristic procedures for the single operation, multi-item loading problem. *AIIE Trans.* **11**(4) 319–326.

Lasdon, L. 1970. *Optimization Theory for Large Systems*. Macmillan Company, New York.

Leung, J. M. Y., T. L. Magnanti, R. Vachani. 1989. Facets and algorithms for capacitated lot sizing. *Math. Programming* **45** 331–359.

Manne, A. S. 1958. Programming of economic lot sizes. *Management Sci.* **4**(2) 115–135.

Martin, K. 1987. Generating alternative mixed-integer programming models using variable redefinition. *Oper. Res.* **35**(6) 820–831.

Miller, A. J., G. L. Nemhauser, M. W. P. Savelsbergh. 2000. On the capacitated lot-sizing and continuous 0-1 knapsack polyhedra. *Eur. J. Oper. Res.* **125** 298–315.

Nemhauser, G. L., L. A. Wolsey. 1988. *Integer and Combinatorial Optimization*. John Wiley & Sons, New York.

Pochet, Y. 1988. Valid inequalities and separation for capacitated economic lot sizing. *Oper. Res. Lett.* **7**(3) 109–115.

Schrage, L. 1995. *LINDO: Optimization Software for Linear Programming*. Lindo Systems Inc., Chicago, IL.

Thizy, J. M., L. N. Van Wassenhove. 1985. Lagrangean relaxation for the multi-item capacitated lot-sizing problem: A heuristic implementation. *IIE Trans.* **17**(4) 308–313.

Trigeiro, W., L. J. Thomas, J. O. McClain. 1989. Capacitated lot sizing with set-up times. *Management Sci.* **35**(3) 353–366.

Vanderbeck, F. 1998. Lot-sizing with start-up times. *Management Sci.* **44**(10) 1409–1425.

Vanderbeck, F. 2000. On Dantzig-Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. *Oper. Res.* **48**(1) 111–128.

Vanderbeck, F., M. W. P. Savelsbergh. 2006. A generic view of Dantzig-Wolfe decomposition in mixed integer programming. *Oper. Res. Lett.* **34** 296–306.

Wagner, H. M., T. M. Whitin. 1958. Dynamic version of the economic lot size model. *Management Sci.* **5**(1) 89–96.