The Knowledge Engineering Review, Vol. 16:1, 2001, 25–39
Printed in the United Kingdom
Copyright © 2001, Cambridge University Press

Branch-and-cut for combinatorial optimisation problems without auxiliary binary variables

I. R. de FARIAS, JR.,[1] E. L. JOHNSON[2] and G. L. NEMHAUSER[2]

[1] *State University of New York at Buffalo, 403 Bell Hall, Buffalo, NY 14260-2050, USA. Partially supported by CNPq, Brazilian reserach agency and NSF grant DMi-0100020*
[2] *School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0205, USA. Partially supported by NSF grant DMI-9700285 and ILOG*

Abstract

Many optimisation problems involve combinatorial constraints on continuous variables. An example of a combinatorial constraint is that at most one variable in a group of nonnegative variables may be positive. Traditionally, in the mathematical programming community, such problems have been modeled as mixed-integer programs by introducing auxiliary binary variables and additional constraints. Because the number of variables and constraints becomes larger and the combinatorial structure is not used to advantage, these mixed-integer programming models may not be solved satisfactorily, except for small instances. Traditionally, constraint programming approaches to such problems keep and use the combinatorial structure, but do not use linear programming bounds in the search for an optimal solution. Here we present a branch-and-cut approach that considers the combinatorial constraints without the introduction of binary variables. We review the development of this approach and show how strong constraints can be derived using ideas from polyhedral combinatorics. To illustrate the ideas, we present a production scheduling model that arises in the manufacture of fibre optic cables.

1 Introduction

Several classes of optimisation problems consist of maximising a linear or a concave quadratic function of continuous decision variables subject to linear constraints. When the variables are subject exclusively to the linear constraints, the resulting problem can be solved in polynomial time, for example by an interior point algorithm. In many applications, however, the variables are also subject to other constraints that are combinatorial and make the set of feasible solutions non-convex. In this case, the resulting problem may be NP-hard. Examples of hard combinatorial constraints that often occur in linear or concave quadratic optimisation problems with linear constraints include

- special ordered sets of type I (SOS1) constraints,
- special ordered sets of type II (SOS2) constraints,
- cardinality constraints and
- semi-continuous constraints.

A set of variables is SOS1 (Beale & Tomlin, 1970) when at most one variable in the set can be nonzero. This constraint appears, for example, in linear complementarity (de Farias *et al*., 1998), refining (Healy, 1964), capital budgeting, pricing, and manufacturing problems (Nauss, 1978).

A set of variables is SOS2 (Beale & Tomlin, 1970) when at most two variables in the set can be nonzero, and in case two variables in the set are nonzero, they must be adjacent in the set. For example, if $\{x_1, x_2, x_3, x_4, x_5\}$ is SOS2 and $x_3 \neq 0$ in a feasible solution, then $x_1 = x_5 = 0$, and at least one of x_2 or x_4 must be 0 in that solution. This constraint appears, for example, in the piecewise linear

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

approximation of nonlinear separable functions and production scheduling (de Farias, Johnson & Nemhauser, 2000; Gue *et al*., 1997).

A cardinality constraint is a generalisation of SOS1 in which no more than a specified number of variables is nonzero. A variable x_j is semi-continuous when it is either equal to 0 or greater than or equal to a positive number, i.e. $x_j \in \{0\} \cup [l_j, u_j]$, where $0 < l_j < u_j$, and $u_j \in \Re \cup \{\infty\}$. Cardinality constraints and semicontinuous variables appear, for example, in portfolio optimisation (Bienstock, 1996), discrete location-allocation (Brimberg & Love, 1998; de Farias, 2001), and synthesis of process networks (Biegler *et al*., 1997; Raman & Grossmann, 1993).

Traditionally, these constraints have been modeled by introducing auxiliary binary variables and additional constraints relating the binary and the continuous variables. For example, suppose that $0 \le x_1 \le u_1, \ldots, 0 \le x_l \le u_l$. If $\{x_1, \ldots, x_l\}$ is SOS1, we introduce the binary variables $y_1, \ldots, y_l$, and the constraints

$$x_j \le u_j y_j, \tag{1}$$

$j \in \{1, \ldots, l\}$, and

$$\sum_{j=1}^{l} y_j \le 1.$$

If $\{x_1, \ldots, x_l\}$ is SOS2, we introduce the binary variables $y_1, \ldots, y_{l-1}$, and the constraints

$$\sum_{j=1}^{l} y_j \le 1,$$

$$x_1 \le u_1 y_1,$$

$$x_j \le u_{j-1} y_{j-1} + u_j y_j,$$

$j \in \{2, \ldots, l-1\}$, and

$$x_l \le u_l y_{l-1}.$$

If x_j is semi-continuous, we introduce the binary variable y_j and the constraints

$$l_j y_j \le x_j \le u_j y_j.$$

These MIP models were introduced first by Markowitz and Manne (1957) and Dantzig (1960). Markowitz and Manne showed that piecewise linear approximations to nonlinear separable functions, commonly used, for example, in problems with economies of scale, can be modelled as MIPs. Dantzig, motivated by the excitement that followed Gomory's cutting plane algorithm (1958), modelled several hard combinatorial constraints that occur in practice as MIPs. In general, large instances of these problems have resisted the attack of general-purpose MIP software.

A different approach was pioneered by Beale and Tomlin (1970). In their seminal paper, they suggested dispensing with the use of auxiliary binary variables to model piecewise linear approximations to nonlinear separable functions and multiple-choice constraints and adopting a specialised branching strategy in a branch-and-bound scheme as a way to enforce the combinatorial constraints. Later, Beale (1979, 1980, 1985) suggested the use of the same principle to deal with semi-continuous variables and other variations of special ordered sets. The suggestions contained in Beale (1979, 1980, 1985) and Beale & Tomlin (1970) are widely recognised as yielding more effective

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

branch-and-bound algorithms, and they have been implemented in commercial and academic optimisation software.

