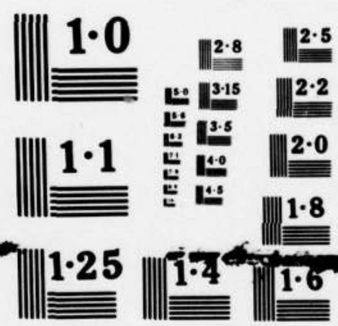
AD-A060 851 STANFORD UNIV CALIF DEPT OF OPERATIONS RESEARCH F/G 12/1
GENERALIZED COVERING RELAXATION FOR 0-1 PROGRAMS.(U)
JUN 78 D GRANOT, F GRANOT N00014-76-C-0418

UNCLASSIFIED TR-79 NL

| OF |
ADA
060851

END
DATE
FILMED
1 -79
DDC



NATIONAL BUREAU OF STANDARDS
MICROCOPY RESOLUTION TEST CHART

LEVEL II

(12)

GENERALIZED COVERING RELAXATION FOR 0-1 PROGRAMS

by

Daniel Granot and Frieda Granot

TECHNICAL REPORT NO. 79
June 1978

PREPARED UNDER CONTRACT
N00014-76-C-0418 (NR-047-061)
FOR THE OFFICE OF NAVAL RESEARCH

Frederick S. Hillier, Project Director

DEPARTMENT OF OPERATIONS RESEARCH
STANFORD UNIVERSITY
STANFORD, CALIFORNIA

ABSTRACT

We construct in this paper a general purpose algorithm for solving polynomial 0-1 programming problems. The algorithm is applied directly to the polynomial problem in its original form. Further, no additional variables are introduced in the solution process.

The algorithm was tested on randomly generated modest size problems and the preliminary computational results obtained are very encouraging.

1. INTRODUCTION

We consider in this paper the general (linear or nonlinear) 0-1 programming (GP) problem of the form

$$
\text{(GP)} \qquad
\begin{aligned}
&\text{Max } c^T x \\
&\text{s.t.} \\
&\qquad f_i(x) \le b_i \qquad (i = 1, \dots, m) \\
&\qquad x_j \in \{0,1\} \qquad (j = 1, \dots, n)
\end{aligned}
$$

where the f_i's are polynomials in the x_j variables.

It was shown that every (GP) problem is equivalent to a special structured linear 0-1 problem - the Generalized Covering (GC) problem - of the form

$$
\text{(GC)} \qquad
\begin{aligned}
&\text{Min } c^T \bar{x} \\
&\text{s.t.} \\
&\qquad A\bar{x} \ge b \\
&\qquad 1 - x_j = \bar{x}_j \in \{0,1\} \qquad (j = 1, \dots, n)
\end{aligned}
$$

where $A = (a_{ij})$ with $a_{ij} \in \{0,1,-1\}$ and $b_i = 1 + \sum_j \text{Min}\{a_{ij},0\}$ $(i = 1, \dots, m)$. See, for example, Granot and Hammer [5,6] for the general (GP) problem and Balas and Jeroslow [1] for the linear (GP) problem.

Granot and Hammer [5] attempted to employ this equivalence relation in an algorithm for solving (GP) problems. Unfortunately, this attempt failed because of the large number of constraints in the equivalent (GC) problem. For example, a constraint of the form $\sum_{j=1}^{n} x_j \le [\frac{n}{2}]$ is equivalent to $\binom{n}{[\frac{n}{2}]+1}$ generalized covering constraints.

Another approach for solving nonlinear (GP) problems suggests to first linearize the nonlinear constraints and then solve the equivalent linear problem, see e.g. [2,3,8]. However, the most severe shortcoming of this linearization approach stems from the radical increase in the dimension of the problem. For example, a (GP) problem with n 0-1 variables, m constraints and k distinct products of variables is transferred, by Glover and Woolsey [3], to an equivalent linear problem with $n + k$ variables and $m + n + k$ constraints. Clearly, this approach will produce unmanageable problems for large k.

A branch and bound algorithm for solving linear and nonlinear 0-1 problems was devised by Hammer in [7].

In [4] an efficient general purpose algorithm (not a branch and bound type) was developed for solving the monotone (GP) problem, i.e., when the coefficients of the f_i's are all nonnegative. In this paper we extend the results of [4] and construct an algorithm for solving general (GP) problems. The algorithm solves a sequence of relatively small (GC) problems, where each (GC) problem is a relaxation of the original (GP) problem as well as a tighter relaxation than its predecessor in the sequence.

