

Integer Programming Solution Approach for Inventory-Production-Distribution Problems with Direct Shipments

Miguel A. Lejeune, François Margot
LeBow College of Business, Drexel University, PA, USA
mlejeune@drexel.edu, fmargot@andrew.cmu.edu

October 18, 2007

Abstract

We construct an integrated multi-period inventory-production-distribution replenishment plan for three-stage supply chains. The supply chain maintains close-relationships with a small group of suppliers, and the nature of the products (bulk, chemical, etc.) makes it more economical to rely upon a direct shipment, full-truck load distribution policy between supply chain nodes. In this paper, we formulate the problem as an integer linear program that proves challenging to solve due to the general integer variables associated with the distribution requirements. We propose new families of valid cover inequalities, and we derive a practical closed-form expression for generating them, upon the determination of a single parameter. We study their performances through benchmarking several branch-and-bound and branch-and-cut approaches. Computational testing is done using a large-scale planning problem faced by a North American company.

Keywords: cover inequality, binary-integer knapsack constraint, production-distribution planning, supply chain management

1 Introduction

In this paper, we consider a general integer linear program with binary and general integer variables that is of the form:

$$\begin{aligned} & \min c^T x \\ & \text{subject to } Ax \geq b \\ & \quad x \geq 0 \\ & \quad x_j \in \mathcal{R}_+ \quad \text{for } j \in V_R \\ & \quad x_j \in \mathcal{Z}_+ \quad \text{for } j \in V_I \\ & \quad x_j \in \mathcal{B} \quad \text{for } j \in V_{0-1} \end{aligned} \tag{1}$$

where A is an $m \times n$ matrix, b is m -dimensional while c and x are n -dimensional vectors. We use \mathcal{R}_+ , \mathcal{Z}_+ and \mathcal{B} to denote the set of positive integer, positive real, and binary (taking values 0 or 1) variables, respectively. The index set $\{1, \dots, n\}$ is partitioned into the three disjoint subsets V_R, V_I, V_{0-1} standing for the index set of real positive, integer positive, and binary (zero-one) variables respectively.

Cutting planes and valid inequalities for integer programming problems [13] are more commonly derived for 0-1 variables. Although considered in, inter alia, [3, 6, 20, 25, 26], mixed integer linear programs (*MILP*) comprising general integer variables have received less attention. Therefore, they are often solved solely relying on branch-and-bound or branch-and-cut algorithms embedded in commercial solvers, with mixed results. However, MILP with general integer variables are very common in industrial applications.

In this paper, we derive strong valid cover inequalities for the convex hull of integer solutions of the following set that contains several binary variables and one integer variable:

$$X = \{(x, y) \in \{0, 1\}^n \times \mathcal{Z}_+ : \sum_{j=1}^n a_j x_j + a_y y \leq b, 0 \leq y \leq M\} .$$

The study of this problem was motivated by the construction of a multi-period replenishment plan for a multi-stage chemical supply chain described in the next section. The construction of integrated multi-functional planning and scheduling problem is well-known to be a very complex task (see *inter alia* [8, 18, 28, 29]) due to the presence of symmetry, and, in particular, the combined presence of large continuous variables (amounts supplied, inventory levels) and discrete variables (choice of carriers, routes, maintenance schedule). Clearly, research pertaining to the joint modeling and development of solution techniques of such problems is needed [9, 14]; this study contributes to fill in this gap.

A very similar problem was handled by one of the 2006 Franz Edelman Award Competition finalists [14]. The objective was to optimize the operational costs of Omya Hustadmarmor, a Norwegian company that supplies calcium carbonate to European paper manufacturers. The similarity is striking with our problem since the same product, i.e., calcium, was also transported from a single processing plant using tank ships of various sizes and incurring various levels of fixed costs. The project resulted in the provision of a decision support system involving the solution of a maritime inventory-routing problem. The cover inequalities proposed in this paper could be implemented in such a framework and would be very useful in solving the combinatorial inventory-distribution problem. Other relevant studies of the integrated production-distribution problem can be found in [7, 9, 15, 17]. We refer the reader to [8, 18, 28, 29] for thorough reviews of the literature on the production-distribution networks, and to [10] for a focused review of such planning problems with maritime distribution. Finally, we note that other real-life industrial problems (see [6]), such as electricity generation and production, network design when capacity can be installed by batches, models supporting the decision to construct a new facility and to determine its capacity involve the analysis of a set similar to the set X defined above.

We propose a new closed-form expression to derive valid cover inequalities for the knapsack constraints described above. This expression depends on a single parameter, so that the associated cuts can be obtained directly, without having to implement a recursive generating procedure. The valid inequalities derived in this paper for the type of knapsack constraints in X are thereafter referred to as *binary-integer cover inequalities*. The advantage of these cuts over other families of cuts for the specific application problem is that they are easy to generate and seem to provide a bigger lower bound improvement.

The remaining part of the paper is organized as follows. In Section 2, the problem motivating our study is presented. In Section 3, the concept of a *binary-integer cover* is introduced. It is then used in Section 4 to derive new families of valid binary-integer cover inequalities. We also provide a very simple and practical closed-form formulation, allowing for very fast generation. We then describe some of their properties such as sufficient conditions under which they are facet-defining. Comparisons with other classes of inequalities are given. In Section 5, we test the efficiency and practicality of the proposed cover inequalities for binary-integer knapsack constraints on instances of the industrial problem described in Section 2. The efficiency and practicality of our binary-integer cover inequalities are discussed with

respect to robustness, and to the reduction in the optimality gap and in the computing time. Section 6 provides concluding remarks.

2 Problem Description

In this section, we describe and formulate a discrete-time (i.e., finite number of periods), multi-functional (i.e, inventory, production and distribution) integrated planning and scheduling model for a three-stage supply chain. The study is motivated by the problem faced by a company which is one of the top five worldwide producers of soda ash and calcium chloride. It operates within a three-stage supply chain that comprises one supplier in Michigan, two production facilities in Michigan and Ontario, and 13 main distributors in harbor cities such as Montreal, Dartmouth, Cleveland, Oshawa, etc. The company has about \$300 million revenue, and one third of which comes from the calcium chloride market. The raw material is the brine and the end-products are several forms (liquid, flake, pellet) of calcium chloride. The demand for calcium chloride is non-stationary and very seasonal. Very large and heterogeneous (i.e., differing in loading capacity and speed) tank ships, vessels or barges are used to transport products over the Great Lakes between supply chain nodes. Large vessels are very expensive to operate; transportation costs represent about 50% of the total product costs. Contracting out transportation capacity with external logistics service providers is also an option. Transportation scheduling is critical for the supply chain, and the lack of proper planning tools leads to both unpredictability and limited responsiveness. The supply chain distribution policy is based on full-load, direct shipments (i.e. a single loading and discharging location) between supply chain nodes. The rationale for this comes from the facts that the demand is much larger than the vessels maximal loading capacity, and that the costs of setting up and operating a vessel is very high. Such a distribution policy is common for the transportation of bulk products [14], in particular, when a maritime distribution fleet is employed. The inventory-production-distribution schedule we construct determines:

- the production schedule at each supply chain facility and at each period;
- the inventory levels at each supply chain node and period;
- the supplied quantities at each supply chain node and period;
- the scheduling and routing scheme for each vessel, i.e. the number of shipments between supplier and manufacturer facilities and between manufacturer and distributor facilities at each period and with each carrier;
- the maintenance schedule for each carrier.