More recently, Beaumont (1990), building upon earlier work by Balas (1979), presented a branch-and-bound algorithm for disjunctive programming problems that dispenses with the use of auxiliary binary variables, by branching on the disjunctions and replacing the continuous relaxation of the usual MIP formulation, which usually contains several inequalities, with a surrogate inequality that is as strong. Computational results reported in Beaumont (1990) indicate the effectiveness of this approach.

It has been demonstrated that branch-and-cut approaches, which use the polyhedral structure of the problem, frequently improve the performance of branch-and-bound methods substantially; see, for instance, Crowder *et al*. (1983), Grötschel *et al*. (1984), Nemhauser & Wolsey (1988) and Padberg & Rinaldi (1987). However, there has been very little progress on the development of branch-and-cut approaches to solve combinatorial optimisation problems without the use of auxiliary binary variables. de Farias (1995) and de Farias, Johnson & Nemhauser (2000) study a scheduling problem with SOS2 without introducing auxiliary binary variables for which branch-and-cut is much more effective than branch-and-bound. Bienstock (1996) uses a branch-and-cut approach without binary variables to study portfolio selection problems. de Farias *et al*. (1998) study the polytope of the complementarity knapsack problem formulated without binary variables.

Constraint programmers often write constraints without auxiliary binary variables, but frequently the resulting relaxations are poor with respect to obtaining bounds. Thus the development of efficient computational methods to derive strong inequalities without binary variables may have a significant impact on constraint programming. It will provide the means to use better relaxations and therefore improve its ability to solve hard combinatorial optimisation problems. On the other hand, by not using auxiliary binary variables, it will be possible to incorporate within branch-and-cut some of the search methods of constraint programming, and thus to build a truly mixed strategy that includes the best from both communities.

In this paper we discuss branch-and-cut approaches to optimisation problems with continuous variables and combinatorial constraints in which auxiliary binary variables are not included in the model. Rather, the combinatorial constraints are enforced directly in the algorithm through the use of specialised branching and strong inequalities valid in the space of the continuous variables.

The paper is organised as follows. In Section 2 we review specialised branching. We show how we can enforce SOS1, SOS2, cardinality and semi-continuous constraints directly in the branch-and-bound algorithm through specialised branching schemes. In Section 3 we show how to generate inequalities in the space of continuous variables that are valid for the convex hull of the set of feasible solutions. These inequalities may be used as cuts in a branch-and-cut scheme. We first present inequalities that are easy to derive, but that in general define faces of the convex hull of the set of feasible solutions that have low dimension. Then, we show how we can derive inequalities that are more expensive but in general define faces of higher dimension. In Section 4 we present a generalised assignment problem that arises in the scheduling of fibre-optic cable manufacturing, in which SOS2 constraints appear naturally in the formulation. We apply the ideas presented in Section 3 to obtain facets of the convex hull of the set of feasible solutions of the problem, and we present computational results that demonstrate the usefulness of these inequalities. In Section 5 we present conclusions and directions for further research.

Throughout the paper we define the relaxation of a problem as the problem resulting by eliminating the combinatorial constraints and we also assume, without loss of generality, that all variables are non-negative.

2 Specialised branching

In this section we study branching schemes specific for problems with SOS1, SOS2, cardinality and semicontinuous constraints. These branching schemes enforce the combinatorial constraints directly in the branch-and-bound algorithm and they dispense with the introduction of binary variables in the

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

model. We also discuss the advantages of the specialised branching schemes relative to the traditional approach of introducing binary variables in the model and enforcing integrality by branching on the binary variables.

2.1 SOS1

Let $\{x_1, \ldots, x_l\}$ be SOS1. Suppose that the relaxed subproblem at the current node of the branch-and-bound tree assigns a nonzero value to x_r and x_t, where $1 \le r \le t \le l$. Let s be an index with $r \le s \le t$. Because of the SOS1 constraint, either

$$x_1 = \cdots = x_s = 0 \tag{2}$$

or

$$x_{s+1} = \cdots = x_l = 0. \tag{3}$$

This means that we can branch on the sequence of variables $(x_1, \ldots, x_l)$ in the same way as we branch on the values of a binary variable. Thus we define one of the descendent branches of the current node in the branch-and-bound enumeration tree by imposing (2), and the other branch by imposing (3). (2) and (3) eliminate the optimal solution of the relaxation at the current node from both branches.

In many applications, it is required that the variables in the SOS1 satisfy the constraint

$$\sum_{j=1}^{l} x_j = 1.$$

In this case, the variables are binary. Practical experience has demonstrated that even in this case, the use of the specialised branching scheme is advantageous.

This specialised branching scheme is more useful, in general, when the combinatorial structure of the problem suggests a natural ordering of the variables. For example, suppose that the cost of building a warehouse varies with the size of the warehouse as in Table 1. The cost can be modelled as

$$0x_1 + 30x_2 + 50x_3 + 60x_4 + 65x_5 + 70x_6,$$

and the size as

$$0x_1 + x_2 + 5x_3 + 15x_4 + 30x_5 + 50x_6, \tag{4}$$

where $x_j \in [0, 1]$, $\forall j \in \{1, \ldots, 6\}$, $\Sigma_{j=1}^{6} x_j = 1$, and $\{x_1, \ldots, x_6\}$ is SOS1. Note that the problem itself suggests the order of the variables in the SOS1 constraint.

Suppose now that the optimal value of the LP relaxation is x^*, given by $x_1^* = \frac{4}{5}$, $x_2^* = x_3^* = x_4^* = x_5^* = 0$, and $x_6^* = \frac{1}{5}$, which gives a size equal to 10. We may use this LP relaxation solution as a suggestion on how to partition the solution set for branching. We may partition the solution set by restricting our search to a size equal to 0, 1 or 5 in one branch, and to a size equal to 15, 30 or 50 in the other branch. This can be accomplished by fixing $x_4 = x_5 = x_6 = 0$ in the first branch, and $x_1 = x_2 = x_3 = 0$ in the other. This branching means that either we do not want to build a warehouse greater than 5, or we want to build a warehouse at least as great as 15.

