TIGHT REPRESENTATION OF LOGICAL CONSTRAINTS AS CARDINALITY RULES

John N. Hooker, Hong Yan

EDRC 70-06-92

Tight Representation of Logical Constraints as Cardinality Rules *

J. N. HOOKER
H. YAN
Graduate School of Industrial Administration
Carnegie Mellon University, Pittsburgh, PA 15213 USA

June 1992

Abstract

We address the problem of finding a "tight" representation of complex logical constraints in a mixed integer programming model by describing a convex hull representation of cardinality rules. A cardinality rule asserts that if at least k of the propositions $A_1, \ldots, A_m$ are true, then at least l of the propositions $B_1, \ldots, B_n$ are true.

As rule-based systems and other types of logic modeling grow in popularity, logical rules and propositions can play an increasingly important role in mathematical programming models. Such simple logical constraints as "if A is produced, then either B or C must be produced" have long been a part of mathematical programming. But much more complex logic models are now being formulated, and they can also be embedded in mathematical programming models.

A logic model is a description of a problem expressed in some logical formalism, usually propositional or predicate logic, from which facts about its solution can be deduced. Expert systems and other rule-based systems are logic models, as are programs written in the logical programming language PROLOG. It is useful to incorporate a logic model into a mathematical programming model when neither model alone adequately describes reality, or when the logic model contains constraints or heuristic rules that ease the

*The first author is partially supported by AFOSR grant 91-0287 and ONR grant N00014-92-J-1028. Both authors are partially supported by NSF grant 1-55093.

solution of the mathematical model by reducing the number of alternatives that must be examined.

When logical constraints become a significant component of mathematical programming models, the quality of their representation becomes an important issue. We wish to address this problem by showing how best to represent a particular type of logical formula, namely a *cardinality rule*, in a mixed integer programming (MIP) model. We focus on cardinality rules because we have observed that logical constraints take this form in a large variety of applications. In fact McKinnon and Williams [11] use cardinality conditions as a basic form in which to express logical constraints.

A cardinality rule has the form,

> If at least (or at most) k of the propositions $A_1, \ldots, A_m$ are true, then at least (or at most) l of the propositions $B_1, \ldots, B_n$ are true.

An *elementary* cardinality rule is one in which each A_i and each B_j is a *literal* (an atomic proposition or its negation) and no atomic proposition occurs more than once. The rules one typically finds in expert systems,

> if $A_1, \ldots, A_m$ are all true, then B is true,

are special cases of elementary cardinality rules, as are *cardinality clauses* of the form,

> at least (or at most) k of $A_1, \ldots, A_m$ are true.

The usual method for representing logical conditions in an MIP model has two stages. They are first rewritten as a conjunction of logical "clauses," which is to say in conjunctive normal form (CNF). A *clause* is a disjunction of literals, such as,

$$x_1 \vee \neg x_2 \vee x_3,$$

where $\neg$ means "not," and $\vee$ means "or." Each clause is then written as an inequality in 0-1 variables, which for this example is,

$$x_1 + (1 - x_2) + x_3 \geq 1,$$

where x_j is interpreted as true when $x_j = 1$ and false when $x_j = 0$.

Unfortunately this is typically a very "loose" representation of the logical conditions, in the sense that its linear relaxation (which replaces $x_j \in \{0, 1\}$ with $0 \leq x_j \leq 1$) describes a polytope that has many fractional extreme

points. This complicates the solution of the model, since most solution techniques make essential use of the linear relaxation.

The difficulty with CNF representation is not that the individual clauses are poorly represented. In fact, each clause receives the tightest possible representation, namely a *convex hull representation.* This is a set of 0-1 inequalities whose linear relaxation describes the convex hull of the 0-1 points satisfying them. Rather, the difficulty is that there are typically a large number of clauses in CNF. This results in a loose representation of the formula as a whole, even though each individual clause is tightly represented. (R. Jeroslow discusses this principle in [9].) This problem is particularly acute for a cardinality rule, since the number of clauses in its CNF equivalent grows exponentially with the rule's length if no new variables are added (and is apparently quite large even if new variables are added).

Thus when logical constraints occur or can be expressed in the form of elementary cardinality rules, it is far better to give each a convex hull representation directly than to convert it to CNF first. The resulting description is no shorter than the CNF equivalent, because it contains all the inequalities that appear in CNF. But it avoids the loose representation that results from using *only* the inequalities in the CNF representation.

We will therefore state an algorithm that generates for any elementary cardinality rule a set of 0-1 inequalities that provide a convex hull representation of it. Our main result is that these inequalities in fact describe the facets of the convex hull.

Several authors have contributed to the inequality representation of logical formulas in CNF. Tseitin [12] showed how to convert any formula of propositional logic to CNF in linear time by adding new variables. Cook [3, 4], Dantzig [5], Blair *et al.* [2], and Williams [13, 14, 15] discussed the use of 0-1 inequalities to represent logical constraints in CNF, and Karp [10] used them to show that integer programming is NP-hard. The shortest such representation, however, was recently proposed by Wilson [16].

Some attention has also been paid to the representation of cardinality clauses (recall that these are a special case of elementary cardinality rules). Hadjiconstantinou and Mitra [6] described an algorithm for the automatic conversion of logical formulas, including cardinality clauses, into inequality form. But they do not consider the tightness of the representation. Hooker [8] described a generalized resolution procedure that generates all the undominated implications of a set of cardinality clauses. This can lead to a tighter representation because it in effect generates valid cuts. But it can be

computationally expensive if carried to completion on a large set of clauses. In research that proceeded concurrently with ours, Araque and Chandru [1] found a convex hull representation of cardinality clauses. But their work has a different focus, because it does not consider cardinality rules in general, and it describes the polyhedron associated with a set covering formulation of the problem, which has twice the dimension of the polyhedron we study.

We begin in Section 1 by augmenting propositional logic with cardinality rules to obtain "cardinality logic." Since our convex hull representation is designed for elementary cardinality rules, we show how to convert any formula of cardinality logic to a conjunction of elementary cardinality rules (*cardinality normal form*) in linear time. We also show that the CNF of an elementary cardinality rule grows exponentially with the length of the rule, when no variables are added. When additional variables are used, CNF is polynomial but still quite large using what is apparently the best known conversion.

Section 2 states our algorithm for generating the convex hull description of a cardinality rule and proves its correctness. In Section 3 we use an example to illustrate its application.