We provide below the formulation of the model. The following set notations will be used in the sequel: T is the set of periods in the planning horizon, V is the set of transportation carriers, I is the set of production facilities (suppliers and producers), J is the set of distribution centers, and $K = I \cup J$ is the set of supply chain nodes. For ease of exposition, we omit the conversion factors for the transformation of raw materials into semi-finished products and of these latter into end-products.

$$\min \quad \underbrace{\sum_{i \in I} \sum_{t \in T} (b[i, t] \cdot w[i, t])}_{\text{Production cost}} + \underbrace{\sum_{k \in K} \sum_{t \in T} (h[k, t] \cdot z[k, t])}_{\text{Inventory cost}} \quad (2)$$

$$+ \underbrace{\sum_{i \in I} \sum_{t \in T} \sum_{k \in K, k \neq i} \sum_{v \in V} ((l[i, v] + 2 \cdot r[i, k, v] + u[k, v]) \cdot c[v] \cdot x[i, k, v, t])}_{\text{Distribution cost}} \quad (3)$$

such that

$$z[k, t] = z[k, t - 1] + \sum_{i \in I} \sum_{v \in V} q[i, k, v, t] - d[k, t], \quad k \in K, t \in T \quad (4)$$

$$z[k, t] \leq z^{\max}[k], \quad k \in K, t \in T \quad (5)$$

$$w[i, t] \leq w^{\max}[i, t], \quad i \in I, t \in T \quad (6)$$

$$q[i, k, v, t] = e[v] \cdot x[i, k, v, t], \quad i \in I, k \in K, v \in V, t \in T \quad (7)$$

$$\sum_{i \in I} \sum_{k \in K} (l[i, v] + 2 \cdot r[i, k, v] + u[k, v]) \cdot x[i, k, v, t] \leq a[v, t], \quad v \in V, t \in T \quad (8)$$

$$x[i, k, v, t] \leq M \cdot \delta[v, t], \quad v \in V, t \in T \quad (9)$$

$$\sum_{t \in T} \delta[v, t] \leq |T| - 1, \quad v \in V \quad (10)$$

$$z[k, \tau] \geq z[k, 0], \quad k \in K \quad (11)$$

$$x[i, k, v, t] = 0, \quad i \in I, k \in K(t), v \in V, t \in T \quad (12)$$

$$\delta[v, t] \in \{0, 1\}, \quad v \in V, t \in T \quad (13)$$

$$x[i, k, v, t] \in \mathcal{Z}_+, \quad i \in I, k \in K, v \in V, t \in T \quad (14)$$

$$z[k, t], w[i, t], q[i, k, v, t] \geq 0, \quad i \in I, j \in J, t \in T, \quad (15)$$

$$v \in V, k \in K$$

The objective function (2)-(3) is linear and represents the sum of the distribution, production and holding costs incurred by the supply chain. The production costs are computed as the product of the production level $w[i, t]$ at facility i and time t by the unit production cost $b[i, t]$. The inventory costs are computed as the product of the ending inventory level $z[k, t]$ at node k and time t by the unit inventory costs $h[k, t]$. The distribution costs are a function of the lead time for delivering products to node k using a carrier v departing from node i . More precisely, the distribution costs are computed by multiplying the product of the shipping cost $c[v]$ (per unit of lead time) associated with v by the lead time $(l[i, v] + 2 \cdot r[i, k, v] + u[k, v])$ to deliver products with v from i to k by the number of shipments $x[i, k, v, t]$ carried out with carrier v between i and k at t . We define the lead time as the sum of the loading time $l[i, v]$ of v at i , the unloading time $u[k, v]$ at node k and the transportation time $r[i, k, v]$ for using v to deliver from i to k . We multiply $r[i, k, v]$ by 2 to account for the round-trip transportation time.

Denoting by $q[i, k, v, t]$ the amount of products delivered to k at t with carrier v leaving from i , the *product flow balance* constraints (4) set the ending inventory level $z[k, t]$ at a period t and node k equal to the ending inventory level at $t - 1$ plus the supply received during period t minus the customers' demands $d[j, t]$ satisfied by the end of t . Combined with the non-negativity constraint on the inventory level $z[j, t]$ (15), constraints (4) enforce a no backlog policy. The backlogging of the unmet demand is not tolerated by demanding customers, or those suffering from high set up costs when there is a shortage [14]. The capacitated inventory level constraints (5) ensure that the inventory level does not exceed the maximum inventory capacity z^{\max} . Similarly, the capacitated production constraints (6) impose an upper bound

$w^{max}[i, t]$ on the production level at production facility k at each period t . The *transportation capacity* constraints, which set the quantity $q[i, k, v, t]$ delivered from i to k at t with ship v equal to the number of shipments $x[i, k, v, t]$ multiplied by the maximal loading capacity $e[v]$ of ship v , account for the limited loading capacity of the carriers, and enforce the full-load shipment policy. Denoting by \mathcal{Z}_+ the set of positive integer numbers, constraints (14) represent the *shipment indivisibility* requirement. The constraints (8) represent the *limited time availability* $a[v, t]$ of carriers, and account for the lead time for a shipment made with v between i and k . Denoting by $|T|$ the cardinality of set T , by M the maximum number of shipments between i and k that can be done at t with v , and by $\delta[v, t]$ a binary variable (13) equal to 1 if the carrier v is set up at time t and equal to 0 otherwise, the requirements to allow for the *maintenance of the carriers* are modeled with constraints (9) and (10). These guarantee that each carrier is not used in at least one period, allowing for its maintenance. The *sustainability* constraints (11) prevent *end-effect* issues (i.e. what is optimal over the short horizon may be suboptimal over the long run [16]), and make the plan reproducible over the next planning horizon by ensuring that inventory levels at the last period τ of the planning horizon are at least equal to the initial ones. Denoting by $K(t)$ the set of supply chain nodes that are not accessible at time t , constraints (12) make the distribution scheme *operational*, accounting for the distribution *time-window constraints* (12) due to the impossibility to supply j at period t due to bad weather conditions, the closing of facilities. Constraints (15) are the non-negativity constraints.

A large part of the difficulty in the construction of the inventory-production-distribution plan stems from the distribution constraints (8), which are knapsack constraints containing general integer variables $x[i, k, v, t]$. These latter represent the number of shipments between nodes of the supply chain at a certain time-period and with a specific carrier. A very high fixed cost is associated with every such shipment; it is therefore crucial to determine the optimal number of such shipments, which makes the handling of the associated distribution constraints so important.

In the studied real world application, it was noticed that the number of supplier-to-manufacturer shipments is very large, possibly occurring many times per period, while the number of manufacturer-to-distributor shipments is much lower (0 or 1). Based on this observation, we replace the constraints (8) for each carrier v associated with the production facility i as follows:

$$\alpha[i, i', v] y[i, i', v, t] + \sum_{j \in J} \alpha[i, j, v] x[i, j, v, t] \leq a[v, t], \quad i, i' \in I, v \in V, t \in T, \quad (16)$$

$$y[i, i', v, t] \in \mathcal{Z}_+, \quad x[i, j, v, t] \in \{0, 1\}, \quad i \in I, j \in J, v \in V, t \in T,$$

where $y[i, i', v, t]$ is the number of shipments carried out with carrier v leaving from supplier i and heading to manufacturer i' in period t , and $x[i, j, v, t]$ is defined as a binary number representing the number of shipments carried out with carrier v leaving from manufacturer i and heading to distributor j at time t , and $\alpha[i, k, v]$ is the lead time for a shipment between nodes i and k with v .

The distribution constraints (16) are highly combinatorial and exacerbate the difficulty of solving the large mixed-integer programming problem at hand. These are $\{(x, y) \in B^n \times \mathcal{Z}_+\}$ -knapsack constraints containing many binary integer variables (manufacturer-to-distributors shipments) and one single general integer variable (supplier-to-manufacturer shipments).