Table 1 Cost of building a warehouse

Size	Cost
0	0
1	30
5	50
15	60
30	65
50	70

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

This branching scheme is more powerful than branching on the variables individually. In the example, we have two options for branching on individual variables. We may branch on x_1 or x_6. Suppose we branch on x_1. The branch $x_1 = 1$ means that we do not want to build a warehouse. The branch $x_1 = 0$ means that we want to build a warehouse. On the other hand, if we branch on x_6, the branch $x_6 = 1$ means that we want to build a warehouse of the largest possible size, and the branch $x_6 = 0$ means that we do not want to build a warehouse of the largest size. In any case, the specialised branching scheme provides us with a more meaningful option. With the specialised branching scheme we can take advantage of the information provided by the optimal solution of the relaxation and in general we progress faster towards an optimal solution.

2.2 SOS2

Let $\{x_1, \ldots, x_l\}$ be SOS2. Suppose that the relaxed subproblem at the current node of the branch-and-bound tree assigns a nonzero value to x_r and x_t, where $1 \le r \le t - 2 \le l$. Let s be an index with $r < s < t$. Because of the SOS2 constraint, either

$$x_1 = \cdots = x_{s-1} = 0 \tag{5}$$

or

$$x_{s+1} = \cdots = x_l = 0. \tag{6}$$

Similarly to the case of SOS1, we can branch on the sequence of variables $(x_1, \ldots, x_l)$. We define one of the descendent branches of the current node in the branch-and-bound enumeration tree by imposing (5), and the other branch by imposing (6). Inequalities (5) and (6) eliminate the optimal solution of the relaxation at the current node from both branches.

The meaning of an SOS2 constraint depends on the order in which the variables occur in the set. As with SOS1, the specialised branching for SOS2 is more useful, in general, when the combinatorial structure of the problem suggests a natural ordering of the variables.

2.3 Cardinality constraints

Let $N = \{1, \ldots, n\}$. Suppose that at most k of the variables $x_1, \ldots, x_n$ can be nonzero in a feasible solution, where $k < n$. Let $r \in N$. Because of the cardinality constraint we may divide the solution set by imposing that

$$x_r = 0 \tag{7}$$

in one subset and that

$$\text{at most } k - 1 \text{ of the variables } x_j, j \in N - \{r\}, \text{ can be nonzero} \tag{8}$$

in the other subset. The union of the two subsets is equal to the solution set. Note that their intersection is nonempty. To simplify notation, suppose that $x_j \le 1$, $\forall j \in N$. Because it may not be possible to impose (8) in a simple way, we will consider a different division of the solution set in which the second subset is defined by imposing that

$$\sum_{j \in N - \{r\}} x_j \le k - 1. \tag{9}$$

Note that the solution to (9) contains the solution to (8). A variable dichotomy branching scheme based on (7) and (8) was proposed in Bienstock (1996). Suppose that at the current node of the branch-and-bound tree $x_j, j \in F$, are the variables that have not been branched on, and that the node was defined by imposing that

$$\sum_{j \in F} x_j \le k - t,$$

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

where $0 \leq t < k$, and $t = 0$ refers to the root node. Suppose now that the relaxed subproblem at the current node of the branch-and-bound tree assigns a nonzero value to $x_j \forall j \in R$, where $R \subseteq F$ and $|R| > k - t$. If $r \in R$, we may branch by imposing (7) in one branch and

$$\sum_{j \in F - \{r\}} x_j \leq k - t - 1 \tag{10}$$

in the other branch. The inequality (10) does not necessarily eliminate the optimal solution of the relaxation at the current node from the branch it defines, except after k branches, when $t = k$.

An alternative branching scheme, in which one branches on sets of variables rather than on individual variables, was proposed by de Farias, Johnson & Nemhauser (forthcoming). Let $S \subseteq N$, and $0 \leq s \leq \min\{k, |S|\}$. We may partition the solution set by imposing that

$$\text{at most } s \text{ of the variables } x_j,\ j \in S, \text{ can be nonzero,} \tag{11}$$

and

$$\text{at least } s + 1 \text{ of the variables } x_j,\ j \in S, \text{ must be nonzero.} \tag{12}$$

Because of the cardinality constraint, (12) implies that

$$\text{at most } k - s - 1 \text{ of the variables } x_j,\ j \in N - S, \text{ can be nonzero.} \tag{13}$$

Again, because it may not be possible to impose (11) and (13) in a simple way, we replace (11) and (13) by

$$\sum_{j \in S} x_j \leq s \tag{14}$$

and

$$\sum_{j \in N - S} x_j \leq k - s - 1, \tag{15}$$

respectively, and we divide the solution set by imposing (14) in one subset and (15) in the other subset. Note that when $S = \{r\}$ and $s = 0$, this division scheme coincides with the variable dichotomy division scheme of Bienstock (1996).

Suppose that the current node of the branch-and-bound tree was defined by imposing (14). The case $S = N$ and $s = k$ refers to the root node. Suppose that the relaxed subproblem at the current node assigns a nonzero value to x_j, $\forall j \in R$, where $R \subseteq S$ and $|R| > s$. We may branch by imposing

$$\sum_{j \in R_1} x_j \leq \left\lfloor \frac{s}{2} \right\rfloor \tag{16}$$

in one branch, and

$$\sum_{j \in R - R_1} x_j \leq s - \left\lfloor \frac{s}{2} \right\rfloor - 1, \tag{17}$$

in the other branch, where $R_1 \subset R$ and $|R_1| = \lfloor |R|/2 \rfloor$. Note that (16) and (17) do not necessarily eliminate the optimal solution of the relaxation at the current node from the branches they define. However, they guarantee to eliminate it in about $\log k$ branches. Preliminary computations reported in de Farias, Johnson and Nemhauser (forthcoming) comparing this branching scheme with the variable dichotomy scheme of Bienstock (1996) indicate that this branching scheme is more effective.