Though the algorithm is applicable to both linear and nonlinear (GP) problems, its promise lies mainly for the nonlinear case. One of its main advantages is that no additional variables are introduced in the solution procedure.

Preliminary computational results for randomly generated (GP) problems are reported in Section 4.

2. PRELIMINARIES

Consider first the linear (GP) problem of the form:

$$\text{Max } c^T x \tag{1}$$

s.t.

$$Ax \leq b \tag{2}$$

$$x \in \{0,1\} \tag{3}$$

where $c \in R^{n\times 1}$, $b \in R^{m\times 1}$, $A \in R^{m\times n}$ are given with $c \geq 0$.

Every constraint of (2) can be replaced by an equivalent set of generalized covering constraints in the following manner. Consider a single inequality of the form:

$$\sum_{j=1}^{n} a'_j x_j \leq b' \tag{4}$$

where $|a'_1| \geq |a'_2| \geq \cdots \geq |a'_n|$. For every j for which $a'_j < 0$ we can substitute $x_j = 1 - \bar{x}_j$ to obtain an equivalent constraint of the form

$$\sum_{j=1}^{n} a_j x_j^{\alpha_j} = \sum_{j=1}^{n} |a'_j| x_j^{\alpha_j} \leq b' - \sum_{j=1}^{n} \min(0,a_j) = b \tag{5}$$

where

$$x_j^{\alpha_j} = \begin{cases} x_j & \text{if } a_j \geq 0 \\ \bar{x}_j & \text{if } a_j < 0 . \end{cases}$$

A subset $V \subset N = \{1,2, \ldots , n\}$ is said to be a <u>cover</u> of (5) if

$$\sum_{j\in V} a_j > b . \tag{6}$$

It is said to be a <u>prime cover</u> if no proper subset of it is a cover.

If we further denote by $S = \{x: \sum_{j=1}^{n} a_j x_j^{\alpha_j} \leq b\}$ and by Ω_S the set of all prime covers V_i of (5), then it is easy to see (e.g. [1,5]) that

(7) $$x \in S \text{ iff } \prod_{j \in V_i} x_j^{\alpha_j} = 0 \ \forall V_i \in \Omega_S \text{ iff } \sum_{j \in V_i} \bar{x}_j^{\alpha_j} \geq 1 \ \forall V_i \in \Omega_S$$

where $\bar{x}_j^{\alpha_j} = 1 - x_j^{\alpha_j}$.

Applying the above procedure to every constraint in (2) will result with an equivalent generalized covering problem.

This result can be further generalized (see [5]) to (GP) problems of the form

(8) $$\begin{aligned} &\text{Max } c^T x \\ &\text{s.t.} \\ &f_i(x) \leq b_i \qquad (i = 1, \ldots, m) \\ &x_j \in \{0,1\} \qquad (j = 1, \ldots, n) \end{aligned}$$

where $f_i(x_1, \ldots, x_n)$ are polynomials of the form:

(9) $$f_i(x_1, \ldots, x_n) = \sum_{k=1}^{p_i} a_{ik} \prod_{j \in N_{ik}} x_j$$

with N_{ik} any subset of N.

In order to obtain the set of generalized covering constraints which is equivalent to a polynomial inequality of the form

(10) $$f = \sum_{k=1}^{p'} a'_k \prod_{j \in N'_k} x_j \leq b' ,$$

one can proceed as follows: For every k for which $a'_k < 0$ substitute

$\sum_{j\in N_k'} \bar{x}_j \prod_{\substack{\ell\in N_k' \\ \ell < j}} x_\ell$ for $\prod_{j\in N_k'} x_j$ in (10) to obtain an equivalent constraint of the form:

(11)
$$\sum_{k=1}^{p} a_k \prod_{j\in N_k} x_j^{\alpha_j} \triangleq \sum_{+} a_k' \prod_{j\in N_k'} x_j + \sum_{-} |a_k'| \sum_{j\in N_k'} \bar{x}_j \prod_{\substack{\ell\in N_k' \\ \ell < j}} x_\ell$$
$$\leq b' - \sum_{-} a_k' = b$$

where $\sum_{+}$ $(\sum_{-})$ denotes summation over positive (negative) values of a_k' and $x_j^{\alpha_j}$ equals to either x_j or $\bar{x}_j$ in accordance with (11). Now, by substituting $y_k = \prod_{j\in N_k} x_j^{\alpha_j}$ in (11) we obtain an equivalent linear inequality in the variables y_k of the form:

(12)
$$\sum_{k=1}^{p} a_k y_k \leq b \ .$$

Using the method described earlier one can now obtain the set $\bar{\Omega}_S$ of prime covers V_i to (12). Then (12) is equivalent to

$$\prod_{k\in V_i} y_k = 0 \ \forall \ V_i \in \bar{\Omega}_S \quad \text{iff} \quad \prod_{k\in V_i} \prod_{j\in N_k} x_j^{\alpha_j} = 0 \ \forall \ V_i \in \bar{\Omega}_S \ .$$

Thus, an inequality of the form (10) is equivalent to the following set of generalized covering inequalities

(13)
$$\sum_{j\in \bar{V}_i} \bar{x}_j^{\alpha_j} \geq 1 \ \forall \ V_i \in \bar{\Omega}_S$$

where $\bar{V}_i = \bigcup_{k\in V_i} N_k$.

Employing the above procedure to every constraint of (GP) results with an equivalent generalized covering problem.

3. AN ALGORITHM FOR SOLVING (GP) PROBLEMS

We construct in this section a general purpose algorithm for solving (GP) problems. We first generate a generalized covering problem which is a relaxation of (GP), referred to as the Generalized Covering Relaxation (GCR) problem. If the optimal solution to (GCR) is feasible to (GP), it is also optimal. Otherwise (GCR) is augmented by additional generalized covering constraints that eliminate this optimal solution. Each augmented constraint will be shown to dominate the corresponding generalized covering constraint produced by the method of Section 2. The (GCR) problem is then resolved. The variables in each (GCR) problem are the original variables of (GP) and its constraints are a small subset of the constraints of the equivalent (GC) problem.

A distinctive feature of this algorithm is that it does not generate the entire equivalent (GC) problem. Rather, the size of each (GCR) problem solved is relatively small. Additional constraints are generated only to eliminate the optimal solution to (GCR) when not feasible to (GP). Moreover, no additional variables are introduced in the solution procedure. The variables in each (GC) problem are the original variables of the (GP) problem.

Consider again the general (GP) problem:

$$\text{Max } c^T x$$

$$\text{s.t.}$$

$$\sum_{k=1}^{p_i} a_{ik} \prod_{j \in N_{ik}} x_j \leq b_i \qquad (i = 1, \dots, m) \tag{14}$$

$$x_j \in \{0,1\} \qquad (j = 1, \dots, n)$$

and assume, without loss of generality, that $c_j \geq 0$. Let (GCR) be any relaxation of (14) with $\hat{x}$ its optimal solution. Further let I denote the set of indices corresponding to constraints in (14) which are violated by $\hat{x}$. For each $i \in I$ denote by:

$$S_i^+(\hat{x}) = \{k: a_{ik} > 0 \text{ and } \prod_{j \in N_{ik}} \hat{x}_j = 1\}$$

$$S_i^-(\hat{x}) = \{k: a_{ik} < 0 \text{ and } \prod_{j \in N_{ik}} \hat{x}_j = 1\}$$

$$S_i^0(\hat{x}) = \{k: a_{ik} < 0 \text{ and } \prod_{j \in N_{ik}} \hat{x}_j = 0\} ,$$

i.e., $S_i^+(\hat{x})(S_i^-(\hat{x}))$ is the set of indices corresponding to terms in the i^{th} constraint with positive (negative) coefficients that do not vanish at $\hat{x}$, and $S_i^0(\hat{x})$ corresponds to terms in the i^{th} constraint with negative coefficients that do vanish at $\hat{x}$. Let $S_i(\hat{x})$ denote the ordered set $S_i(\hat{x}) = \{s(1), s(2), \ldots , s(\ell)\}$ where $s(i) \in S_i^+(\hat{x}) \cup S_i^0(\hat{x})$ and $s(k) < s(j)$ if either $|a_{is(k)}| > |a_{is(j)}|$, or if $|a_{is(k)}| = |a_{is(j)}|$ and $k < j$.

Consider now the constraint

$$(15) \qquad \sum_{k=1}^{\ell} a_{is(k)} \prod_{j \in N_{is(k)}} x_j \leq b_i - \sum_{j \in S_i^-(\hat{x})} a_{ij} = \bar{b}_i$$

which was derived from the i^{th} constraint in (14), $i \in I$. Since (15) is violated by $\hat{x}$, it can serve for generating generalized covering constraints which when augmented to (GCR) will eliminate $\hat{x}$. The constraint we will generate, using the following procedure, dominates that constraint produced from an original violated constraint using Granot and Hammer's method [5] (as described in Section 2).