1 Cardinality Rules

We begin by augmenting propositional logic with cardinality rules. We then show how to write any formula of cardinality logic in cardinality normal form. Finally we discuss what is involved in the reduction of elementary cardinality rules to CNF.

1.1 Cardinality Logic

A cardinality rule is written,

$$(A_1, \ldots, A_m)_k \Rightarrow (B_1, \ldots, B_n)_l, \tag{1}$$

and is read, "if at least k of $A_1, \ldots, A_m$ are true, then at least l of $B_1, \ldots, B_n$ are true." We assume $m \geq k \geq 0$, $n \geq 0$, and $l \geq 1$. The phrase "at least" can be replaced with "at most" by writing,

$$(\neg A_1, \ldots, \neg A_m)_{m-k} \Rightarrow (\neg B_1, \ldots, \neg B_n)_{n-l}.$$

Ordinary rules have $m = k$ and $l = 1$ in (1). When $m = 1$ or $n = 1$ we will omit the parentheses, so that $(A)_1 \Rightarrow (B)_1$ is written $A \Rightarrow B$.

When the antecedent is a tautology, which is to say $k = 0$, (1) asserts that at least l of $B_1, \ldots, B_n$ are true. In such cases we can assume without loss of generality that $m = 0$. When the consequent is a contradiction, which is to say $l > n$, (1) asserts that fewer than k of $A_1, \ldots, A_m$ are true. In such cases we can assume that $n = 0$ and $l = 1$. We therefore suppose that every cardinality rule satisfies a) either $m \geq k \geq 1$ or $m = k = 0$, and b) either $n \geq l \geq 1$ or $(n, l) = (0, 1)$.

Propositional logic is recursively defined to consist of *atomic propositions* $x_1, x_2, \ldots$, which are regarded as formulas, plus all formulas of the forms $\neg A$, $A \wedge B$, $A \vee B$, $A \supset B$, and $A \equiv B$, where A and B are formulas. Here $\wedge$ means "and," $\vee$ means "or," $\supset$ means "implies," and $\equiv$ means "is equivalent to." Implication and equivalence are defined by,

$$(A \supset B) \quad =_{def} \quad (\neg A \vee B) \tag{2}$$

$$(A \equiv B) \quad =_{def} \quad ((A \supset B) \wedge (B \supset A)). \tag{3}$$

Cardinality logic is recursively defined to consist of all the above formulas and all formulas of the form (1), where $A_1, \ldots, A_m$ and $B_1, \ldots, B_n$ are formulas of cardinality logic.

1.2 Cardinality Normal Form

We now show how to convert any formula of cardinality logic to cardinality normal form in linear time. If the formulas to be represented are already elementary cardinality rules, this conversion is not necessary, and one can proceeed directly to the linear inequality representation (Section 2).

We first review conversion to conjunctive normal form (CNF) in ordinary propositional logic. This can be done by a) using (2) and (3) to remove all occurrences of $\supset$ and $\equiv$; b) using De Morgan's laws,

$$\begin{aligned} \neg(A \wedge B) &\equiv (\neg A \vee \neg B) \\ \neg(A \vee B) &\equiv (\neg A \wedge \neg B), \end{aligned}$$

and the equivalence $\neg\neg A \equiv A$ to absorb all negations into literals; and c) using the distribution law,

$$A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C),$$

to transform the result to a conjunction of clauses.

It is well known that if no new variables are used, CNF can grow exponentially with the size of the formula. For instance, a formula of the form

$$(x_1 \wedge y_1) \vee \ldots \vee (x_n \wedge y_n) \tag{4}$$

translates to the conjunction of 2^n clauses of the form $A_1 \vee \ldots \vee A_n$, where each A_j ranges over x_j and y_j. But by adding new variables, conversion to CNF can always be accomplished in linear time. Using Wilson's conversion [16], we can write formula (4), for instance, as the conjunction,

$$(z_1 \vee \ldots \vee z_n) \wedge \bigwedge_{j=1}^{n} (\neg z_j \vee x_j) \wedge (\neg z_j \vee y_j). \tag{5}$$

Each new variable z_i represents a clause $x_i \vee y_i$. The clauses $\neg z_j \vee x_j$ and $\neg z_j \vee y_j$ encode implications $z_j \supset x_j$ and $z_j \supset y_j$, respectively. (5) is a reformulation of (4) in the sense that a) any assignment of truth values to atomic propositions that satisfies (5) also satisfies (4), and b) for any assignment of values to $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ satisfying (4), there is some assignment of values to $z_1, \ldots, z_n$ for which (5) is also satisfied.

We will use this same principle in cardinality logic. Conversion to cardinality normal form is somewhat complex in general because cardinality formulas can be embedded in cardinality formulas. But it should ordinarily be straightforward in practice because such embedding is probably unusual. In any case the conversion algorithm, below, can be automated.

Step 1. Use (2) and (3) to remove occurrences of $\supset$ and $\equiv$.

Step 2. Use De Morgan's laws and the following to absorb all negations into literals:

$$\neg[(x_1 \ldots x_m)_k \Rightarrow (y_1 \ldots y_n)_l] \equiv [\Rightarrow (x_1 \ldots x_m)_k] \wedge [(y_1 \ldots y_n)_l \Rightarrow] \tag{6}$$

Let F be the resulting formula.

Step 3. F is a conjunction of one or more formulas. Let G be a conjunct of F that is not an elementary cardinality rule. If there is no such conjunct, stop. If G is a disjunction $B_1 \vee \ldots \vee B_n$ with $n \geq 2$, write G as a cardinality rule $\Rightarrow (B_1, \ldots, B_n)_1$.

Step 4. If G is an elementary cardinality rule except that some atomic proposition A occurs more than once, replace each occurrence of A

after the first with a distinct variable z_p that does not occur in F. Rewrite G as the conjunction of the resulting cardinality rule with rules $A \Rightarrow z_p$ for each z_p that occurs in the antecedent and with rules $z_p \Rightarrow A$ for each z_p that occurs in the consequent. Let F be the resulting formula and go to Step 3.