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

2.4 Semi-continuous variables (*Beale*, 1985)

Let $x_j \in \{0\} \cup [l_j, u_j]$, where $0 < l_j < u_j$. Suppose that the relaxed subproblem at the current node of the branch-and-bound tree assigns a value $x_j^* \in (0, l_j)$ to x_j. We can then define one branch by imposing

$$x_j = 0 \tag{18}$$

and the other branch by imposing

$$x_j \geq l_j. \tag{19}$$

Constraints (18) and (19) eliminate the optimal solution of the relaxation at the current node for both branches.

2.5 Motivation for specialised branching

The most obvious drawback of introducing binary variables is that they can double the number of variables and they increase substantially the number of rows. The introduction of binary variables, however, has other negative effects.

In many real-life situations it is not clear how large the variable values can be. When we introduce binary variables, the continuous variables must be bounded. To overcome this situation one often has to introduce big-M constraints, such as (1). Frequently, these big-M constraints are not tight. When they are, it is usually because $y_j = 0$, and in this case they increase the degeneracy of the solution of the relaxation (Savelsbergh, personal communication). The big-M constraints also increase substantially the rank of the constraint matrix. If the objective function is quadratic, for example, this will make the relaxations much harder (Bienstock, 1996).

The introduction of binary variables also tends to obscure the combinatorial structure of the problem. For exarmple, as noted in Hooker & Osorio (1997), when we introduce binary variables we may obtain basic optimal solutions of the resulting model that are fractional but satisfy the combinatorial constraints. General purpose MIP software, in this case, will continue to branch unnecessarily, trying to achieve integrality of the binary variables.

3 Valid inequalities

In this section we discuss how we can derive inequalities that are valid for the set of feasible solutions of the problem, modelled with continuous variables only, that cut off solutions that do not satisfy the combinatorial constraints. Such inequalities are called cuts. First, we present cuts that are simple to derive but in general define faces of the convex hull of the set of feasible solutions that have low dimension. Then, we show, by considering problems with SOS1 constraints, how we can derive cuts that are harder to compute but in general define faces of higher dimension.

3.1 Simple inequalities

Given a basic optimal solution of the relaxation in the current node that is not feasible, it is usually easy to derive an inequality that is valid and that cuts off this solution (Ibaraki, 1973; Jeroslow, 1978).

Theorem 1 *Suppose that at most k of the variables* $x_1, \ldots, x_{k+1}$ *can be positive and that, in a basic solution,*

$$\begin{aligned} x_1 &= \alpha_{10} - \Sigma_{j \in N}\, \alpha_{1j} x_j \\ &\vdots \\ x_{k+1} &= \alpha_{k+1,0} - \Sigma_{j \in N}\, \alpha_{k+1,j} x_j, \end{aligned}$$

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

where $\alpha_{i0} > 0 \forall i \in \{1, \ldots, k+1\}$, *and* x_j, $j \in N$, *are the nonbasic variables. Let*

$$\beta_j = max\left\{\frac{\alpha_{1j}}{\alpha_{10}}, \ldots, \frac{\alpha_{k+1,j}}{\alpha_{k+1,0}}\right\}.$$

Then,

$$\sum_{j \in N} \beta_j x_j \geq 1 \tag{20}$$

is a valid inequality.

Proof Because at most k variables can be positive at least one of the following equalities must hold:

$$\sum_{j \in N} \frac{\alpha_{1j}}{\alpha_{10}} x_j = 1$$

$$\vdots$$

$$\sum_{j \in N} \frac{\alpha_{k+1,j}}{\alpha_{k+1,0}} x_j = 1.$$

This implies that (20) is valid. □

The inequality (20) cuts off the current basic optimal solution of the relaxation that does not satisfy the cardinality constraint. A similar type of inequality can be derived when the variables are subject to SOS1 and SOS2 constraints.

We now consider the case of semi-continuous constraints.

Theorem 2 *Suppose that* $x_r \in \{0\} \cup [l_r, u_r]$, *where* $0 < l_r < u_r$. *Suppose that in a basic optimal solution of the relaxation*,

$$x_r = \alpha_{r0} - \sum_{j \in N} \alpha_{rj} x_j,$$

where $0 < \alpha_{r0} < l_r$, *and* $x_j, j \in N$, *are the non-basic variables. Let* $N^- = \{j \in N : \alpha_{rj} < 0\}$ *and* $N^+ = \{j \in N : \alpha_{rj} > 0\}$. *If* $N^+ = N^- = \emptyset$, *the problem is infeasible. If* $N^+ \neq \emptyset$, *and* $N^- = \emptyset$, *then*

$$x_r = 0$$

is a valid inequality. If $N^+ = \emptyset$ *and* $N^- \neq \emptyset$, *then*

$$x_r \geq l_r$$

is a valid inequality. If $N^+ \neq \emptyset$ *and* $N^- \neq \emptyset$, *then*

$$\sum_{j \in N^+} \frac{\alpha_{rj}}{\alpha_{r0}} x_j + \sum_{j \in N^-} \frac{\alpha_{rj}}{\alpha_{r0} - l_r} x_j \geq 1 \tag{21}$$

is valid.

Proof We only need to prove the last part of the theorem. If $N^+ \neq \emptyset$, then either

$$\sum_{j \in N} \alpha_{rj} x_j = \alpha_{r0}$$

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

or

$$\sum_{j \in N} -\alpha_{rj} x_j \geq l_r - \alpha_{r0}.$$

In the first case,

$$\sum_{j \in N^+} \alpha_{rj} x_j \geq \alpha_{r0}, \tag{22}$$

and in the second case,

$$\sum_{j \in N^-} -\alpha_{rj} x_j \geq l_r - \alpha_{r0}. \tag{23}$$

Because one of (22) or (23) must be satisfied, (21) must hold. □

In Section 2.3 we saw that the branching strategies that we presented for problems with cardinality constraints do not guarantee that the optimal solution of the relaxation at the current node will be eliminated from both branches of the branch-and-bound tree. Inequality (20), however, always cuts off any basic solution of the relaxed problem that is not feasible.