Procedure I

Step 0: Set $\hat{b} = 0$, $k = 1$, Cover $= \phi$

$$b = \bar{b}_i - \sum_{j \in S_i^0(\hat{x})} a_{is(j)} .$$

Go to Step 1.

Step 1: Set $\hat{b} = \hat{b} + |a_{is(k)}|$. If $a_{is(k)} > 0$, then set

$$\text{Cover} = \text{Cover} \cup \{j: j \in N_{is(k)}\} .$$

Otherwise if $a_{is(k)} < 0$ and j is any index in $N_{is(k)}$ for which $\hat{x}_j = 0$ set

$$\text{Cover} = \text{Cover} \cup \{j\} .$$

Step 2: If $\hat{b} > b$, then augmented (GCR) with

$$(16) \qquad \sum_{j \in \text{Cover}} \text{sign}(a_{is(j)}) \cdot \bar{x}_j \geq 1 - \sum_{j \in \text{Cover}} \text{Min}(0, \text{sign}(a_{is(j)}) .$$

Otherwise, if $\hat{b} \leq b$, set $k = k + 1$ and go to Step 1.

Observe that since Procedure I is applied only to constraints i, $i \in I$, the procedure will terminate after applying Step 2 at most $|S_i(\hat{x})|$ times, where $|S_i(\hat{x})|$ is the cardinality of $S_i(\hat{x})$. Moreover (16) dominates the corresponding constraint obtained using Granot and Hammer's method [5]. This since a term in (15) with a negative coefficient contributes at most a single variable to (16).

Example 1:

Consider the following constraint:

$$5x_1x_2 - 3x_3x_4x_5 + 2x_2x_6 \leq 4$$

and the solution $\hat{x} = (1,1,1,1,0,0)$. Since $\hat{x}$ violates the given constraint, we apply Procedure I to the following:

$$5x_1x_2 - 3x_3x_4x_5 \leq 4 \tag{17}$$

which will result with the generalized covering constraint:

$$\bar{x}_1 + \bar{x}_2 - x_5 \geq 0 \; . \tag{18}$$

Applying the method described in Section 2 for generating a generalized cover, we first transfer (17) to the following equivalent constraint with positive coefficients

$$5x_1x_2 + 3\bar{x}_3 + 3x_3\bar{x}_4 + 3x_3x_4\bar{x}_5 \leq 7 \; .$$

Now since $\bar{x}_3 = \bar{x}_4 = 0$ we will derive a generalized cover from

$$5x_1x_2 + 3x_3x_4\bar{x}_5 \leq 7$$

which results with

$$\bar{x}_1 + \bar{x}_2 + \bar{x}_3 + \bar{x}_4 - x_5 \geq 0 \; .$$

This constraint is obviously weaker than (18).

Lemma 1:

Augmenting (GCR) with (16) eliminates $\hat{x}$ and results with a new (GCR) problem which is a relaxation of (GP).

Proof:

Since $\hat{x}$ violates the i^{th} constraint in (14), we have

$$(19) \qquad b < \sum_{k=1}^{p_i} a_{ik} \prod_{j \in N_{ik}} \hat{x}_j = \sum_{j \in S_i^+(\hat{x})} a_{ij} + \sum_{j \in S_i^-(\hat{x})} a_{ij} .$$

But

$$(20) \qquad \sum_{j \in S_i^+(\hat{x})} a_{ij} = \sum_{k=1}^{\ell} a_{is(k)} \prod_{j \in N_{is(k)}} \hat{x}_j > b - \sum_{j \in S_i^-(\hat{x})} a_{ij} = \bar{b}_i$$

and thus $\hat{x}$ violates (15). Any cover derived from (15) will result with a constraint that when augmented to (GCR) will eliminate $\hat{x}$. Further, since (16) is valid for (15), it is also valid for (14) and thus no feasible solutions to (14) are eliminated.

An Algorithm for Solving (GP) Problems:

Step 0: Start with (GCR) being the trivial relaxation, i.e., $\text{Max}\{c^T x: x \in \{0,1\}\}$, whose optimal solution is $\hat{x} = (1, \ldots, 1)$.