Step 5. G is a cardinality rule of the form,

$$(C_1, \ldots, C_p, c_1, \ldots, c_{p'})_r \Rightarrow (D_1, \ldots, D_q, d_1, \ldots, d_{q'})_s, \qquad (7)$$

where $c_1, \ldots, c_{p'}$ and $d_1, \ldots, d_{q'}$ are literals. Replace the conjunct G of F with the conjunction,

$$(t_1, \ldots, t_p, c_1, \ldots, c_{p'})_r \Rightarrow (u_1, \ldots, u_q, d_1, \ldots, d_{q'})_s \\ \wedge \bigwedge_{i=1}^{p}(C_i \Rightarrow t_i) \wedge \bigwedge_{i=1}^{q}(u_i \Rightarrow D_i),$$

where $t_1, \ldots, t_p$ and $u_1, \ldots, u_q$ are new variables that do not occur in F.

Step 6. For $i = 1, \ldots, q$ do the following:

- If D_i is a cardinality rule of the form (1) with $m + n > 1$, replace $u_i \Rightarrow D_i$ with

$$(A_1, \ldots, A_m)_k \Rightarrow (B_1, \ldots, B_n, w_1, \ldots, w_l)_l \qquad (8) \\ \wedge \bigwedge_{j=1}^{l}(w_j \Rightarrow \neg u_i)$$

  where $w_1, \ldots, w_l$ are new variables.

- If D_i is a conjunction $B_1 \wedge \ldots \wedge B_n$ of 2 or more formulas, replace $u_i \Rightarrow D_i$ with

$$\bigwedge_{j=1}^{n}(u_i \Rightarrow B_j).$$

- If D_i is a disjunction $B_1 \vee \ldots \vee B_n$ of 2 or more formulas, replace $u_i \Rightarrow D_i$ with the cardinality rule,

$$\Rightarrow (\neg u_i, B_1, \ldots, B_m)_1.$$

Step 7. For $i = 1, \ldots, p$ do the following:

- If C_i is a cardinality rule of the form (1) with $m+n>1$, replace $C_i \Rightarrow t_i$ with

$$\neg[(A_1,\ldots,A_m,v_1,\ldots,v_k)_k \Rightarrow (B_1,\ldots,B_n,w_1,\ldots,w_{n-l+1})_{n+1}] \\ \wedge \ \bigwedge_{j=1}^{k}(v_j \Rightarrow t_i) \ \wedge \ \bigwedge_{j=1}^{n-l+1}(\neg w_j \Rightarrow t_i),$$

  where $v_1,\ldots,v_k$ and $w_1,\ldots,w_{n-l+1}$ are new variables. Then apply (6) and De Morgan's laws to move negations inside.

- If C_i is a conjunction $A_1 \wedge \ldots \wedge A_m$ of 2 or more formulas, replace $C_i \Rightarrow t_i$ with the cardinality rule,

$$\Rightarrow (\neg A_1,\ldots,\neg A_m,t_i)_1.$$

  Apply (6) and De Morgan's laws to move negations inside.

- If C_i is a disjunction $A_1 \vee \ldots \vee A_m$ of 2 or more formulas, replace $C_i \Rightarrow t_i$ with,

$$\bigwedge_{j=1}^{m}(A_i \Rightarrow t_i).$$

Step 8. Go to Step 3.

This algorithm runs in time that increases at worst linearly with the number of symbols in the original formula. The argument for this is quite similar to that for ordinary propositional logic.

1.3 Conversion of Cardinality Rules to CNF

We first show that elementary cardinality rules have an exponential CNF expansion. For ease of exposition we will show this for *Horn cardinality rules* in particular, which have the form,

$$(x_1,\ldots,x_m)_k \Rightarrow y. \qquad (9)$$

Theorem 1 *No CNF formula equivalent to (9) whose variables are in* $\{x_1,\ldots,x_m,y\}$ *has fewer than* $\binom{m}{k}$ *clauses.*

Proof. Let F be any CNF equivalent. Let $(x^*,y^*) = (x_1^*,\ldots,x_m^*,y^*) \in \{0,1\}^{(m+1)}$ be a *minimal violator* of (9) if it violates (9) but would satisfy it if any x_j^* equal to 1 were switched to 0. There are $\binom{m}{k}$ minimal violators,

since each has exactly k x_j^*'s equal to 1, and $y^* = 0$. Every minimal violator (x^*, y^*) must violate some clause C_{x^*} in F. We show that every C_{x^*} is necessarily distinct and that none implies another, from which the theorem follows.

Since (x^*, y^*) violates C_{x^*}, C_{x^*} cannot contain the positive literal x_j when $x_j^* = 1$. But since (x^*, y^*) is a ***minimal*** violator, C_{x^*} must contain x_j for each $x_j^* = 0$. It follows that the positive literals in C_{x^*} are precisely those for which $x_j^* = 0$. Thus every C_{x^*} is distinct and no one implies another. □

When additional variables are used, there is a polynomial CNF expansion, but it is still long. We will show this using the pigeon hole principle, which Haken used to prove that the resolution method of theorem proving has exponential complexity [7]. To convert (9) to CNF, we create new variables z_{ij} for $i = 1, \ldots, m$ and $j = 1, k-1$. The CNF equivalent is the conjunction of the following clauses.

$$\neg x_i \vee y \vee \bigvee_{j=1}^{k-1} z_{ij}, \quad i = 1, \ldots, m, \tag{10}$$

$$\neg z_{ij} \vee \neg z_{i'j}, \quad \text{all } i, i' \in \{1, \ldots, m\}\ (i \neq i'),\ j = 1, \ldots, k-1. \tag{11}$$

We interpret the formula (9) as saying that if we have k or more pigeons (i.e., k or more x_j's are true), then we cannot put them in $k-1$ holes with at most one per hole (y is true). In (10)-(11), $z_{ij} = 1$ is interpreted as saying that pigeon j is put in hole i. The clauses (10) say that every pigeon is put into a hole, or else y is true. The clauses (11) say that no 2 pigeons are put into the same hole. Thus (10)-(11) force y to be true precisely when we have k or more pigeons, just as (9) does.

The above method of producing a CNF equivalent generates $m(k-1)$ additional variables and $m + (1/2)(k-1)m(m-1)$ clauses. We are aware of no shorter conversion.

2 Convex Hull Representation of Cardinality - Rules

In this section, we describe the facets of the convex hull of the feasible points of an elementary cardinality rule. We also state an algorithm for generating the facets.

2.1 Description of the Facets

Without loss of generality we consider rules of the form

$$(x_1, \ldots, x_m)_k \Rightarrow (y_1, \ldots, y_n)_l. \tag{12}$$

If a rule has a negated literal $\neg x_i$ or $\neg y_j$, we can replace every occurrence of x_i or y_j in the facet-defining inequalities with $1 - x_i$ or $1 - y_j$, respectively.