It is also very important to note that in some of the methods [6] described in Section 4.5 the coefficient of the general integer variable is assumed to be larger than these of the binary variables. Such an assumption does not hold in distribution problems. Indeed, it can be observed that manufacturers are often located very close from the suppliers facilitating the numerous shipments between each other, while the distance separating manufacturers from distributors are generally more important. It follows from this that the coefficients of the general integer variables are generally, and must be allowed being, smaller than those of the binary integer variables.

3 Binary-Integer Cover

Our aim is to generalize the concept of a cover inequality for a 0-1 knapsack constraint to a knapsack constraint with 0-1 variables and a single general integer variable. In this case, the values assumed by the general integer variable must also be taken into account in the construction of the cover.

We consider the binary-integer knapsack set

$$X = \{(x, y) \in \{0, 1\}^n \times \mathcal{Z}_+ : \sum_{j=1}^n a_j x_j + a_y y \leq b, 0 \leq y \leq M \leq \left\lfloor \frac{b}{a_y} \right\rfloor, a_j \leq b, j = 1, \dots, n\}. \quad (17)$$

Definition 3.1 For $W \subseteq \{1, \dots, n\}$ and $K \in \{0, 1, \dots, M\}$ we say that:

- the pair (W, K) is a cover if

$$\sum_{j \in W} a_j + K a_y = b + \lambda, \quad \lambda > 0; \quad (18)$$

- the cover (W, K) is a minimal cover if

$$\sum_{j \in W} a_j + (K - 1) a_y \leq b, \quad (19)$$

and, for all $i \in W$,

$$\sum_{j \in W \setminus \{i\}} a_j + K a_y \leq b. \quad (20)$$

In other words, (W, K) is a cover if the point with coordinates $x_j = 1, j \in W, x_j = 0, j \notin W$, and $y = K$ is infeasible. The cover is minimal, if decreasing by 1 any of the positive values of the coordinates of the point makes it feasible. The concept of a binary-integer cover is an extension of the concept of cover; indeed, in case of $M = 1$, the definition above reduces to the cover definition of 0-1 knapsack constraints.

Inequalities (19) and (20) imply that:

$$\lambda \in]0, \min\{\min_{j \in W} a_j, a_y\}]. \quad (21)$$

A simple way to construct valid inequalities for the set X defined in (17) is to binarize the general integer variable y : $y = \sum_{k=1}^M y_k$, where $y_k \in \{0, 1\}, k = 1, \dots, M$. By doing so, we transform the binary-integer knapsack constraint into a 0-1 knapsack constraint:

$$X = \{(x, y) \in \{0, 1\}^{n+M} : \sum_{j=1}^n a_j x_j + a_y \sum_{k=1}^M y_k \leq b\}. \quad (22)$$

However, due to the massive symmetry introduced in that way, the reformulation of the general integer variables as a sum of binary ones is known to lead to problems which are difficult to solve [26, 27].

Another reformulation of the general integer variable y into a sum of binary variables is given by:

$$y = y_0 + 2y_1 + 4y_2 + \dots + 2^r y_r$$

with $y_k \in \{0, 1\}$, $k = 1, \dots, r$, $r = \lfloor \log_2 M \rfloor$.

Here also, the reformulation leads to difficult problems [26, 27].

4 Binary-Integer Cover Inequalities

Let (W, K) be a minimal cover for the set X defined in (17). We are interested in generating valid inequalities for X of the form

$$\sum_{j \in W} x_j + sy \leq t \quad (23)$$

where s and t are arbitrary numbers. To generate the strongest inequalities of this form, project X onto the 2-dimensional space $\{(y, w) \mid w = \sum_{j \in W} x_j\}$. A typical graphical representation of such a set is given in Figure 1.

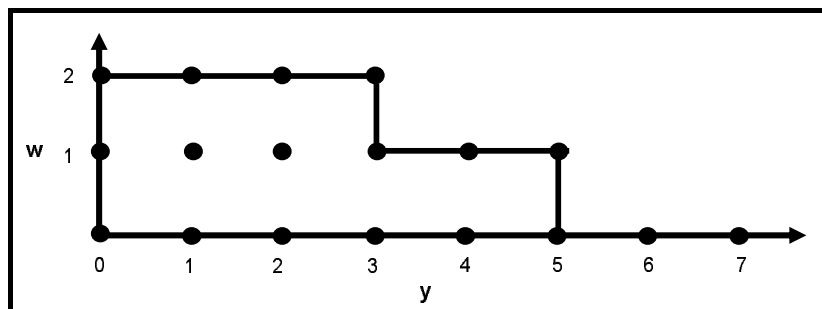


Figure 1: Projection of X onto a 2-dimensional space.

The projection and its convex hull can easily be computed in time linear in $|W|$ and K , if we assume that the variables x_j are already sorted according to non decreasing value of a_j . Any inequality defining a facet of the convex hull of the form $\alpha w + \beta y \leq \gamma$ with $\alpha \neq 0$ gives the inequality of the form (23)

$$\sum_{j \in W} x_j + \frac{\beta}{\alpha} y \leq \frac{\gamma}{\alpha} \quad (24)$$

We call these inequalities *Convex Hull inequalities*. While generating these inequalities is relatively straightforward, we give below simpler ways to generate (possibly weaker) inequalities in the family (23). Our justification for doing so lies in the results of the computational experiments described in Section 5.

We first observe that, in many cases, the following inequality, directly generalizing the binary knapsack case, holds true:

$$\sum_{j \in W} x_j + y \leq |W| + K - 1. \quad (25)$$

If (i) $y \leq K$ or (ii) if $y > K$ with $a_y \geq \max_{j \in W} a_j$, then inequality (25) is valid. However, if $y > K$ and $a_y < \max_{j \in W} a_j$, inequality (25) may be violated by some feasible solutions.

Below, we propose new families of binary-integer cover inequalities that are valid for each case mentioned above. We derive a closed-form formulation for these inequalities, which are extremely easy to generate, and discuss their properties.

4.1 Family I of Binary-Integer Cover Inequalities

Theorem 4.1 *If (W, K) is a minimal cover, then the binary-integer cover inequality*

$$\gamma \sum_{j \in W} x_j + y \leq \gamma |W| + K - 1 \quad (26)$$

with

$$\gamma = \left\lceil \frac{\max_{j \in W} a_j}{a_y} \right\rceil \quad (27)$$

is a valid inequality for X .

Proof. Since (W, K) is a cover, equality (18) holds true. Subtracting from it the knapsack inequality

$$\sum_{j \in W} a_j x_j + a_y y \leq b$$

we obtain:

$$\sum_{j \in W} a_j (1 - x_j) \geq \lambda + (y - K) a_y . \quad (28)$$

Let γ be the smallest integer greater than or equal to 1 such that

$$\max_{j \in W} a_j \leq \gamma a_y .$$

Then

$$\sum_{j \in W} a_j (1 - x_j) \leq \left(\max_{j \in W} a_j \right) \sum_{j \in W} (1 - x_j) \leq \gamma a_y \sum_{j \in W} (1 - x_j) .$$

This combined with (28) implies

$$\gamma a_y \sum_{j \in W} (1 - x_j) \geq \lambda + (y - K) a_y ,$$

that is equivalent to (26) since $0 < \lambda/a_y < 1$. Since γ is integer, we conclude that

$$\gamma \sum_{j \in W} x_j + y \leq \gamma |W| + K - 1 ,$$

which was set out to prove. □

4.2 Properties

Let (W, K) be a minimal cover. We define X^W to be the projection of X on the subspace of the variables $\{x_j \mid j \in W\} \cup y$. In this section, we define sufficient conditions under which the binary-integer cover inequality (26) is facet-defining for X^W , similarly to binary minimal cover inequalities. Lifting of binary-integer cover inequalities can be done similarly to lifting of binary cover inequalities.