Step 1: Let I be the set of indices corresponding to constraints of (14) which are violated by $\hat{x}$. Apply procedure I to generate k generalized covering constraints for each $i \in I$ and add them to (GCR).

Step 2: Solve the augmented (GCR) problem to obtain its optimal solution $\hat{x}$. If $x_j^* = 1 - \hat{x}_j$ $(j = 1, \ldots, n)$ is feasible to (GP), terminate with x^* an optimal solution to (GP). Otherwise go to Step 1.

Theorem:

The algorithm described above converges to an optimal solution of (GP) in finitely many iterations.

Proof:

From Lemma 1 we conclude that (GCR) is a relaxation of (GP), which implies optimality. In step 1 of the algorithm we eliminate an optimal solution to (GCR) which is not feasible to (GP). Since the number of binary vectors is finite, the convergence follows.

Remark 1:

Observe that (15) is violated by $\hat{x}$ for every $i \in I$ and thus every one of the $k \times |I|$ (where $|I|$ is the cardinality of I) constraints added to (GCR) is sufficient to eliminate $\hat{x}$. Generating more than one generalized cover will hopefully reduce the number of (GCR) problems needed to be solved.

Remark 2:

Note that no additional variables are introduced by the algorithm in the solution process. Further, no special devices are needed to handle nonlinear constraints.

Example 2:

Solve

$$\text{Max } 3x_1 + 4x_2 + 6x_3 + x_4 + x_5 + 5x_6$$

s.t.

$$7x_1 - 2x_2x_4 + 4x_3 - 2x_4 - x_5 + 6x_6 \leq 4$$
$$4x_1x_3 - 8x_2x_3 + 3x_1x_2x_3 - 2x_3x_4 - 3x_5x_6 + 7x_6 \leq -1$$
$$3x_1x_2 + 4x_3 - x_4x_5 \leq 5$$
$$5x_1x_3 + 4x_2x_6 - 4x_1x_4 + 2x_3 - x_4x_5x_6 \leq 4$$
$$x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \in \{0,1\} .$$

We start with $\hat{x} = (1,1,1,1,1,1)$.

Iteration I:

For $\hat{x} = (1,1,1,1,1,1)$ the left-hand sides equal $(12,1,6,6)$, hence every constraint is initially violated, i.e., $I = \{1,2,3,4\}$. We start by setting $k = 1$ and thus generate one generalized covering constraint from every violated constraint. Since $S_i^-(\hat{x}) = \phi$, $i \in I$, the generalized covers are derived from the following constraints:

$$7x_1 + 4x_3 + 6x_6 \leq 9$$
$$4x_1x_3 + 3x_1x_2x_3 + 7x_6 \leq 12$$
$$3x_1x_2 + 4x_3 \leq 6$$
$$5x_1x_3 + 4x_2x_6 + 2x_3 \leq 4 .$$

The (GCR) problem obtained is

$$\text{Min } 3\bar{x}_1 + 4\bar{x}_2 + 6\bar{x}_3 + \bar{x}_4 + \bar{x}_5 + 5\bar{x}_6$$

s.t.

$$\begin{aligned}
&\bar{x}_1 + \bar{x}_6 \geq 1 \\
&\bar{x}_1 + \bar{x}_3 + \bar{x}_6 \geq 1 \\
&\bar{x}_1 + \bar{x}_2 + \bar{x}_3 \geq 1 \\
&\bar{x}_1 + \bar{x}_2 + \bar{x}_3 + \bar{x}_6 \geq 1 \\
&x_i \in \{0,1\} \quad (i = 1, \ldots, 6)
\end{aligned}$$

whose optimal solution is $\bar{x}_1 = 1$, $\bar{x}_i = 0$, $(i = 2, \ldots, 6)$.

Iteration 2:

For $\hat{x} = (0,1,1,1,1,1)$ the left-hand sides equal to $(5,-6,3,5)$. Since the only violated constraints are the first and the last one, $I = \{1,4\}$ and we derive additional generalized covers from the following constraints

$$\begin{aligned}
&4x_3 + 6x_6 \leq 9 \\
&4x_2x_6 - 4x_1x_4 + 2x_3 \leq 5 .
\end{aligned}$$

When augmenting the newly derived generalized covers to the previous (GCR) problem we obtain, after some simplifications,

$$\text{Min } 3\bar{x}_1 + 4\bar{x}_2 + 6\bar{x}_3 + \bar{x}_4 + \bar{x}_5 + 5\bar{x}_6$$

s.t.