3.2 Strong inequalities

It has been demonstrated that branch-and-cut approaches frequently improve the performance of pure branch-and-bound methods when they use inequalities which define high-dimensional faces of the convex hull of the set of feasible solutions of the problem (see, for instance, Crowder *et al*. (1983), Grötschel *et al*. (1984), Nemhauser & Wolsey (1988) and Padberg & Rinaldi (1987)). However, unless the problem has some special structure that we can take advantage of, it is difficult to derive valid inequalities that are guaranteed to define faces of high dimension.

Crowder *et al*. (1983) suggested using strong inequalities derived from the individual linear constraints of the problem (knapsack relaxations). They were able to solve to optimality several real-life 0-1 programming problems that were unsolvable at the time.

Some facet-defining inequalities of these knapsack relaxations are easy to identify. For example, consider the case of problems with SOS1 constraints. Let $M = \{1, \ldots, m\}$, $N_i = \{1, \ldots, n_i\}$, $i \in M$, $d = \sum_{i=1}^{m} n_i$, and

$$S = \{x \in [0, 1]^d : \sum_{i \in M} \sum_{j \in N_i} a_{ij} x_{ij} \leq b \text{ and } \{x_{i1}, \ldots, x_{in_i}\} \text{ is SOS1 } \forall i \in M\}.$$

Assume that all inequality coefficients are positive. Let $PS = \text{conv}(S)$. We have that (de Farias *et al*., 1998)

Proposition 1 *$x_{ij} > 0$ are facet-defining inequalities for PS $\forall j \in N_i$, $i \in M$.*

Proposition 2 *Let $j_i \in N_i$ be such that $a_{ij_i} = max\{a_{ij} : j \in N_i\}$. $\Sigma_{i \in M} \Sigma_{j \in N_i} a_{ij} x_{ij} \leq b$ is facet-defining iff $\Sigma_{i \in M - i'} a_{ij_i} + a_{i'j'} \geq b$ $\forall j' \in N_{i'}$, $i' \in M$.*

Proposition 3 *For $i \in M$,*

$$\sum_{j \in N_i} x_{ij} \leq 1 \tag{24}$$

is facet-defining iff $a_{ij} < b$ for some $j \in N_i$.

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

These inequalities are called trivial.

In general, identifying non-trivial strong valid inequalities is not obvious even for knapsack relaxations. However, we can simplify the problem further, so that finding such inequalities is easier, and then build from these inequalities to obtain facets of the knapsack polytopes. One such procedure, called lifting, was introduced by Gomory (1965) in the context of the group problem, and was later studied by several authors (see de Farias (1995) and Nemhauser & Wolsey (1988) for a discussion of lifting). In a lifting procedure, we initially project the polyhedron for which we want to derive facets to a lower dimensional space by fixing some of the variables. Then we obtain a facet-defining inequality of the projected polyhedron. Finally, we derive a facet-defining inequality of the original polyhedron by introducing the fixed variables in the inequality obtained in the previous step. The variables may be introduced sequentially (one at a time) or simultaneously (in groups). Although simultaneous lifting is more general, it is usually prohibitively expensive computationally. So in practice sequential lifting procedures are usually used. For a general result on sequential lifting, see Wolsey (1976).

An important issue is what the initial projected polyhedron should be. We illustrate the answer to this question by considering the polytope *PS*. Let $C = \{i_1 j_1, \ldots, i_r j_r\} \subset M \times N$. Suppose that we fix the variables indexed by $M \times N - C$ at 0. Clearly, the projected polytope is interesting only when

$$\sum_{ij \in C} a_{ij} > b.$$

The set C is called a cover, and the inequality

$$\sum_{ij \in C} a_{ij} x_{ij} \leq b$$

is called a cover inequality. As shown in de Farias *et al*. (1998), in the case of *PS* it suffices to fix the variables at 0. In de Farias (1995) it is shown that we can obtain all non-trivial facets of *PS* by lifting cover inequalities.

The following example illustrates lifting.

Example 1 Let $m = 3$, $n_1 = n_2 = n_3 = 2$, and the knapsack inequality be

$$(6x_{11} + x_{12}) + (8x_{21} + 3x_{22}) + (12x_{31} + 9x_{32}) \leq 13. \tag{25}$$

We take $C = \{11, 21\}$, and fix x_{12}, x_{22}, x_{31}, and x_{32} at 0. The cover inequality is

$$6x_{11} + 8x_{21} \leq 13. \tag{26}$$

Let

$$(6x_{11} + \alpha_{12} x_{12}) + 8x_{21} \leq 13$$

be the inequality obtained by lifting (26) with respect to x_{12}. When x_{12} is positive, $x_{11} = 0$. This means that α_{12} can be as great as 5. By repeating the same argument, we can now introduce x_{22} in the inequality with a coefficient equal to 7, i.e.

$$(6x_{11} + 5x_{12}) + (8x_{21} + 7x_{22}) \leq 13. \tag{27}$$

We now introduce x_{31} in the inequality. Let α_{31} be the coefficient, i.e.

$$(6x_{11} + 5x_{12}) + (8x_{21} + 7x_{22}) + \alpha_{31} x_{31} \leq 13. \tag{28}$$

Because $x_{11} = x_{22} = 1$, $x_{31} = \frac{1}{3}$, $x_{12} = x_{21} = x_{32} = 0$ is a feasible solution, $\alpha_{31} = 0$. In the same way, the coefficient of x_{32} is 0. This means that (27) defines a facet of *PS*. (27) cuts off the point $x_{11} = x_{12} = 1$, $x_{21} = \frac{3}{4}$, $x_{22} = x_{31} = x_{32} = 0$, which is a vertex of $\{x \in [0, 1]^6 : (6x_{11} + x_{12}) + (8x_{21} + 3x_{22}) + (12x_{31} + 9x_{32}) \leq 13\}$. □

de Farias (1995) and de Farias *et al*. (1998) contain a polyhedral study of *PS*. de Farias (1995) and de Farias, Johnson & Nemhauser (forthcoming) contain several results on lifted cover inequalities for

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

problems with SOS2. de Farias, Johnson nad Nemhauser (forthcoming) contains results on lifted cover inequalities for problems with cardinality constraints.