Denote by S the set of feasible points satisfying (12), and by $conv(S)$ the convex hull of S. We begin with a negative result.

Lemma 1 *When $m > k$ and $l \geq 2$, (12) is equivalent to no single 0-1 linear inequality.*

Proof: Because of the symmetry of all x_i and of all y_i in (12), if there is an equivalent linear inequality, there is one of the form:

$$-a(x_1 + \ldots + x_m) + b(y_1 + \ldots + y_n) \geq -c \tag{13}$$

For brevity we will write (13) as follows, where e is a row vector of ones.

$$-aex + bey \geq -c.$$

Since $(ex, ey) = (k, l-1)$ does not satisfy (12), we have

$$-ak + b(l-1) < -c. \tag{14}$$

Since $(ex, ey) = (k+1, l)$ satisfies (12), we have

$$-a(k+1) + bl \geq -c. \tag{15}$$

(14) and (15) imply

$$a < b. \tag{16}$$

Since $l \geq 2$, (14) implies

$$b + c < ak. \tag{17}$$

From (16),

$$ak < a(k-1) + b. \tag{18}$$

Combining (17) and (18),

$$c < a(k-1). \tag{19}$$

On the other hand, since $ex = k - 1$ and $ey = 0$ satisfy (12), we have

$$-a(k-1) \geq -c,$$

which contradicts (19). □

When $k = m$ or $l = 1$, however, we have an equivalent 0-1 linear inequality.

Lemma 2 *If $k = m$, (12) is equivalent to the 0-1 linear inequality,*

$$-lex + ey \geq l(1-m). \tag{20}$$

Proof: If (12) is true, then either $ey \geq l$ or $ex < m$ (or both). It is easily checked that (20) is satisfied in either case. Also, if (12) is false, then $ex = m$ and $ey < l$, which violates (20). □

Similarly, it is easily verified that:

Lemma 3 *If $l = 1$, (12) is equivalent to,*

$$-ex + (1+m-k)ey \geq 1-k. \tag{21}$$

□

The following lemma gives an inequality which will start our description of the convex hull of (12).

Lemma 4 *If both of the following hold: a) $k \geq 2$ or $m = k = 1$ or $m = k = 0$, b) $n > l$ or $n = l = 1$ or $(n, l) = (0, 1)$, then the 0-1 linear inequality*

$$-lex + (1+m-k)ey \geq l(1-k) \tag{22}$$

describes a facet of $conv(S)$.

Proof: It is easily checked that any point satisfying (12) satisfies (22) for all the cases. So we only need to find $m + n$ affinely independent points (x, y) in S such that

$$-lex + (1+m-k)ey = l(1-k)$$

We first consider the case $k \geq 2$ and $n > l$. We construct a matrix D whose rows are $m + n$ affinely independent points, such that the first m points satisfy $ex = k - 1$, $ey = 0$ and the other n points satisfy $ex = m$, $ey = l$.

$$D = \begin{pmatrix} A_1 & O_1 & O_2 \\ A_2 & I_1 & O_3 \\ C_1 & B_1 & O_4 \\ C_2 & B_2 & I_2 \end{pmatrix}.$$

Here A_1 is a $k \times k$ matrix of all ones except for zeros on the diagonal. A_2 is a $(m-k)\times k$ 0-1 matrix with $k-2$ ones in each row. I_1 is a $(m-k)\times(m-k)$ identity matrix. O_1, O_2 and O_3 are zero matrices of the appropriate sizes.

B_1 is a $(l+1) \times (l+1)$ matrix of ones except for zeros on the diagonal. O_4 is a $(n-l-1) \times n$ zero matrix. B_2 is a $(n-l-1) \times (l+1)$ 0-1 matrix with exactly $l-1$ zeros in each row. And I_2 is a $(n-l-1) \times (n-l-1)$ identity matrix. C_1 and C_2 are matrices of the appropriate sizes containing all ones.

To show that these $m+n$ points are affinely independent, it suffices to show that $det(D) \neq 0$. It is clear that

$$det(D) = det(A_1)det(B_1)$$

To compute $det(A_1)$, add all other rows of A_1 to the first row and then subtract all other rows from the new first row divided by $(k-1)$. Then it is clear that $det(A_1) = (-1)^{k-1}(k-1)$. Similarly, $det(B_1) = (-1)^l l$. Therefore, we get

$$det(D) = (-1)^{k-1}(k-1)(-1)^l l.$$

The first case follows.

We will use appropriate portions of matrix D to exhibit the needed points for other cases. Their affine independence is obvious.

When $m = k = 1$ and $n > l$, the matrix

$$\begin{pmatrix} 0 & 0 & 0 \\ 1 & B_1 & O_4 \\ 1 & B_2 & I_2 \end{pmatrix}$$

gives the $n+1$ points.

When $m = k = 0$ and $n > l$, the matrix

$$\begin{pmatrix} B_1 & O_4 \\ B_2 & I_2 \end{pmatrix}$$

gives the n points.

When $k \geq 2$ and $n = l = 1$, the matrix

$$\begin{pmatrix} A_1 & O_1 & 0 \\ A_2 & I_1 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

gives the $m + 1$ points.

When $k \geq 2$ and $n = 0$, $l = 1$, the matrix

$$\begin{pmatrix} A_1 & O_1 \\ A_2 & I_1 \end{pmatrix}$$

gives the m points.

All others are trivial cases. Thus the lemma follows. □

To describe the entire set of facets, we first show that the inequality (22) is the only facet-defining inequality,

$$ax + by \geq c, \tag{23}$$

in which each $a_j \neq 0$ and each $b_j \neq 0$. We then obtain the remaining facets recursively by showing that they are the facets of simpler cardinality rules. To do the latter we exploit the facts a) that the remaining facets are facets of the convex hull's projections onto lower dimensional spaces, and b) that each of these projections is itself the convex hull description of a simpler cardinality rule.