Lemma 4.2 *Let (W, K) be a minimal cover. A sufficient condition for the binary-integer cover inequality (26) to be facet-defining for X^W is that $\frac{b}{a_y} \leq 1$.*

Proof. If $\frac{b}{a_y} = 1$, it is trivial to verify that the only minimal binary-integer cover (W, K) involving the general integer variable y is such that $|W| = 1$ and $K = 1$. The binary-integer cover inequality (26) reads $x_j + y \leq 1, j \in W$, and is facet-defining for X^W . Otherwise, i.e. if $\frac{b}{a_y} = 1$ and $K = 0$ ((W, K) is a minimal cover involving binary variables only), or if $\frac{b}{a_y} < 1 (\Rightarrow y = 0)$, the binary-integer cover inequality is the minimal 0/1 cover inequality $\sum_{j \in W} x_j \leq |W| - 1$, which is well known to be facet-defining for X^W [4, 19]. \square

Proposition 4.3 *Let (W, K) be a minimal binary-integer cover. A sufficient condition for the binary-integer cover inequality (26) to be facet-defining for X^W is that $a_y \geq \max_{j \in W} a_j$.*

Proof. An alternative formulation for X^W is obtained through the binarization [26] of the general integer variable: $y = \sum_{k=1}^M y_k$ (22). The set X'' equivalent to X^W is thus

$$X'' = \{(x, y) \in \{0, 1\}^{|W|+M} : \sum_{j \in W} a_j x_j + \sum_{k=1}^M a_y y_k \leq b\}. \quad (29)$$

Let $C = W \cup C''$, where C'' contains exactly K of the M binary variables $\{y_k \mid k = 1, \dots, M\}$. Since $C = W \cup C''$ is a minimal cover for X'' ,

$$\sum_{j \in W} x_j + \sum_{k \in C''} y_k \leq |W| + |C''| - 1$$

is a valid facet-defining cover inequality [4, 19] for X'' and thus for X^W . Since $|C''| = K$, the inequality above reads

$$\sum_{j \in W} x_j + \sum_{k \in C''} y_k \leq |W| + K - 1, \quad (30)$$

with $\sum_{k \in C''} y_k \leq y = \sum_{k=1}^M y_k$. In that case, γ is equal to one in inequality (26) that has the same form as (30) and is thus facet-defining for X^W . \square

Proposition 4.4 *Let (W, K) be a minimal binary-integer cover. A sufficient condition for the binary-integer cover inequality (26) to be facet-defining for X^W is that $y \leq K$. Then, (26) becomes*

$$\sum_{j \in W} x_j + y \leq |W| + K - 1. \quad (31)$$

Proof. If $y = K$, then (26) becomes

$$\gamma \sum_{j \in W} x_j \leq \gamma |W| - 1,$$

and it is obvious from the definition of a minimal binary-integer cover that the inequality above is facet-defining for X^W : (18) and (20) show that, for (17) to be satisfied when $y = K$, at least one binary decision variable in W (anyone of those in W) takes value 0. So, clearly, $\gamma \sum_{j \in W} x_j \leq \gamma |W| - 1$ requires this, and cuts all integer solutions that are not feasible for (17) with $y = K$.

Otherwise, if $\Upsilon = K - y \geq 1$, then (26) becomes

$$\gamma \sum_{j \in W} x_j \leq \gamma |W| \leq \gamma |W| - 1 + \Upsilon.$$

The above inequality does not allow any integer solution not feasible for (17) when $y < K$. □

4.3 Family II of Binary-Integer Cover Inequalities

The strength of the valid binary-integer cover inequality (26) depends on the value of γ . If we can make γ smaller, then the inequality becomes stronger. The value of γ (27) can be decreased by making a_y larger or by making $\max_{j \in W} a_j$ smaller.

The first way to make (26) stronger consists of keeping a_y unchanged, and decreasing the value of some a_j , $j \in W$. We can decrease these coefficients to some values $\bar{a}_j \leq a_j$ as long as the pair (W, K) remains a cover for the new set

$$X' = \left\{ (x, y) \in \{0, 1\}^n \times \mathcal{Z}_+ : \sum_{j=1}^n \bar{a}_j x_j + a_y y \leq b, 0 \leq y \leq M \leq \left\lfloor \frac{b}{a_y} \right\rfloor \right\}.$$

Clearly, we should try to reduce the largest a_j first as it is the bottleneck in reducing γ (27).

In this section, we propose a new family of binary-integer cover inequalities that can strengthen inequality (26), and that is obtained by reducing the values of the coefficients a_j of the binary variables $x_j \in W$.

Theorem 4.5 *If (W, K) is a minimal cover, then the binary-integer cover inequality*

$$\mu \sum_{j \in W} x_j + y \leq \mu |W| + K - 1 \tag{32}$$

is valid for the binary-integer knapsack set X , provided that

$$\mu = \left\lceil \frac{\bar{a}}{a_y} \right\rceil, \tag{33}$$

and the value of \bar{a} is the optimal solution of the following linear program

$$\begin{aligned} \bar{a} = & \min a \\ & a \geq \bar{a}_j, j \in W \end{aligned} \tag{34}$$

$$\text{subject to } \sum_{j \in W} (a_j - \bar{a}_j) \leq \lambda - \epsilon, \tag{35}$$

$$a_j \geq \bar{a}_j, j \in W \tag{36}$$

where ϵ is an infinitesimal positive number and λ is defined in (18).

Problem (34) minimizes the largest coefficient (\bar{a}) of the binary variables in the cover W ; it decreases the coefficient value (from a_j to \bar{a}_j) as much as possible provided that the reduction in the values of the coefficients remains strictly smaller than the slack λ .

Proof. Since (W, K) is a cover, inequality (18) holds true, and we have (28):

$$\sum_{j \in W} a_j(1 - x_j) \geq \lambda + (y - K)a_y .$$

From (35), we have

$$\sum_{j \in W} a_j(1 - x_j) \geq \sum_{j \in W} (a_j - \bar{a}_j) + \epsilon + (y - K)a_y , \quad (37)$$

that can be equivalently formulated as:

$$\sum_{j \in W} \bar{a}_j - \sum_{j \in W} a_j x_j \geq \epsilon + (y - K)a_y .$$

Since $a_j \geq \bar{a}_j, j \in W$ (36), we have

$$\sum_{j \in W} \bar{a}_j(1 - x_j) \geq \epsilon + (y - K)a_y ,$$

which from (34) implies

$$\bar{a} \sum_{j \in W} (1 - x_j) \geq \epsilon + (y - K)a_y .$$

Since μ is the smallest positive integer such that: $\bar{a} \leq \mu a_y$, we obtain

$$\mu a_y \sum_{j \in W} (1 - x_j) \geq \epsilon + (y - K)a_y .$$

Therefore,

$$\mu(|W| - \sum_{j \in W} x_j) \geq \frac{\epsilon}{a_y} + (y - K) .$$

It follows that

$$\mu \sum_{j \in W} x_j + y \leq \mu|W| + K - 1 ,$$

which was set out to prove. □

As an example, we consider the binary-integer knapsack set X (38)

$$X = \{(x, y) \in \{0, 1\}^2 \times \mathcal{Z}_+ : 4x_1 + 5x_2 + 2y \leq 15\} , \quad (38)$$

and the minimal cover $(W, K) = (\{1, 2\}, 4)$, in which x_2 is the binary variable with the largest coefficient (5) and $\lambda = 2$. Let $\epsilon = 0.0001$, the solution of (34) returns $\bar{a}_1 = \bar{a}_2 = \bar{a} = 3.50005$. Therefore, $\mu = 2$, and the binary-integer cover inequality (32) is:

$$2x_1 + 2x_2 + y \leq 7 .$$

The inequality above cuts the infeasible solutions $x_1 = 1, x_2 = 0, y = 6$ and $x_1 = 0, x_2 = 1, y = 6$, which are not cut by the inequality

$$3x_1 + 3x_2 + y \leq 9$$

obtained using (26).