4 A Generalised assignment problem with SOS2

In this section we present a simplified version of a production scheduling problem in which the concept of SOS2 arises naturally in the formulation. This problem was first discussed in Gue *et al*. (1997) in the context of fibre-optic cable manufacturing. de Farias (1995) and de Farias, Johnson & Nemhauser (2000) contain a polyhedral discussion of the problem. Based on these polyhedral results, de Farias and Nemhauser (2000) developed an efficient branch-and-cut scheme for the 0-1 generalised assignment problem.

4.1 Problem formulation

We are given a machine and several orders that must be produced within a time horizon. The processing of an order is non-pre-emptible. The time horizon is divided into time periods of equal length, corresponding to shifts. The shifts are long enough to produce any order. Instead of defining starting and ending times of the production of an order, we decide, at this level, what fraction of the order is produced in each time period. The capacity of the machine in each time period is equal to the total machine-hours available in the time period. The capacity consumed by an order is equal to the number of hours needed to produce the order. The goal is to minimise the total scheduling cost, which consists of lateness and inventory penalties.

The decision variables of the problem are x_{it}, the fraction of order i produced during time period t. Let $M = \{1, \ldots, m\}$ be the set of orders and $N = \{1, \ldots, n\}$ the set of time periods. Because the time periods are long enough to produce any order, and because the orders are non-pre-emptible, at most two variables among $x_{i1}, \ldots, x_{in}$ can be positive $\forall i \in M$, and when two variables are positive they must refer to adjacent time periods. This means that $\{x_{i1}, \ldots, x_{in}\}$ is SOS2 $\forall i \in M$. The problem can then be formulated as

$$\min \ \Sigma_{i \in M} \Sigma_{t \in N} c_{it} x_{it}$$

$$\Sigma_{i \in M} a_i x_{it} \leq b, \qquad t \in N, \tag{29}$$

$$\Sigma_{t \in N} x_{it} = 1, \qquad i \in M, \tag{30}$$

$$x_{it} \geq 0, \qquad i \in M, \ t \in N, \tag{31}$$

$$(x_{i1}, \ldots, x_{in}\} \text{ SOS2}, \qquad i \in M, \tag{32}$$

where c_{it} is the cost of executing order i completely in time period t; b is the capacity of the machine in a time period; and a_i is the capacity consumed by order i when it is fully executed.

By modifying the objective function coefficients appropriately, (30) can be replaced by

$$\sum_{t \in N} x_{it} \leq 1, \ i \in M \tag{33}$$

(see de Farias (1995).) In our polyhedral analysis we use the inequality formulation because, in it, the convex hull of the set of feasible solutions is a full-dimensional polytope.

4.2 Polyhedral results

Let F be the set of solutions, $PF = \text{conv}(F)$, and LPF the set of solutions of the LP relaxation, i.e. $LPF = \{x \in \Re^{mn} : x \text{ satisfies (29), (31), and (33)}\}$. We can obtain facet-defining inequalities for PF using the procedure illustrated in Section 3.2, i.e. we fix some of the variables, we obtain a facet-defining inequality for the projected polytope and we lift the inequality with respect to the variables that were fixed.

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

Example 2 Let $m=3$, $n=4$, $a_1=2$, $a_2=4$, $a_3=5$, and $b=5$. Fix $x_{it}=0\ \forall (i,t)\neq(1,1),(2,1)$. The inequality

$$2x_{11}+4x_{21}\leq 5 \tag{34}$$

defines a facet of the resulting polytope. We now lift (34) with respect to x_{13}. Let x_{13} be the lifting coefficient, i.e.

$$2x_{11}+\alpha_{13}x_{13}+4x_{21}\leq 5.$$

Because of the SOS2 constraint, when $x_{13}>0$, $x_{11}=0$. This means that α_{13} can be as large as 1 and therefore,

$$2x_{11}+x_{13}+4x_{21}\leq 5 \tag{35}$$

defines a facet of $PF\cap\{x\in\Re^{mn}: x_{it}=0\ \forall (i,t)\neq(1,1),(1,3),(2,1)\}$. If we now lift (35) with respect to x_{14}, x_{23}, x_{24}, x_{12}, x_{22}, and x_{31}, . . . , x_{24}, in the given order, we obtain

$$2x_{11}+x_{13}+x_{14}+4x_{21}+3x_{23}+3x_{24}\leq 5, \tag{36}$$

which defines a facet of *PF*. Inequality (36) cuts off the point $x_{11}=1$, $x_{21}=\frac{3}{4}$, $x_{23}=14$, which is a vertex of *LPF* that does not satisfy (32). □

We say that $I\subseteq M$ is an l-cover if $lb<\Sigma_{i\in I}\,a_i<(l+1)b$. Given an l-cover I, we denote

$$a_i^{(l)}=\max\{0,\ lb-\Sigma_{j\in I-\{i\}}\,a_j\},\ i\in M.$$

The inequality (36) can be generalised as follows.

Theorem 3 *Let $l\in[1,n-2]$ be an integer, and $I\subseteq M$ be an l-cover. Suppose $\exists i'\in I$ with $a_{i'}^{(l)}>0$. Let $T=\{t_1,\ldots,t_l\}\subset N$ be a set of adjacent indices, and define $T'=\{t\in N:|t-t'|\geq 2\forall t'\in T\}$. When $T'\neq\emptyset$*

,

$$\sum_{i\in I}a_i\sum_{t\in T}x_{it}+\sum_{i\in I}a_i^{(l)}\sum_{t\in T'}x_{it}\leq lb \tag{37}$$

is valid and facet-defining. □

For *PF* it does not suffice to fix variables at 0 to derive all its facets. Also, as shown in Example 3, it is not always possible to lift facet-defining inequalities of the projected polytope in any order.