Lemma 5 *Assume (23) defines a facet of conv(S). If $a_i \neq 0$ for all i and $b_j \neq 0$ for all j, then (23) is a nonzero scalar multiple of (22).*

Proof: We first show $a_i < 0$ for all i and $b_j > 0$ for all j. Since (23) defines a facet of $conv(S)$, it must contain a set T of $m + n$ affinely independent points in S. For any $i \in \{1, \ldots, m\}$ we know that some $(\overline{x}, \overline{y}) \in T$ has $\overline{x}_i = 1$. Otherwise the facet is defined by $x_i \geq 0$. Since $(\overline{x}, \overline{y})$ satisfies (12), it must either falsify the antecedent of (12) or satisfy the consequent. In either case a point $(x', \overline{y})$ identical to $(\overline{x}, \overline{y})$ except that $x'_i = 0$ also satisfies (12), so that $ax' + b\overline{y} \geq c$. Subtracting $a\overline{x} + b\overline{y} = c$ from this, we get $a_i < 0$.

Now consider b_j. Similarly, some $(\overline{x}, \overline{y}) \in T$ has $\overline{y}_j = 0$. Since $(\overline{x}, \overline{y})$ satisfies (12), it must either falsify the antecedent of (12) or satisfy the consequent. In either case a point $(\overline{x}, y')$ identical to $(\overline{x}, \overline{y})$ except that $y'_j = 1$ also satisfies (12), so that $a\overline{x} + by' \geq c$. Subtracting $a\overline{x} + b\overline{y} = c$ from this, we get $b_j > 0$.

Without loss of generality, assume $a_1 \leq a_2 \leq \cdots \leq a_m < 0$, and $0 < b_1 \leq b_2 \leq \cdots \leq b_n$.

Second, we show that for any $(x, y) \in T$, $ey \geq l$ implies $ex = m$. Consider a point $(\overline{x}, \overline{y}) \in T$ such that $e\overline{y} \geq l$. If $e\overline{x} < m$, (that is, $\overline{x}_t = 0$ for some t), then a point $(x', \overline{y})$ identical to $(\overline{x}, \overline{y})$ except $x'_t = 1$ satisfies (12). Thus $ax' + b\overline{y} \geq c$, which implies $a_t > 0$, a contradiction. With similar arguments, it is easily verified that points $(x, y) \in T$ with $ex = m$ satisfy $ey = l$.

We can also show that for any $(x, y) \in T$, $ey < l$ implies $ey = 0$. Consider a point $(\overline{x}, \overline{y}) \in T$ with $e\overline{y} < l$. We must have $e\overline{x} \leq k - 1$. And if $e\overline{x} < k - 1$, we have as above that $a_t > 0$ for some t. Thus $e\overline{x} = k - 1$. But if $e\overline{x} = k - 1$, it must be that $e\overline{y} = 0$. Otherwise, if $\overline{y}_t = 1$ for some t, consider a point $(\overline{x}, y')$ identical to point $(\overline{x}, \overline{y})$ but with $y'_t = 0$. In a way similar to the above, we will have $b_t < 0$, a contradiction.

It follows that the points in T can be partitioned into subsets T_1 and T_2, with $ex = k - 1$ and $ey = 0$ for all $(x, y) \in T_1$, and $ex = m$ and $ey = l$ for all $(x, y) \in T_2$. Furthermore, we can see as follows that there are m points in T_1 and n points in T_2. It is clear that T_1 has at most m points, since $ey = 0$ and they are independent. In T_2, there are at most n points since $ex = m$ and they are independent. Since T has $m + n$ points, the claim follows.

Now we show that $a_i = \alpha \ \forall i$ and $b_j = \beta \ \forall j$. First consider the points in T_1. Set $a_m = \alpha$. We will build recursively a set $Q \subseteq \{1, \ldots, m\}$ of indices such that $a_i = \alpha$ for all $i \in Q$. Initially $Q = \{m\}$, and we will augment Q until $Q = \{1, \ldots, m\}$.

At each step of the recursion we have $Q = \{q, \ldots, m\}$. Let T_1^q be the set of points in T_1 satisfying $x_i = 1$ for some $i \in \{q, \ldots, m\}$. Since T_1 contains m independent points with $ey = 0$, T_1^q must be nonempty. Let $(\overline{x}, 0) \in T_1^q$ be a point satisfying $\overline{x}_t = 0$ for some $t < q$. We suppose for the moment that such a point exists. Let the point $(x', 0)$ be identical to $(\overline{x}, 0)$ except $x'_t = 1$ and $x'_i = 0$ for some $i \in Q$ with $\overline{x}_i = 1$. Then $(x', 0)$ satisfies $ax' \geq c$, since $ex' = k - 1$. This implies $a_t \geq a_i$. Since by assumption $a_t \leq a_{t+1} \leq \ldots \leq a_i$, we have $a_t = a_{t+1} = \ldots = a_i = \alpha$, and we update $Q = \{t, t + 1, \ldots, m\}$.

It remains to show, then, that there is a point $(\overline{x}, 0) \in T_1^q$ with $x_t = 0$ for some $t < q$ if $q \neq 1$. Suppose to the contrary, We consider two cases:

Case 1. $T_1 = T_1^q$. Then all the points in T_1 have $x_p = 1, \forall p < q$, since otherwise $(\overline{x}, 0)$ exists. But this implies for $p < q$ that $x_p = 1$ for all points in T, which is impossible since otherwise $x_p \leq 1$ would define the facet.

Case 2. $T_1 \neq T_1^q$. Consider any point $(\overline{x}, 0) \in T_1 \setminus T_1^q$. Since it satisfies $e\overline{x} = k - 1$, we must have $q > k - 1$. On the other hand, consider any point $(x, 0) \in T_1^q$. By hypothesis $x_p = 1$ for all $p < q$, and we must therefore have

$ex \geq k$, which is impossible because $(x, 0) \in T_1$.

We conclude that Q can be updated until $q = 1$. Thus $a_1 = \ldots = a_m = \alpha$.

Similar arguments using T_2 verify that $b_j = \beta, \forall j$. Therefore, (23) becomes

$$\alpha ex + \beta ey \geq c \tag{24}$$

Substituting into (24) a point $(x, y) \in T_1$, we get

$$\alpha(k-1) = c. \tag{25}$$

Doing the same for $(x, y) \in T_2$, we get,

$$\alpha m + \beta l = c.$$

Thus

$$\beta l = -\alpha(m - k + 1)$$

Substituting this and (25) into (24), we have

$$\alpha ex - (\alpha/l)(m - k + 1)ey \geq \alpha(k-1). \tag{26}$$

Since $\alpha < 0$, the lemma follows. □

In the following lemma and elsewhere, we say $\{i_1, \ldots, i_{m-1}\} \subset \{1, \ldots, m\}$ to indicate that $\{i_1, \ldots, i_{m-1}\}$ is a subset of $m-1$ *distinct* indices in $\{1, \ldots, m\}$.