Considering the example introduced above, we represent below the feasible sets (Figure 2) determined with Family I ($3x_1 + 3x_2 + y \leq 9$) and Family II ($2x_1 + 2x_2 + y \leq 7$) binary-integer cover inequalities. Black dots represent the integer solutions feasible for both types of binary-integer cover inequalities, while white dots represent integer solutions cut off by the Family II-type inequality. The black dots on or below the bold black lines represent the feasible integer points for (38).

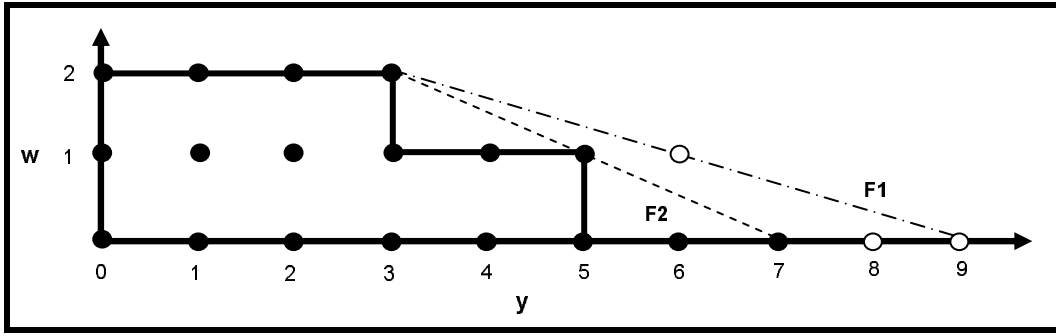


Figure 2: Feasible sets with Families I and II of Binary-Integer Cover Inequalities.

4.4 Extended Covers and Extended Cover Inequalities

In this section, an extension of the valid cover inequality (26) is proposed for the extended cover (W', K) defined below.

Definition 4.6 *The pair (W', K) is an extended cover if*

$$W' = W \cup \{j' : a_{j'} \geq a_j \text{ for all } j \in W, a_{j'} \geq a_y\} \quad \text{and} \quad (W, K) \text{ is a cover.}$$

Proposition 4.7 *If (W', K) is an extended cover, then the inequality*

$$\gamma' \sum_{j \in W'} x_j + y \leq \gamma' |W'| + K - \vartheta$$

is valid, with

$$\gamma' = \left\lceil \frac{\max_{j \in W'} a_j}{a_y} \right\rceil$$

and

$$\vartheta = \left\lceil \frac{\lambda}{a_y} \right\rceil.$$

Proof. The proof is very similar to that for obtaining (26). We consider the system:

$$\sum_{j \in W'} a_j x_j + a_y y \leq b, \quad (39)$$

$$\sum_{j \in W'} a_j + K a_y = b + \lambda, \quad \lambda > 0. \quad (40)$$

Subtracting (39) from (40) and defining γ' as the smallest integer such that

$$\max_{j \in W'} a_j \leq \gamma' a_y,$$

we obtain the following inequality:

$$\gamma' a_y \sum_{j \in W'} (1 - x_j) \geq \lambda + (y - K) a_y.$$

This can be equivalently transformed to

$$\gamma' \sum_{j \in W'} x_j + y \leq \gamma' |W'| + K - \frac{\lambda}{a_y}.$$

Since γ' is integer and $\lambda > 0$, we conclude that

$$\gamma' \sum_{j \in W'} x_j + y \leq \gamma' |W'| + K - \vartheta,$$

with $\vartheta = \left\lceil \frac{\lambda}{a_y} \right\rceil$. □

The extended cover is not necessarily a minimal cover, thus explaining why we cannot substitute 1 for λ/a_y in the proof.

4.5 Comparison with Other Valid Inequalities

In this section, we review several existing methods that can be employed to generate valid inequalities for the binary-integer knapsack set X as given in (17). Simple examples allow us to compare the strength and computational requirements of the proposed binary-integer cover inequalities with respect to the valid inequalities obtained by using disjunctive cuts (Owen and Mehrotra [25]), Chvátal-Gomory cuts [11], the mixed integer rounding inequalities (MIR) (Marchand and Wolsey [22]) and the methods of Ceria et al. [6], Atamtürk [2], Atamtürk and Rajan [3] and van Hoesel et al. [20]. The cover inequalities we propose compare very well with those methods, while their closed-form formulation and the simplicity of the algorithm to generate them are marked advantages. In Section 5.2, we compare empirically the proposed cuts with cuts of some of the families below by reporting results at the root node for the application considered in this paper.

It can be shown that the cover inequalities proposed here are not a subclass of rank one Chvátal-Gomory cuts.

4.5.1 Disjunctive Cuts

Another group of valid inequalities can be obtained using disjunctive arguments. If (W, K) is a minimal cover, the following disjunction applies:

$$\begin{aligned} \text{If } \sum_{j \in W} x_j = |W| \quad \text{then } y &\leq K - 1, \\ \text{If } \sum_{j \in W} x_j < |W| \quad \text{then } y &\leq K + \left\lfloor \frac{\sum_{j \in W} a_j(1 - x_j) - \lambda}{a_y} \right\rfloor. \end{aligned}$$

In the special case when $K = M$, inequality (25) is always valid. This is in the line of research of Owen and Mehrotra [25], whose disjunctive cutting planes are obtained from the generalization of the 0-1 disjunction proposed by Balas et al. [5]. Having to set $K = M$ limits the effectiveness of these cuts when M is large.

4.5.2 Valid Inequalities for Integer Programs with General Integer Variables

Ceria et al. [6] proposed a way for constructing valid inequalities for knapsack constraints containing general integer variables. In particular, their work applies in our case as the bounds on the integer variables can be chosen equal to 1. Their valid inequalities are not proved to be facet-defining, and are obtained by fixing the integer variables at their upper bounds. Atamtürk [2] extends the work of [6] and rewrites their cover inequality for the integer knapsack set. The new formulation states that at least one of the general integer variables included in the cover cannot be equal to its upper bound. Recently, a procedure to derive strong valid inequalities for the mixed-integer knapsack set containing two general integer variables was proposed [1].

It can be shown that the cover inequalities proposed here are not a subclass of these inequalities.

4.5.3 MIR Inequalities

The MIR inequalities are a variant of the Gomory mixed integer cuts. Considering the mixed integer set

$$X = \left\{ (x, y) \in \mathcal{R}_+^2 \times \mathcal{Z}_+^n : \sum_{j=1}^n a_j y_j + x^+ \leq b + x^- \right\}$$

and setting $f = b - \lfloor b \rfloor$ and $f_j = a_j - \lfloor a_j \rfloor$, $j = 1, \dots, n$, the mixed integer rounding inequality

$$\sum_{j=1}^n \left(\lfloor a_j \rfloor + \frac{(f_j - f)^+}{1 - f} \right) y_j \leq \lfloor b \rfloor + \frac{x^-}{1 - f}$$

can be shown to be valid for the above set X .

It can be shown that the cover inequalities proposed here are not a subset of the MIR inequalities which are, on the other side, a generalization of the inequalities [6] described above.

Marchand and Wolsey [22] define *c-MIR* inequalities for a MIP by aggregating inequalities of the original formulation, bound substitutions, and scaling before applying the above formula. These operations are done in a heuristic way. We compare empirically the binary-integer cover inequalities with *c-MIR* inequalities in Section 5.3.

4.5.4 Relaxation-Based Method

In addition to the two constructive methods described above, a much more computationally intensive method allowing the derivation of valid inequalities for similar knapsack constraints has been proposed in [3] and [20]. It was developed for the analysis of unsplittable single arc-set relaxations for the multi-commodity flow capacitated network design. The generation of these inequalities requires the solution of an auxiliary large-scale integer programming problems. Such a procedure could turn out to be moderately practical but likely to be very resource-consuming, in particular for large binary-integer knapsack constraints with hundreds of binary variables.