$$\begin{aligned}
&\bar{x}_1 + \bar{x}_6 \geq 1 \\
&\bar{x}_3 + \bar{x}_6 \geq 1 \\
&\bar{x}_1 + \bar{x}_2 + \bar{x}_3 \geq 1 \\
&\bar{x}_2 + \bar{x}_3 + \bar{x}_6 - \bar{x}_4 \geq 0 \\
&\bar{x}_i \in \{0,1\} \quad (i = 1, \ldots, 6)
\end{aligned}$$

whose optimal solution is $\bar{x}_1 = \bar{x}_6 = 1$, $\bar{x}_2 = \bar{x}_3 = \bar{x}_4 = \bar{x}_5 = 0$. Since $\hat{x} = (0,1,1,1,1,0)$ is feasible to the original problem, it is also an optimal solution.

Observe that if we attempt to solve the above example by first transferring it to a linear 0-1 problem, via Glover and Woolsey's most efficient transformation [3], the resulting equivalent problem will have 11 additional variables and 17 additional constraints.

4. SIMPLIFICATION AND COMPUTATIONAL RESULTS

It is well known that simplification and elimination rules can be applied to the rows and columns of the coefficient matrix of a covering problem before solving the problem. Similar simplification and elimination rules can be devised for the generalized covering problem. These simplification rules, to be described below, are employed to any newly generated generalized cover in order to reduce the size of the coefficient matrix A of the (GCR) problem.

Let $a_i(a^j)$ denote the i^{th} row (j^{th} column) of the matrix A. Then the following simplifications are applied at each iteration, see also [5].

Simplification 1

If a row a_i contains a unique 1 (unique -1) in the j^{th} column, then set $x_j = 0$ ($x_j = 1$) in the original (GP) problem.

Simplification 2

If there exist rows a_i, a_j for which $a_{ik} \neq 0$ implies $a_{ik} = a_{jk}$, then row a_j can be eliminated.

Simplification 3

If there exist rows a_i, a_j for which $a_{ik} = a_{jk}\ \forall\, k \neq \ell$ and $a_{i\ell} = 1 = -a_{j\ell}$, then substitute $a_{j\ell} = 0$ and eliminate row a_i from A.

Simplification 4

If there exist rows a_i, a_j and index p such that $a_{ip} = -a_{jp} \neq 0$ and $a_{ik} \neq 0$ implies $a_{ik} = a_{jk}$, $k \neq p$, then substitute $a_{jp} = 0$ and eliminate row a_i.

Simplification 5

If there exist rows a_i, a_j, and indices r, p for which $a_{ik} = a_{jk} = 0$, $k \neq r, p$ and $a_{ir} = -a_{jr} = -a_{ip} = a_{jp} \neq 0$, then substitute $x_r = x_p$ in the original (GP) problem.

Simplification 6

If there exist rows a_i, a_j and indices r, p for which $a_{ik} = a_{jk} = 0$, $k \neq r, p$ and $a_{ir} = -a_{jr} = a_{ip} = -a_{jp} \neq 0$, then substitute $x_r = 1 - x_p$ in the original (GP) problem.

Simplification 7

If there exist a row a_i for which $a_{ik} = 0 \; \forall \, k \neq \ell, r$ and $a_{i\ell} = a_{ir} = 1$ $(a_{i\ell} = a_{ir} = -1)$, then in any row a_j for which $a_{j\ell} = -a_{jr} = 1$ substitute $a_{jr} = 0$ $(a_{j\ell} = 0)$, and in any row a_j for which $a_{j\ell} = -a_{jr} = -1$ substitute $a_{j\ell} = 0$ $(a_{jr} = 0)$.

Simplification 8

If there exist a row a_i for which $a_{ik} = 0$, $k \neq \ell, r$ and $a_{i\ell} = -a_{ir} = 1$, then in any row a_j for which $a_{j\ell} = a_{jr} = 1$ $(a_{j\ell} = a_{jr} = -1)$ substitute $a_{jr} = 0$ $(a_{j\ell} = 0)$.

Simplification 9

If there exist columns a^i, a^j for which $a^i \geq a^j$ and $c_i \leq c_j$, then column a^j can be eliminated (i.e., $x_j = 0$ in the current (GCR) problem).

Simplification 10

If there exist a column a^i and an index set J, $i \notin J$, for which $\sum_{j \in J} a^j \geq a^i$ and $\sum_{j \in J} c_j \leq c_i$, then column a^i can be eliminated.