Example 3 Consider the instance defined in Example 2. Fix $x_{13}=x_{14}=x_{23}=x_{24}=x_{32}=x_{33}=x_{34}=0$, $x_{12}=1$, and $x_{22}=\frac{4}{3}$. Start with

$$4x_{21}+5x_{31}\leq 5. \tag{38}$$

Lifting (38) with respect to x_{23}, x_{33}, x_{34}, x_{22}, x_{12}, x_{11}, x_{32}, x_{13}, x_{23}, x_{14}, and x_{24}, in the given order, we obtain

$$2x_{11}+2x_{12}+x_{14}+4x_{21}+4x_{22}+3x_{24}+5x_{31}+4x_{32}+4x_{33}+4x_{34}<10, \tag{39}$$

which defines a facet of *PF*.

Now, change the sequence above so that we lift (38) with respect to x_{12} before x_{22}. After lifting with respect to x_{23}, x_{34}, and x_{12}, we obtain

$$4x_{21}+5x_{31}+4x_{33}+4x_{34}\leq 5, \tag{40}$$

which defines a facet of $PF\cap\{x\in\Re^{12}: x_{11}=x_{13}=x_{14}=x_{24}=x_{32}=0,\text{ and }x_{22}=\frac{3}{4}\}$. However, we cannot lift (40) with respect to x_{22}. This is true because if α_{22} is the lifting coefficient,

$$4x_{21}+\alpha_{22}x_{22}+5x_{31}+4x_{33}+4x_{34}\leq 5+\tfrac{3}{4}\,\alpha_{22}.$$

Now,

$$x_{21}=x_{33}=x_{34}=0,\ x_{22}=x_{31}=1\Rightarrow\alpha_{22}\leq 0.$$

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

On the other hand

$$x_{22} = x_{31} = x_{34} = 0,\ x_{21} = x_{33} = 1 \Rightarrow \alpha_{22} \geq 4.$$

which is a contradiction. □

4.3 Computational results

We report computational experience using a branch-and-cut scheme without binary variables and a branch-and-bound algorithm without binary variables to solve the problem. In the branch-and-cut algorithm we use inequalities (37), plus two other families of inequalities described in de Farias, Johnson & Nemhauser (2000), as cuts.

Our computational experiments were performed over a set of 55 test problems randomly generated. In all test problems each SOS2 had 20 variables, and the number of sets ranged from 100 to 200 with intervals of 10. For a given problem size we tested 5 different instances. The size of the problems ranged from 120 constraints (not including non-negativity) and 2000 variables to 220 constraints and 4000 variables Although this study was based on a real application, described in detail in Gue *et al.* (1997), and solved by branch-and-bound, it was necessary to use random data in the computation since the real data was no longer available. However, as far as possible, we based the random data on the type of data in the application. For other details, such as node selection, branching and separation strategies, we refer to de Farias, Johnson & Nemhauser (2000). We used an IBM RS6000/590 to run our test problems and MINTO 3.0 as branch-and-bound algorithm and CPLEX 6.0 as LP solver. We limited the size of the branching tree to 50,000 nodes.

Of the 55 problems tested, the pure branch-and-bound approach could not find a feasible solution for 19 problems, and it could not find an optimal solution or prove optimality for 5 of the remaining problems. With the branch-and-cut approach, all 55 problems were solved to proven optimality.

Table 2 has, for each problem size, the average CPU time in seconds with and without cut generation to obtain a proven optimal solution and the first feasible solution. Table 3 has, for each problem size, the average number of nodes processed with and without cut generation, as well as the number of cuts generated, to obtain a proven optimal solution and the first solution. For those problems that were not solved to optimality we included in the averages of Table 3 the number of nodes processed (50,000) when the algorithm was halted, as well as the computational time spent so far in the time averages of Table 2. The same was done for the averages relative to the first solution for those problems for which

Table 2 Time

			OPTIMALITY		1st SOLUTION	
SETS	CONS	VARS	TIME NO CUTS	TIME W/CUTS	TIME NO CUTS	TIME W/CUTS
100	120	2,000	2,833	90	2,590	56
110	130	2,200	3,312	2,290	2,999	1,063
120	140	2,400	2,397	83	2,241	73
130	150	2,600	2,311	104	2,204	65
140	160	2,800	2,314	227	2,278	206
150	170	3,000	956	73	457	50
160	180	3,200	2,998	731	2,913	554
170	190	3,400	1,967	158	1,561	108
180	200	3,600	4,261	345	4,192	135
190	210	3,800	3,160	1,045	3,155	599
200	220	4,000	1,310	43	982	37
Total			27,819	5,189	25,572	2,946

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

Table 3 Nodes and cuts

	OPTIMALITY			1st SOLUTION		
SETS	NODES NO CUTS	NODES W/CUTS	CUTS	NODES NO CUTS	NODES W/CUTS	CUTS
100	38,967	881	138	32,351	456	135
110	33,933	15,429	641	31,251	7,151	510
120	30,523	1,067	130	26,662	898	101
130	27,679	1,147	151	25,094	654	121
140	24,191	1,735	143	23,674	1,499	96
150	18,180	792	117	8,243	421	110
160	29,708	4,645	201	28,020	3,239	118
170	21,707	1,462	138	20,532	900	95
180	35,388	2,424	459	33,599	875	250
190	25,889	5,515	232	25,791	2,817	169
200	14,260	344	107	10,437	276	103
Total	300,425	35,441	2,457	265,654	19,186	1,808

branch-and-bound failed to deliver a feasible solution. The most important inequality in our tests was (37). Approximately, 60% of the cuts generated were (37).

From Tables 2 and 3 we can see that with the branch-and-cut algorithm, the computational time was reduced by an overall factor of 5.3, and the number of nodes processed was reduced by a factor of 8.5.