5 Application

5.1 Test Laboratory

In this section, we test the efficiency and contribution of eight solution methods based on the developments described above, and compare them with the branch-and-cut algorithm implemented in CPLEX 10.0. The eight methods will thereafter be referred to as:

- *BB1* (*BB2*, *BB12*, *BBCH*, respectively) is a branch-and-bound algorithm in which all Family I (Family II, Family I and II, Convex Hull, respectively) inequalities are introduced at the root node in the formulation of the original problem;
- *BC1* (*BC2*, *BC12*, *BCCH*, respectively) is a branch-and-cut algorithm in which Family I (Family II, Family I and II, Convex Hull) inequalities violated by the optimal solution of the (current node) continuous relaxation are introduced in the formulation of the problem.

We use the real-life problem that motivated this paper as a test bed for the aforementioned integer solution approaches. The company provided us with the historical monthly demand levels of 13 distributors. We generate 25 problem instances (*BI-SC-01* to *BI-SC-25*), each of them differing in the $12 * 13 = 156$ demand levels that must be satisfied by the supply chain. (Most of the instances are available in MPS format at [23].) The 25 instances were generated so that:

- they are representative of various demand types (smooth demand over the entire planning horizon, seasonal demand at different periods, etc.),
- the resulting problem is feasible with the production, inventory and distribution capacities of the supply chain.

Each of the 25 problem instances have been solved by the nine solution approaches discussed above, using an IBM IntellistationZ Pro with an Intel Xeon 3.2GHz CPU, 2GB of RAM and running Linux Fedora Core 5. Problem instances satisfying the characteristics above but resulting in optimization problem that could be solved in less than one minute CPU time were dropped and replaced by more complex ones. The optimization process was terminated when the relative mixed-integer (MIP) optimality gap falls below 0.15% or after 1 hour of CPU time. The total costs of the supply chain is a seven-digit number, and, thus, a 0.15% MIP gap represents a very significant amount worth consideration. Preliminary tests show that none of the instances could be solved to optimality within 4 hours of computing time with any of the proposed solution methods. This is the reason why we do not solve the problem instances to optimality but instead opt for a low tolerance level on the value of the MIP optimality gap.

We use a code in `Delphi` to generate a priori the binary-integer minimal covers and cuts. The cuts are then inserted in the `AMPL` formulation of the models. At each node of the tree, the branch-and-cut algorithm screens the cuts and incorporate these that are violated. We identify 3211 binary-integer covers for the problem on-hand, and we generate one Family I, one Family II and, on average, slightly more than four Convex Hull inequalities per minimal cover.

5.2 Computational results

To analyze the performance and reliability of the proposed integer methods, we draw the performance plot [24] of each of them. The curve associated with an algorithm, say *BCI*, represents the percentage of problems solved using *BCI* within a factor m of the time required by the fastest algorithm. If we denote by $t^*(P)$ the time needed by the fastest algorithm to solve problem P , then P is solved within a factor m by all solution approaches requiring less than $m t^*(P)$. It follows that, at $m = 1$, the plotted point for *BCI* returns the fraction of problem instances on which *BCI* is the fastest. The plotted point for *BCI* at $m = 60$ represents the proportion of problems that could be solved using up to 60 times more time than the fastest of the nine algorithms.

We first analyze the performance plot of the four branch-and-bound algorithms. Figure 3 clearly shows that *BB1* strictly dominates the three other branch-and-bound algorithms for any value of m . The *BB1* algorithm is the fastest and the most robust branch-and-bound algorithm: it is the fastest algorithm (among the nine proposed) for 12% of the considered problems and permits to reduce the MIP gap under 0.15% for 80% of the problem instances: *BB1* outperforms the other three branch-and-bound algorithms for these two criteria. It is important to note that *BB12* uses all the Family I cover inequalities used by *BB1*, and that *BBCH* uses the Family I facet-defining inequalities and replaces the non facet-defining ones by others that are stronger. The fact that *BB1* performs better than *BB12* and *BBCH* indicates that limiting the number of the considered cover inequalities is beneficial. We also observe that *BB12* dominates *BBCH*, and strictly dominates *BB2* except at $m = 1$. Finally, it appears that *BB12* and *BBCH* are more robust than *BB2*, since they find solutions with minimal MIP gap for a larger percentage of the problems.

We continue with the analysis of the four branch-and-cut algorithms (Figure 4). The *BC1* and *BC2* algorithms dominate *BC12* and *BCCH*. The *BC1* algorithm is the fastest (it is the fastest among the nine algorithms for 28% of the problems) and is with *BC12* the most robust branch-and-cut algorithm (i.e., both solve 68% instances with a MIP gap lower than or equal to 0.15%).

Figure 5 displays the performance plot of the `Cplex` 10.0 branch-and-cut algorithm, the non-dominated branch-and-bound (*BB1*) and branch-and-cut (*BC1* and *BC2*) algorithms.

The three non-dominated proposed algorithms (*BB1*, *BC1* and *BC2*) strictly dominate `Cplex` branch-and-cut algorithm. A possible explanation for this lies in the features of the considered problems. The presence of symmetry and the combined presence of large continuous (supply and inventory levels) and discrete variables make the bounds obtained by linear relaxation of the integer linear programming model not very tight, and limit the efficiency of general cutting planes used by MIP solvers [14]. The (*BB1* is the most robust by far, solving 80% of the considered instances, while `Cplex` (respectively, *BC1*, *BC2*) solves 56% (68%, 64%) of them. The *BB1* algorithm strictly dominates the other algorithm for m equal to or larger than 20, while the *BC1* algorithm is the fastest and strictly dominates the other algorithm for m strictly smaller than 20.

Table 1 reports the MIP optimality gap, the CPU computing time (in seconds) and the number of processed nodes for each problem instance solved using the *BB1*, *BC1* and *CPL* solution methods. The notation * indicates that the problem was solved to a MIP optimality gap of at most 0.15%. It is worth noting that, for the five instances whose MIP optimality gap is larger than 0.15% with each algorithm,

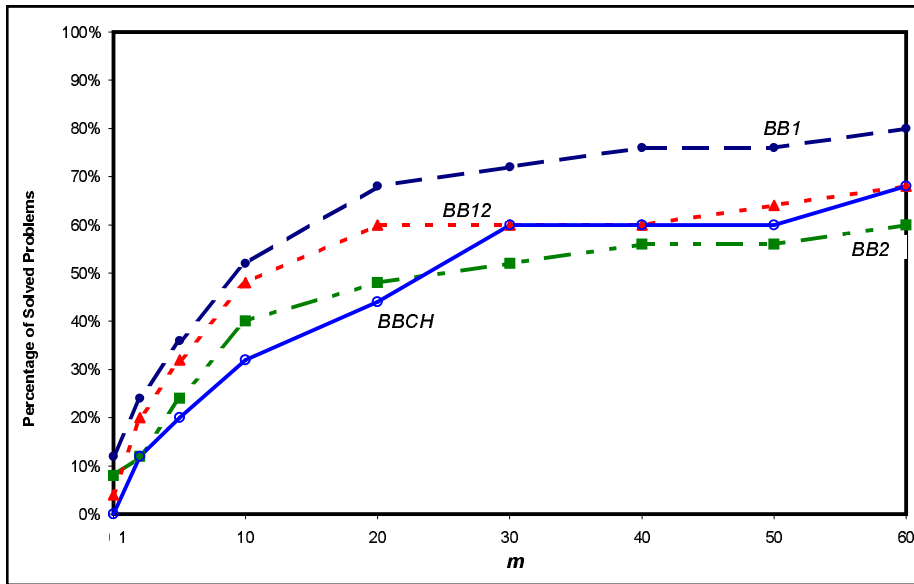


Figure 3: Performance plot of branch-and-bound algorithms.

the *BB1* algorithm is the one that allows closing the largest part of the MIP gap. Finally, we observe without surprise that the up-front introduction of all Family I cover inequalities in the *BB1* algorithm makes its average computing time per node much larger than that of the *BC1* and *CPL* algorithms.