Our algorithm was programmed in Fortran IV, implemented on an IBM 370/168 and tested on randomly generated problems.

In the preliminary computational results to be reported, the vector c was determined by setting $c_0 = 0$ and $c_{j+1} = c_j + k$ where k is uniform on $[0,10]$. Every constraint is of the form

$$\sum_{j=1}^{k} a_{ij} T_j \leq b_i \quad \text{with} \quad T_j = \prod_{r=1}^{f} x_{D_r}$$

and is generated as follows:

- k - the number of terms is chosen uniformly between 3 and K.
- f - the number of variables in each term is chosen uniformly between 1 and 6.
- a_{ij} - is chosen uniformly between -20 and 20.
- D_r - is chosen uniformly between 1 and the number of variables.
- R_i - is the range $[\sum_{-} a_{ij}, \sum_{+} a_{ij}] = [L_i, U_i]$ where $\sum_{-}$ $(\sum_{+})$ denotes summation over negative (positive) coefficients. Thus $[L_i, U_i]$ denotes the range of the left-hand side of the i^{th} constraint.
- α - a constant.
- M - the number of constraints.
- N - the number of variables.

By changing α, we tested the influence of large and small values of b on the computation time.

In the following four tables we summarize the preliminary computational results. Each cell is averaged from 4 runs and contains three entries, A, B, C, which designate

A - Number of iterations (i.e., (GCR) problems solved)

B - Number of covers in the last (GCR) problem solved (after simplifications)

C - True execution time in seconds, i.e., total CPU time minus loading, input/output and overhead time.

	$\alpha = .6$	$\alpha = .75$	$\alpha = .9$
N = 30	11, 28, 1.30	*	*
N = 40	11, 32, 2.43	3.6, 9.3, .07	*
N = 50	12, 34, 4.68	6.6, 15.5, .11	1, 3.6, .03

Table 1: $M = 20,\ K = 7,\ b_i = \alpha U_i + (1-\alpha)L_i\ (i = 1,\dots,20)$

	$\alpha = .6$	$\alpha = .75$	$\alpha = .9$
N = 30	17, 49, 2.08	*	*
N = 40	17.5, 51, 4.82	5.2, 13, .08	*
N = 50	20, 60, 12.81	7.2, 18.5, .24	1.5, 4.0, .04

Table 2: $M = 20,\ K = 10,\ b_i = \alpha U_i + (1-\alpha)L_i\ (i = 1,\dots,20)$

Observe that the columns in Tables 1, 2 correspond to setting up the right-hand side b_i to equal a certain value in the range R_i of the i^{th} constraint.

---

*The missing entries in the above tables correspond to trivial cases, and thus are not reported.

M = 15	M = 20	M = 25	M = 30
5, 13, .16	7, 20, .32	9.5, 32.5, 1.14	14, 54, 4.4

Table 3: $N = 40$, $K = 7$; b_i^+ is uniform on

$$[\tfrac{7}{12} U_i + \tfrac{5}{12} L_i, \tfrac{9}{12} U_i + \tfrac{3}{12} L_i] \ (i = 1,\ldots,M)$$

M = 15, N = 30	M = 20, N = 40	M = 25, N = 50
6.2, 17, .25	10, 26.5, 2.14	17, 36, 6.38

Table 4: $K = 7$; b_i^+ is uniform on

$$[\tfrac{7}{12} U_i + \tfrac{5}{12} L_i, \tfrac{9}{12} U_i + \tfrac{3}{12} L_i] \ (i = 1, \ldots, M)$$

The following conclusions can be drawn from the preliminary computational results.

a) Tables 1 and 2 reveal that the tighter the constraints are the more difficult it is to solve a (GP) problem. We recall that the covering relaxation algorithm for solving monotone 0-1 problems [4] has precisely the same property.

We remark that when we attempted to further restrict the r.h.s. vector b (i.e., decreasing α) almost all of the randomly generated problems were found to be infeasible by our algorithm. Further, increasing the number of constraints M beyond the ratio $N/M = 2$ also results with almost all randomly generated problems being infeasible.