Lemma 6 *Any facet of $conv(S)$ other than (22) is a facet of the projection of $conv(S)$ onto $(x_{i_1}, \ldots, x_{i_{m-1}}, y_1, \ldots, y_n)$ for some $\{i_1, \ldots, i_{m-1}\} \subset \{1, \ldots, m\}$ provided $k < m$, or its projection onto $(x_1, \ldots, x_m, y_{j_1}, \ldots, y_{j_{n-1}})$ for some $\{j_1, \ldots, j_{n-1}\} \subset \{1, \ldots, n\}$ provided $l > 1$.*

Proof: Consider a facet F of $conv(S)$ other than (22). Thus at least one variable is missing from the inequality describing F. The proof is similar whether the missing variable is an x_j or a y_j. Without loss of generality, assume it is x_1. We wish to show that F is a facet for the projection P of $conv(S)$ onto $(x_2, \ldots, x_m, y_1, \ldots, y_n)$. F is clearly valid for P. So it suffices to show that there are $m + n - 1$ affinely independent points of P on F.

Consider $m + n$ independent points of S on F. Let A be the $(m + n) \times (m + n)$ 0-1 matrix with each point as a row. Hence $det(A) \neq 0$. On the other hand, $det(A)$ can be expended in the cofactors of its first column:

$$det(A) = a_{11}A_{11} + \cdots + a_{m+n,1}A_{m+n,1}$$

Clearly, each of the cofactors A_{j1} corresponds to a set of $m+n-1$ vectors in P. Since $det(A) \neq 0$, we must have that at least one of these cofactors is non-zero. This implies that at least one set of $m+n-1$ points in P are independent. Furthermore, they are all on F since its defining inequality does not contain x_1. Thus, F is a facet of P. □

Lemma 7 *If $m > k$, the projection of S onto $(x_{i_1}, \ldots, x_{i_{m-1}}, y_1, \ldots, y_n)$ is precisely the set of 0-1 points satisfying $(x_{i_1}, \ldots, x_{i_{m-1}})_k \Rightarrow (y_1, \ldots, y_n)_l$.*

Proof: Without loss of generality, we again assume

$$(x_{i_1}, \ldots, x_{i_{m-1}}) = (x_2, \ldots, x_m).$$

Now denote by $P_1(S)$ the projection of S onto $(x_2, \ldots, x_m)$. Also let

$$S_1 = \{(x,y) | x \in \{0,1\}^{m-1}, y \in \{0,1\}^n, (x_2, \ldots, x_m)_k \Rightarrow (y_1, \ldots, y_n)_l\}.$$

We wish to show $S_1 = P_1(S)$.

Take a point $(x_2, \ldots, x_m, y) \in S_1$. It is clear that for either $ey \geq l$ or $ey \leq l-1$ we have $(0, x_2, \ldots, x_m, y) \in S$. Thus $(x_2, \ldots, x_m, y) \in P_1(S)$. That is, $S_1 \subseteq P_1(S)$. Now, take $(x_2, \ldots, x_m, y) \in P_1(S)$. Then there exists an x_1 such that $(x_1, x_2, \ldots, x_m, y) \in S$. If $ey \geq l$, clearly $(x_2, \ldots, x_m, y) \in S_1$. If $ey \leq l-1$, we know that at most $k-1$ of $(x_1, x_2, \ldots, x_m)$ can be 1. Therefore, at most $k-1$ of $(x_2, \ldots, x_m)$ can be 1. That is, $(x_2, \ldots, x_m, y) \in S_1$. Thus $P_1(S) \subseteq S_1$. □

Similarly, we have

Lemma 8 *If $l > 1$, the projection of S onto $(x_1 \ldots, x_m, y_{j_1}, \ldots, y_{j_{n-1}})$ is precisely the set of 0-1 points satisfying $(x_1, \ldots, x_m)_k \Rightarrow (y_{j_1}, \ldots, y_{j_{n-1}})_{l-1}$*
□

It is clear now that the facets of $conv(S)$ can be generated recursively. The full description of the convex hull of the general cardinality rule is stated in the following theorems. Here, a "clause" is a logical clause in the traditional sense: a cardinality rule (12) in which $m = k$ and $l = 1$.

Theorem 2 *If (12) is not a clause, and if $k \geq 2$ and $n > l$, the facets of $conv(S)$ are*

$$-l(x_1 + \cdots + x_m) + (m-k+1)(y_1 + \cdots + y_n) \geq l(1-k),$$

plus the facets of

$$(x_{i_1}, \ldots, x_{i_m})_k \Rightarrow (y_1, \ldots, y_n)_l$$

for all sets $\{i_1, \ldots, i_m\} \subset \{1, \ldots, m\}$ *provided* $m > k$, *and the facets of*

$$(x_1, \ldots, x_m)_k \Rightarrow (y_{j_1}, \ldots, y_{j_{n-1}})_{l-1}$$

for all sets $\{j_1, \ldots, j_{n-1}\} \subset \{1, \ldots, n\}$ *provided* $l > 1$.

Proof: The theorem follows from Lemmas 4-8. □

Theorem 3 *If (12) is a clause, the facets of* $conv(S)$ *are given by*

$$-ex + ey \geq 1 - m, \tag{27}$$

plus a) $x_i \geq 0$ *for* $i = 1, \ldots m$, $y_j \leq 1$ *for* $j = 1, \ldots n$ *provided* $m + n > 1$, *and b)* $x_i \leq 1$ *for* $i = 1, \ldots m$, $y_j \geq 0$ *for* $j = 1, \ldots, n$ *provided* $m + n > 2$.

Proof: Lemma 4 implies that (27) defines a facet.

To show that a bound on a variable x_i or y_j is facet defining, we only need to prove that it is tight at $m+n$ affinely independent points. Since the inequality (27) contains all variables, it cuts off only one vertex of the $m+n$ dimensional unit cube. Thus it cuts off at most one of the 2^{m+n-1} vertices at which any given bound is tight. When $m + n > 2$, we have $2^{m+n-1} > m + n$ and the bound defines a facet that contains at least $m + n$ affinely independent points. The theorem is easily verified for the case $m + n = 2$. □

Theorem 4 *If* $m > k = 1$, *the facets of* $conv(S)$ *are the facets of*

$$x_i \Rightarrow (y_1, \ldots, y_n)_l,$$

for $i = 1, \ldots, m$.