The following conclusions can be safely drawn from the results of the computational experiments:

- the integer solution methods proposed in this paper clearly dominate CPLEX 10.0's branch-and-cut algorithm in terms of both speed and robustness;
- a branch-and-bound approach is to be employed if one favors robustness;
- a branch-and-cut approach is to be employed if one favors speed;
- the Family I of cover inequalities is much more efficient than the Family II of binary integer cover inequalities and the cover inequalities defining the convex hull: the *BC1* algorithm dominates all other algorithms for small values of m (i.e., when speed matters) and the *BB1* algorithm does the same for larger values of m (i.e., when robustness matters);
- the simultaneous consideration of many cover inequalities is detrimental to the efficiency of the algorithm. The solution approaches relying on the joint consideration of the Family I and Family II of cover inequalities (twice more inequalities than by using a single family at a time) and on the cover inequalities defining the convex hull (4.2 more inequalities than by using a single family at a time) does not help.

5.3 Empirical Comparison of Families of Cuts at the Root

We made a simple comparison of the lower bounds obtained on the 25 problem instances obtained at the root node by the Family I cuts, by Residual Capacity cuts [3] and by c-MIR inequalities [22].

The cuts generators used for residual capacity and MIR inequalities are from COIN-OR [12], respectively *CglResidualCapacity* and *CglMixedIntegerRounding*. All default values of the parameters of the

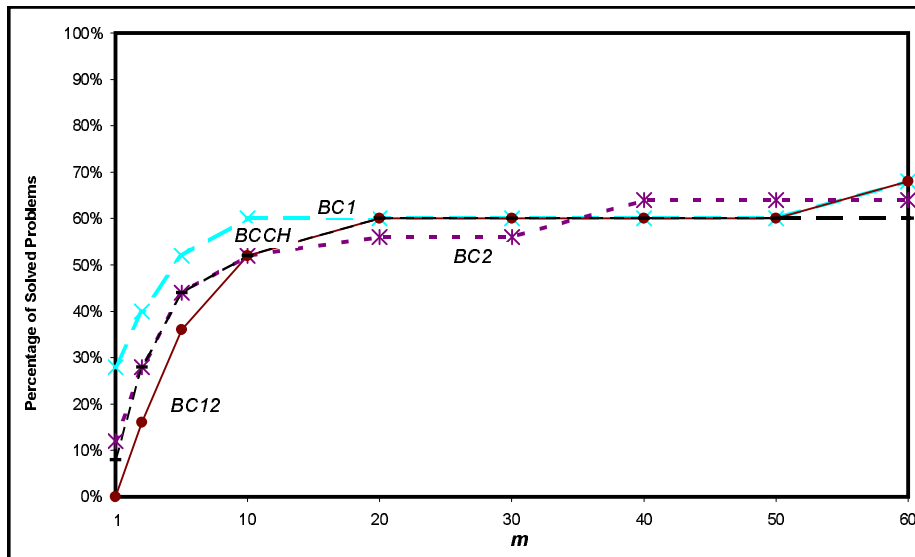


Figure 4: Performance plot of branch-and-cut algorithms.

generators are used, except that the aggregation parameter $MAXAGGR$ for c -MIR inequalities is set to 6, a fairly high value. No limit on the number of cutting rounds was set, i.e. cuts are added and the linear relaxation resolved until no more cuts are generated.

To our surprise, not a single residual capacity cut was generated, on any of the problems. Comparing the lower bound L_1 obtained with the Family I cuts with the lower bound L_2 obtained with the c -MIR cuts (percentages taken with respect to the best known feasible solution value z^*), L_1 is between 0.019% lower and 2.03% higher than L_2 and, on average, it is 0.42% higher. Lower bound L_1 is higher than L_2 on 22 out of 25 instances. The value of the average gap between z^* and L_1 is $(z^* - L_1)/z^* = 6.24\%$.

6 Conclusion

In this paper, we propose and test several exact integer solution approaches for the construction of a multi-period production-distribution scheduling problem.

This study is motivated by the problem faced by a chemical supply chain. The supply chain uses a heterogeneous fleet of ships to distribute its products and relies on a direct shipment policy with full-ship load deliveries. The construction of this model requires the solution of a mixed-integer programming problem whose complexity mainly originates from the distribution constraints. These are highly combinatorial, taking the form knapsack constraints containing many binary integer variables (manufacturer-to-distributors shipments) and one single general integer variable (supplier-to-manufacturer shipments).

The proposed integer solution techniques are based on the detailed analysis of these distribution constraints. We introduce the concept of binary-integer cover for binary-integer constraints and we study their convex hull. We then develop a constructive way of building valid cover inequalities: indeed, we derive a practical closed-form expression for generating them, upon the determination of a single parameter. We derive sufficient conditions for them to be facet-defining of a projection and develop procedures to strengthen them. We generalize the idea to extended covers.

The computational study shows the superiority of the solution techniques using the proposed cover inequalities over the state-of-the art CPLEX 10.0 branch-and-cut algorithm. Superiority must here be

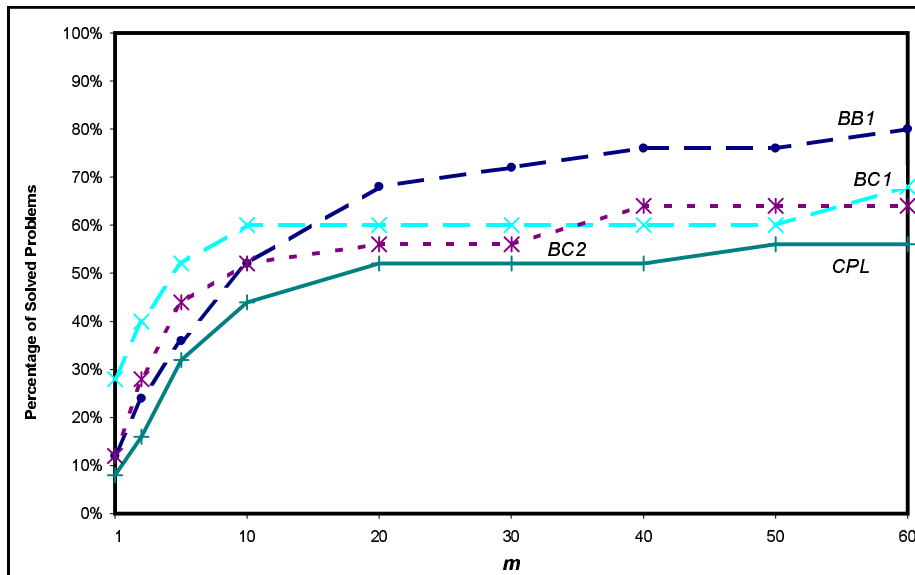


Figure 5: Performance plot of CPLEX and non-dominated algorithms.

understood in terms of robustness (percentage of problems solved with small MIP gap) and speed. The Family I of binary-integer cover inequalities is clearly the dominant family. A branch-and-cut algorithm using only the Family I inequalities is recommended if speed is the main criterion, while a branch-and-bound algorithm relying on the same cuts is to be preferred if robustness is the driving objective. It is worth underlining that the close-form expression of the proposed cover inequalities makes them very easy to generate and particularly attractive from an application perspective. This is what made the supply chain operator implement and use them to construct a significantly cheaper integrated inventory-production-distribution plan obtained through a much savvier usage of the distribution resources.