+ The r.h.s. b_i in Tables 3, 4 was chosen so as to roughly vary between 2/3 and 3/4 of the range R_i.

b) By comparing Tables 1 and 2 we conclude that the difficulty of solving a (GP) problem increases with an increase in the number of terms in each constraint (i.e., an increase in K). This result again was expected since by increasing K we increase the number of equivalent generalized covering constraints.

c) Tables 1 and 2 exhibit also the significant effect of an increase in the number of variables on the execution time. We remark though that the increase in both the number of iterations and the number of generalized covers is more moderate.

d) Table 3 reveals that an increase in the number of constraints increases significantly the execution time. Again, the increase in the number of iterations and the size of the (GCR) problems is more moderate.

e) Table 4 exhibits the effect of increasing both M and N, in a fixed ratio, on the solution characteristics. Again, the increase in execution time is very significant while the increase in the number of iterations and the number of generalized covers is more moderate.

It appears that the genralized covering relaxation approach for solving modest size (GP) problems is quite promising. For example, (GP) problems with 50 variables, 20 constraints and $K = 10$ are solved, on the average, in 12.81 seconds. What is particularly encouraging is the relatively modest number of iterations required to solve a (GP) problem, for example,(GP) problems with $N = 50$, $M = 20$, $K = 10$ are solved on the average in 20 iterations. Clearly, by improving the efficiency of our generalized covering algorithm we can further reduce the execution time.

Some improvements of the algorithm are being currently considered. In particular, devising and employing approximate algorithms for both the (GP) and the (GCR) problems in order to accelerate the implicit enumeration algorithm used for solving the (GCR) problem.

References

1. E. Balas and R. Jeroslow, "Canonical Cuts on the Unit Hyperplane," Carnegie-Mellon University. Management Science Research Report No. 196 (R), August-December 1969. Revised February 1971.

2. F. Glover and E. Woolsey, "Further Reduction of Zero-One Polynomial Programming Problems to Zero-One Linear Programming Problems," Operations Research, Vol. 21, No. 1 (1973), pp. 141-161.

3. F. Glover and E. Woolsey, "Converting the 0-1 Polynomial Programming Problem to a 0-1 Linear Program," Operations Research, Vol. 22, No. 1 (1974), pp. 180-182.

4. D. Granot, F. Granot and J. Kallberg, "Generalized Covering Relaxation in Monotone {0-1} Programming," submitted to Management Science.

5. F. Granot and P. L. Hammer, "On the Use of Boolean Functions in 0-1 Programming," Methods of Operations Research, XII, July 1971, pp. 154-184.

6. F. Granot and P. L. Hammer, "On the Role of Generalized Covering Problems," Cahiers du Centre D'Etudes de Recherche Operationelle, Vol. 17 (1975), pp. 277-289.

7. P. L. Hammer, "A B-B-B Method for Linear and Non-Linear Bivalent Programming," Operations Research, Statistics and Economics Mimeograph Series No. 48, Technion, May 1969.

8. L. J. Watters, "Reduction of Integer Polynomial Programming Problems to Zero-One Linear Programming Problems," Operations Research, Vol. 15 (1967), pp. 1171-1174.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE (14)	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: TR-79 — 2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER (9)
4. TITLE (and Subtitle): (6) GENERALIZED COVERING RELAXATION FOR 0-1 PROGRAMS.	5. TYPE OF REPORT & PERIOD COVERED: Technical Report.
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): (10) DANIEL GRANOT and FRIEDA GRANOT	8. CONTRACT OR GRANT NUMBER(s): (15) N00014-76-C-0418
9. PERFORMING ORGANIZATION NAME AND ADDRESS: DEPARTMENT OF OPERATIONS RESEARCH, STANFORD UNIVERSITY, STANFORD, CALIF.	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: NR-047-061
11. CONTROLLING OFFICE NAME AND ADDRESS: OPERATIONS RESEARCH PROGRAM, CODE 434 OFFICE OF NAVAL RESEARCH, ARLINGTON, VIRGINIA 22217	12. REPORT DATE: (11) June 1978
	13. NUMBER OF PAGES: 21
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): (12) 25p.	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE, DISTRIBUTION IS UNLIMITED.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

A060 977

18. SUPPLEMENTARY NOTES

This report also issued as Technical Report #78-16 on Contract F 44620-74-C-0079, Air Force

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Polynomial 0-1 Program
Covering Relaxation
Integer Programming

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

We construct in this paper a general purpose algorithm, is constructed for solving polynomial 0-1 programming problems. The algorithm is applied directly to the polynomial problem in its original form. Further, no additional variables are introduced in the solution process.
The algorithm was tested on randomly generated modest size problems and the preliminary computational results obtained are very encouraging.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

402 766

LB