Proof: It suffices to show that (22) is not a facet, since in this case the theorem follows from Lemmas 6 and 7.

Here (22) has the form,

$$-lex + mey \geq 0. \tag{28}$$

If (28) defines a facet, it contains a set T of $n + m$ affinely independent points. Take a point $(x, y) \in T$. If $ey \leq l - 1$, then $x = 0$. But since

$m > 0$, $x = 0$ implies $mey = 0$ and therefore $y = 0$. On the other hand, if $ey = l' \geq l$, then we have $-lex + ml' = 0$. Since $ex \leq m$, this implies $l' = l$ and $ex = m$. Thus any point in T is either $(0,0)$ or has form $ex = m$ and $ey = l$.

Since x is a vector of ones, T contains at most $n+1$ affinely independent points, including $(0,0)$. Thus when $m > 1$, (28) does not define a facet. □

The next theorem is similarly proved.

Theorem 5 *If $n = l > 1$ the facets of $conv(S)$ are the facets of*

$$(x_1, \ldots, x_m)_k \Rightarrow y_j$$

for $j = 1, \ldots, n$. □

The next two theorems follow from Lemmas 4 to 8.

Theorem 6 *If $m = k \leq 1$ and $l > 1$, the facets of $conv(S)$ are (22) plus the facets of*

$$x_1 \Rightarrow (y_{j_1}, \ldots, y_{j_{n-1}})_{l-1}$$

if $m = 1$, or those of

$$\Rightarrow (y_{j_1}, \ldots, y_{j_{n-1}})_{l-1}$$

if $m = 0$, for all sets $\{j_1, \ldots, j_{n-1}\} \subset \{1, \ldots, n\}$. □

Theorem 7 *If $n \leq l = 1$ and $m > k$, the facets of $conv(S)$ are (22), plus the facets of*

$$(x_{i_1}, \ldots, x_{i_{m-1}})_k \Rightarrow y_1$$

if $n = 1$, or those of

$$(x_{i_1}, \ldots, x_{i_{-1}m})_k \Rightarrow$$

if $n = 0$, for all sets $\{i_1, \ldots, i_{m-1}\} \subset \{1, \ldots, m\}$ □

2.2 Generating the Facets

To summarize the results of the previous section, we state below a recursive procedure FACET that generates all facets for the cardinality rule $(x_1, \ldots, x_m)_k \Rightarrow (y_1, \ldots, y_n)_l$. Recall that we assume $m = 0$ if $k = 0$, and $l = 1$ if $n = 0$. The procedure can be readily automated.

To generate a 0-1 inequality representation for an arbitrary formula of cardinality logic, one may use the algorithm of Section 1.2 to convert the formula to cardinality normal form—i.e., to a conjunction of elementary

cardinality rules (1). Each elementary cardinality rule may then be given a convex hull representation. If one or more of the literals A_i in (1) is negative and therefore has the form $\neg x_i$, $\neg x_i$ should be replaced by x_i before applying FACET. Then each occurrence of x_i in the generated inequalities should be replaced with $(1 - x_i)$. Negative B_j's should be similarly treated.

Procedure FACET$((x_1, \ldots, x_m)_k \Rightarrow (y_1, \ldots, y_n)_l)$

If $m > k = 1$ then
    For all $i \in \{1, \ldots, m\}$ call FACET$(x_i \Rightarrow (y_1, \ldots, y_n)_l)$.
Else if $n = l > 1$ then
    For all $j \in \{1, \ldots, n\}$ call FACET$((x_1, \ldots, x_m)_k \Rightarrow y_j)$.
Else
    Generate the facet
        $-l(x_1 + \ldots + x_m) + (1 + m - k)(y_1 + \ldots + y_n) \geq l(1 - k)$.
    If $m > k$ then
        For all $\{i_1, \ldots, i_{m-1}\} \subset \{1, \ldots, m\}$,
            call FACET$((x_{i_1}, \ldots, x_{i_{m-1}})_k \Rightarrow (y_1, \ldots, y_n)_l)$.
    If $l > 1$ then
        For all $\{j_1, \ldots, j_{n-1}\} \subset \{1, \ldots, n\}$,
            call FACET$((x_1, \ldots, x_m)_k \Rightarrow (y_{j_1}, \ldots, y_{j_{n-1}})_{l-1})$.
    If $m = k$ and $l = 1$ then
        If $m + n > 1$ then
            For all $i \in \{1, \ldots, m\}$ generate the facet $x_i \geq 0$.
            For all $j \in \{1, \ldots, n\}$ generate the facet $y_j \leq 1$.
            If $m + n > 2$ then
                For all $i \in \{1, \ldots, m\}$ generate the facet $x_i \leq 1$,
                For all $j \in \{1, \ldots, n\}$ generate the facet $y_j \geq 0$.
Return.

3 An Example

A firm wants to build new plants at as many as three sites in order to make as many as three new products.

- If at least 2 plants are built, at least 2 new products should be made.
- If any plants are built, product 1 or 2 should be made.

- If a plant is built at site 1 or 2, then product 1 or 2 should be made only if at least two products are made.

Let

$x_i = 1$ when a plant is built at site i, 0 otherwise.
$y_j = 1$ when product j is made, 0 otherwise.

The three rules can be written,

$$(x_1, x_2, x_3)_2 \Rightarrow (y_1, y_2, y_3)_2 \tag{29}$$
$$(x_1, x_2, x_3)_1 \Rightarrow (y_1, y_2)_1 \tag{30}$$
$$(x_1, x_2)_1 \Rightarrow [(y_1, y_2)_1 \Rightarrow (y_1, y_2, y_3)_2] \tag{31}$$

Rule (29) is already an elementary cardinality rule. We can therefore apply the algorithm of Section 2 directly to obtain the following convex hull description. The indentation reflects the level of recursion. Redundant inequalities and bounds $0 \leq x_i \leq 1$, $0 \leq y_j \leq 1$ are omitted.