Acknowledgment: The authors express their appreciation to Professor A. Ruszczyński for his very insightful comments.

References

- [1] Agra A., Constantino M.F. 2007. Lifting Two-Integer Knapsack Inequalities. *Mathematical Programming* 109, 115-154.
- [2] Atamtürk A. 2005. Cover and Pack Inequalities for (Mixed) Integer Programming. *Annals of Operations Research* 139, 21-38.
- [3] Atamtürk A., Rajan D. 2002. On Splittable and Unsplittable Flow Capacitated Network Design Arc-Set Polyhedra. *Mathematical Programming* 92, 315-333.
- [4] Balas E. 1975. Facets of the Knapsack Polytope. *Mathematical Programming* 8, 146-164.
- [5] Balas E., Ceria S., Cornuejols G. 1993. A Lift-an-Project Cutting Plane Algorithm for Mixed 0-1 Programs. *Mathematical Programming* 58, 295-324.

Instances BI - SC	BB1			BC1			CPLEX		
	Gap	Time	# nodes	Gap	Time	# nodes	Gap	Time	# nodes
1	*	82.12	12524	*	90.45	97432	*	96.98	149076
2	*	69.77	10212	*	63.23	80038	*	79.61	128994
3	*	13.62	4184	*	323.10	48451	*	412.3	536805
4	*	132.66	57258	*	1639.77	3652746	0.1969	3600	6387401
5	0.167	3600	1616316	0.1669	3600	7451501	0.1669	3600	6919801
6	0.159	3600	1653931	0.1590	3600	7865801	0.1731	3600	8883901
7	*	668.45	116431	0.1841	3600	5265801	0.1906	3600	4997701
8	*	767.50	196222	*	637.64	1174924	0.1639	3600	6926801
9	*	328.12	82708	*	55.31	91450	*	2041.42	5206140
10	*	594.64	144000	*	125.35	117135	*	453.27	494430
11	*	94.25	15601	*	541.95	648240	*	111.82	157300
12	*	3457.45	663624	*	527.08	522530	*	1187.25	221533
13	*	559.26	133860	*	447.38	452398	*	1275.44	2610732
14	*	126.73	37680	*	1013.82	301440	*	243.56	562404
15	0.163	3600	1265901	0.1799	3600	4390072	0.1936	3600	7456739
16	*	341.52	64750	*	1707.61	323750	0.1557	3600	6970625
17	*	471.8	76560	*	324.76	502960	*	48.84	58080
18	0.241	3600	505501	0.4466	3600	5415015	0.2492	3600	4714201
19	0.241	3600	503101	0.4454	3600	5032393	0.3310	3600	4727872
20	*	101.53	22500	*	913.74	202500	*	151.43	309069
21	*	163.57	57258	*	163.57	57258	0.1897	3600	5854601
22	*	127.61	14760	*	510.43	640040	*	166.82	117408
23	*	2456.56	548941	0.1743	3600	1347296	0.1802	3600	7225136
24	*	612.34	142285	0.1523	3600	4980562	*	784.93	856103
25	*	449.98	72307	*	511.58	789024	*	649.45	690426

Table 1: Results for *BC1*, *BB1* and *CPLEX*; Gap is in percent and Time in seconds.

- [6] Ceria S., Cordier C., Marchand H., Wolsey L.A. 1998. Cutting Planes for Integer Programs with General Integer Variables. *Mathematical Programming* 81 (2), 201-214.
- [7] Chandra P., Fisher M.L. 1994. Coordination of Production and Distribution Planning. *European Journal of Operational Research* 72 (3), 503-517.
- [8] Chen Z.-L. 2003. Integrated Production and Distribution Operations: Taxonomy, Models, and Review. In: *Supply Chain in the E-Business Era*, edited by Simchi-Levi D. and Shen Z.-J. Kluwer Academic Publishers.
- [9] Chen Z.-L., Vairaktarakis G.L. 2005. Integrated Scheduling of Production and Distribution Operations. *Management Science* 51 (4), 614-628.
- [10] Christiansen M., Fagerholt K., Ronen D. 2004. Ship Routing and Scheduling: Status and Perspectives. *Transportation Science* 38 (1), 118.
- [11] Chvátal V. 1973. Edmonds Polytopes and a Hierarchy of Combinatorial Problems. *Discrete Mathematics* 87, 37-56.
- [12] COIN-OR Cut generator Library, <https://projects.coin-or.org/Cgl>

- [13] Cornuéjols G. 2007. Valid Inequalities for Mixed Integer Linear Programs. *Mathematical Programming*. Forthcoming.
- [14] Dauzère-Pérès G., Nordli A., Olstad A., Haugen K., Koester U., Myrstad P.O., Teistklub G., Reistad A. 2007. Omya Hustadmarmor: Optimizing the Supply Chain of Calcium Carbonate Slurry to the European Paper Making Industry. *Interfaces* 37, 39-51.
- [15] Dhaenens-Flipo C., Finke G. 2001. An Integrated Model for an Industrial Production-Distribution Problem. *IIE Transactions* 33, 705 - 715.
- [16] Fisher M., Kamalani R., Yu-Sheng Z. 2001. Ending Inventory Valuation in Multiperiod Production Scheduling. *Management Science* 45 (5), 679-692.
- [17] Fumero F., Vercellis C. 1999. Synchronized Development of Production, Inventory, and Distribution Schedules. *Transportation Science* 33, 330-340.
- [18] Geunes J., Pardalos P.M. 2003. Network Optimization in Supply Chain Management and Financial Engineering: An Annotated Bibliography. *Networks* 42 (2), 66-84.
- [19] Hammer P.L., Johnson E.L., Peled U.N. 1975. Facets of Regular 0-1 Polytopes. *Mathematical Programming* 8, 179-206.
- [20] van Hoesel S.P.M., Koster A.M.C.A., van de Leensel R.L.M.J., Savelsbergh M.W.P. 2004. Bidirected and Unidirected Capacity Installation in Telecommunication Networks. *Discrete Applied Mathematics* 133 (1-3), 103-121.
- [21] Lejeune M.A., Ruszczyński A. 2007. An Efficient Method Trajectory Method for Probabilistic Production-Inventory-Distribution Problems. *Operations Research* 55, 378-394.
- [22] Marchand H., Wolsey L.A. 2001. Aggregation and Mixed Integer Rounding to Solve MIPS. *Operations Research* 49 (3), 363-371.
- [23] <http://wpweb2.tepper.cmu.edu/fmargot/mps.html>
- [24] Molan E.D., Moré J.J. 2002. Benchmarking Optimization Software with Performance Profiles *Mathematical Programming* 91, 201-213.
- [25] Owen J.H., Mehrotra S. 2001. A Disjunctive Cutting Plane Procedure for General Mixed-Integer Linear Programs. *Mathematical Programming* 89 (3), 437-448.
- [26] Owen J.H., Mehrotra S. 2002. On The Value of Binary Expansions for General Mixed-Integer Linear Programs. *Operations Research* 50 (5), 810-819.
- [27] Rothberg E. 2000. Using Cuts to Remove Symmetry. *Presentation at 17th International Symposium on Mathematical Programming*. Atlanta, GA.
- [28] Sarmiento A.M., Nagi R. 1999. A Review of Integrated Analysis of Production-Distribution Systems. *IIE Transactions* 31 (11), 1061-1074.
- [29] Swaminathan J.M., Tayur S.R. 2003. Tactical Planning Models for Supply Chain Management. In *OR/MS Handbook on Supply Chain Management: Design, Coordination and Operation*, edited by Graves S.C. and de Kok T., Elviesier Publishers, 423-456.
- [30] Wolsey L.A. 1998. *Integer Programming*. Wiley. New York, NY.