5 Conclusions and further research

This paper provides an alternative approach for solving linear programs with additional combinatorial constraints. We avoid the operations research community approach of using binary variables, leading to mixed integer programs. Nevertheless, we use bounds based on linear programming relaxations, which is the strength of the operations research approach and makes it possible to solve optimisation problems efficiently. Typically, constraint programming approaches are designed to solve feasibility problems and have difficulty with optimisations. The computational results we have presented for the generalised assignment problem, and the computational results in de Farias, Johnson & Nemhauser (forthcoming) for cardinality-constrained linear programs are encouraging and seem to indicate that our approach is better than mixed-integer programming for many instances. This research has just begun, and substantially more work is needed in areas such as pre-processing, cuts and separation to determine the viability of dealing with combinatorial constraints directly.

Acknowledgments

We would like to thank two anonymous referees, whose suggestions helped to improve the quality of this paper.

References

Balas, E, 1979, “Disjunctive programming” in Hammer, PL, Johnson, EL and Korte, BH (eds) *Annals of Discrete Mathematics 5: Discrete Optimization* North Holland Publishing Company.

Beale, EML, 1979, “Branch-and-bound methods for mathematical programming systems” in Hammer, PL, Johnson, EL and Korte, BH (eds) *Annals of Discrete Mathematics 5: Discrete Optimization* North Holland Publishing Company.

Beale, EML, 1980, “Branch-and-bound methods for numerical optimization” in Barritt, MM and Wishart, D (eds) *COMPSTAT 80: Proceedings in Computational Statistics* 11–20, Physica Verlag.

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press

Beale, EML, 1985, "Integer programming" in Schittkowski, K (ed.) *Computational Mathematical Programming* NATO ASI Series, Springer-Verlag.

Beale, EML and Tomlin, JA, 1970, "Special facilities in a general mathematical programming system for non-convex problems using ordered sets of variables" in Lawrence, J (ed.) *Proceedings of the Fifth Intenational Conference on Operations Research* 447–454, Tavistock Publications.

Beaumont, N, 1990, "An algorithm for disjunctive programming" *European Journal of Operational Research* **48** 362–371.

Biegler, LT, Grossmann, IE and Westerberg AW, 1997, *Systematic Methods of Chemical Process Design* Prentice Hall, Inc.

Bienstock, D, 1996, "Computational study of a family of mixed-integer quadratic programming problems" *Mathematical Programming* **74** 121–140.

Brimberg, J, and Love, RF, 1998, "Solving a class of two-dimensional uncapacitated location-allocation problems by dynamic programming" *Operations Research* **46** 702–709.

Crowder, H, Johnson, EL and Padberg, MW, 1983, "Solving large scale zero-one linear programming problems" *Operations Research* **31** 803–834.

Dantzig, GB, 1960, "On the significance of solving linear programming problems with some integer variables" *Econometrica* **28** 30–44.

de Farias, IR, 1995, "A polyhedral approach to combinatorial complementarity problems" *Ph.D. Thesis*, School of Industrial and Systems Engineering, Georgia Institute of Technology.

de Farias, IR, 2001, "A family of facets for the uncapacitated p-median polytope" *Operations Research Letters* **28** 161–167.

de Farias, IR, Johnson, EL and Nemhauser, GL, 1998, "Facets of the complementarity knapsack polytope" LEC-98-08, School of Industrial and Systems Engineering, Georgia Institute of Technology.

de Farias, IR, Johnson, EL and Nemhauser GL, 2000, "A generalized assignment problem with special ordered sets: a polyhedral approach" *Mathematical Programming* **89** 187–203.

de Farias, IR, Johnson, EL and Nemhauser, GL, forthcoming, "A branch-and-cut approach to cardinality constrained optimization problems" in preparation.

de Farias, IR and Nemhauser, GL, 2000, "A family of inequalities for the generalized assignment polytope". To appear in *Operations Research Letters*.

Gomory, RE, 1958, "Outline of an algorithm for integer solutions to linear programs" *Bulletin of the American Mathematical Society* **64** 275–278.

Gomory, RE, 1965, "On the relation between integer and non-integer solutions to linear programs" *Proceedings of the National Academy of Science* **53** 260–265.

Grotschel, M, Junger, M and Reinelt, G, 1984, "A cutting plane algorithm for the linear ordering problem" *Operations Research* **32** 1195–1220.

Gue, KR, Nemhauser, GL and Padron, M, 1997, "Production scheduling in multiprocessor flowshops" *IIE Transactions* **29** 391–398.

Healy, WC, 1964, "Multiple-choice programming (a procedure for linear programming with zero-one variables)" *Operations Research* **12** 122–138.

Hooker, JN and Osorio, MA, 1999, "Mixed logical/linear programming," *Discrete Applied Mathematics* **96** 395–442.

Ibaraki, T, 1973, "The use of cuts in complementary programming" in Balas, E, Cottle, RW, Hu, TC and Kirby, MJL (eds) *Contributions to Programming and Applications*, ORSA.

Jeroslow, RG, 1978, "Cutting planes for complementarity constraints" *SIAM Journal of Control and Optimization* **16** 56–62.

Markowitz, HM and Manne, AS, 1957, "On the solution of discrete programming problems" *Econometrica* **25** 84–110.

Nauss, RM, 1978, "The 0-1 knapsack problem with multiple-choice constraints" *European Journal of Operations Research* **2** 125–131.

Nemhauser, GL and Wolsey, LA, 1988, *Integer Programming and Combinatorial Optimization*, John Wiley and Sons.

Padberg, M and Rinaldi, G, 1987, "Optimization of a 532 city symmetric traveling salesman problem by branch and cut" *Operations Research Letters* **6** 1–7.

Raman, R and Grossmann, IE, 1993, "Symbolic integration of logic in MILP branch-and-bound methods for the synthesis of process networks" *Annals of Operations Research* **42** 169–191.

Savelsbergh, MWP, personal communication.

Wolsey, LA, 1976, "Facets and strong valid lnequalities for integer programs" *Operations Research* **24** 267–372.

https://doi.org/10.1017/S0269888901000030 Published online by Cambridge University Press