$$\begin{aligned}
-2(x_1 + x_2 + x_3) + 2(y_1 + y_2 + y_3) &\geq -2 \\
-2(x_1 + x_2) + y_1 + y_2 + y_3 &\geq -2 \\
-2(x_1 + x_3) + y_1 + y_2 + y_3 &\geq -2 \\
-2(x_2 + x_3) + y_1 + y_2 + y_3 &\geq -2 \\
-x_1 - x_2 - x_3 + 2(y_1 + y_2) &\geq -1 \\
-x_1 - x_2 - x_3 + 2(y_1 + y_3) &\geq -1 \\
-x_1 - x_2 - x_3 + 2(y_2 + y_3) &\geq -1 \\
-x_1 - x_2 + y_1 + y_2 &\geq -1 \\
-x_1 - x_2 + y_1 + y_3 &\geq -1 \\
-x_1 - x_2 + y_2 + y_3 &\geq -1 \\
-x_1 - x_3 + y_1 + y_2 &\geq -1 \\
-x_1 - x_3 + y_1 + y_3 &\geq -1 \\
-x_1 - x_3 + y_2 + y_3 &\geq -1 \\
-x_2 - x_3 + y_1 + y_2 &\geq -1 \\
-x_2 - x_3 + y_1 + y_3 &\geq -1 \\
-x_2 - x_3 + y_2 + y_3 &\geq -1
\end{aligned}$$

The 9 inequalities at the lowest level of recursion represent the clauses in the CNF expansion of (29). The complete convex hull description therefore requires only 7 inequalities in addition to those that would be used in the conventional representation of (29).

Rule (30) is also an elementary cardinality rule and has the convex hull representation,

$$-x_1 + y_1 + y_2 \geq 0$$
$$-x_2 + y_1 + y_2 \geq 0$$
$$-x_3 + y_1 + y_2 \geq 0,$$

which is the same as the conventional CNF representation.

We must use the algorithm of Section 1.2 to put rule (31) in cardinality normal form. This yields the conjunction of the following elementary cardinality rules.

$$(x_1, x_2)_1 \Rightarrow u \tag{32}$$
$$(y_1, y_2)_1 \Rightarrow (z_1, z_2, y_3, w_1, w_2)_2 \tag{33}$$
$$w_1 \Rightarrow \neg u \tag{34}$$
$$w_2 \Rightarrow \neg u \tag{35}$$
$$z_1 \Rightarrow y_1 \tag{36}$$
$$z_2 \Rightarrow y_2 \tag{37}$$

Rule (32) has the convex hull description,

$$-x_1 + u \geq 0$$
$$-x_2 + u \geq 0$$

Rule (33) has the description,

$$-2y_1 + z_1 + z_2 + y_3 + w_1 + w_2 \geq 0$$
$$-y_1 + z_1 + z_2 + y_3 + w_1 \geq 0$$
$$-y_1 + z_1 + z_2 + y_3 + w_2 \geq 0$$
$$-y_1 + z_1 + z_2 + w_1 + w_2 \geq 0$$
$$-y_1 + z_1 + y_3 + w_1 + w_2 \geq 0$$
$$-y_1 + z_2 + y_3 + w_1 + w_2 \geq 0$$
$$-2y_2 + z_1 + z_2 + y_3 + w_1 + w_2 \geq 0$$
$$-y_2 + z_1 + z_2 + y_3 + w_1 \geq 0$$
$$-y_2 + z_1 + z_2 + y_3 + w_2 \geq 0$$
$$-y_2 + z_1 + z_2 + w_1 + w_2 \geq 0$$
$$-y_2 + z_1 + y_3 + w_1 + w_2 \geq 0$$
$$-y_2 + z_2 + y_3 + w_1 + w_2 \geq 0$$

Thus a full convex hull description adds only 2 inequalities to the conventional CNF description.

Rules (34)-(37) respectively have the descriptions,

$$-w_1 - u \geq -1$$
$$-w_2 - u \geq -1$$
$$-z_1 + y_2 \geq 0$$
$$-z_2 + y_2 \geq 0$$

References

[1] Araque, J.R., and V. Chandru, Tight polyhedral representation of logical connectives, Technical Report CC-92-3, IIES, Purdue University, West Lafayette, IN 47970 USA, June 1992.

[2] Blair, C., R. G. Jeroslow, and J. K. Lowe, Some results and experiments in programming techniques for propositional logic, *Computers and Operations Research* **13** (1988) 633-645.

[3] Cook, S. A., The complexity of theorem-proving procedures, *Proceedings of the Third Annual ACM Symposium on the Theory of Computing* (1971) 151-158.

[4] Cook, S. A., Feasibly constructive proofs and the propositional calculus, *Proceedings of the Seventh Annual ACM Symposium on the Theory of Computing* (1975) 83-97.

[5] Dantzig, G. B., *Linear Programming and Extensions*, Princeton University Press (1963).

[6] Hadjiconstantinou, E., and G. Mitra, Transformation of propositional calculus statements into integer and mixed integer programs: An approach toward automatic reformulation, Technical report, Mathematics and Statistics Dept., Brunel University, Uxbridge, Middlesex UB8 3PH, U.K.

[7] Haken, A., The intractability of resolution, *Theoretical Computer Science* **39** (1985) 297-308.

[8] Hooker, J. N., Generalized resolution and cutting planes, *Annals of Operations Research* **12** (1988) 217-239.

[9] Jeroslow, R. S., *Logic-Based Decision Support: Mixed Integer Model Formulation*, *Annals of Discrete Mathematics* **40**, North-Holland (Amsterdam, 1989).

[10] Karp, R. M., Reducibility among combinatorial problems, in R. E. Miller and J. W. Thatcher, eds., *Complexity of Computer Computations*, Plenum Press (1972) 85-103.

[11] McKinnon, K. I. M., and H. P Williams, Constructing integer programming models by the predicate calculus, *Annals of Operations Research* **21** (1989) 227-246.

[12] Tseitin, G. S., On the complexity of derivations in the propositional calculus, in A. O. Slisenko, ed., *Structures in Constructive Mathematics and Mathematical Logic, Part II* (translated from Russian, 1968) 115-125.

[13] Williams, H. P., Logical problems and integer programming, *Bulletin of the Institute of Mathematics and its Implications* **13** (1977) 18-20.

[14] Williams, H. P., Linear and integer programming applied to the propositional calculus, *International Journal of Systems Research and Information Science* **2** (1987) 81-100.

[15] Williams, H. P., *Model Building in Mathematical Programming*, Wiley (Chichester, 1988).

[16] Wilson, J. M., Compact normal forms in propositional logic and integer programming formulations, *Computers and Operations Research* **90** (1990) 309-314.