THE GROUP APPROACH TO INTEGER PROGRAMMING:
 AN OUTLINE OF RECENT PROGRESS

by

D. E. BELL

Introduction

This paper will give a brief and simplified view of progress in solving linear integer programs with the group relaxation approach introduced by Gomory [7]. It is intended that this should give an overall viewpoint from which a more detailed study may be made by consulting the references. The first section describes a standard form of algorithm for discrete programs, the branch and bound technique. It is essential to bear this framework in mind when considering the later algorithms of this paper for their main function is to provide good lower bounds quickly rather than to provide the optimal solution, even though they could be used for that purpose.

For the purposes of Section 1 the optimization

$$\text{minimize } f(x)$$
$$x \in Q$$

(1.1)

will be called “problem $Q$” where $Q \subseteq \mathbb{R}^n$ is a discrete set and $f : \mathbb{R}^n \rightarrow \mathbb{R}$. In later sections this will be specialized to the case where $f(x)$ is linear in $x$ and where $Q = \{x \mid Ax = b, \ x \geq 0 \text{ and integer}\}$, $A$ and $b$ having integral coefficients.

1. The Branch and Bound Framework

Branching. A partition of the set $Q$ is its subdivision into disjoint subsets, $Q_1, \ldots, Q_k$. The key point is that if $x^j$ is the optimal solution to problem $Q_j$:

$$\text{minimize } f(x)$$
$$x \in Q_j$$

for all $j = 1, \ldots, k$ then if

$$f(x^j) \leq f(x^l)$$

for all $j = 1, \ldots, k$ for some $i, 1 \leq i \leq k$ then $x^l$ solves problem $Q$.

Bounding. If an upper bound $u(Q)$ is known for the optimal value $u(Q)$ of problem $Q$ and if $l(Q_i)$ is a lower bound for problem $Q_i$, where $Q_i$ is a subset of $Q$ and

$$l(Q_i) > u(Q)$$

(1.2)

then the optimal solution of problem $Q$ does not lie in $Q_i$. When (1.2) occurs the subset $Q_i$ has been fathomed.
The general strategy is thus, given a problem $P$, to attempt to find an optimal solution, or if this proves to be too difficult, to find a good lower bound for its value and then to partition $P$ and repeat the process on the subproblems. The upper bound for the original problem $u(Q)$ is decreased as further information is obtained to make easier the task of fathomimg.

The flow chart in Fig. 1 gives the main ideas.

The algorithm ends when the list is empty, that is, all problems and subproblems have been solved or fathomed. At least one subproblem must have produced an optimal feasible solution since any subproblem containing the optimal solution cannot be fathomed. The current minimum feasible solution is optimal.

In the flow chart Boxes 1, 3, 4, 8, 9 and 10 can all be subjected to analysis in order to find the combination of speed and efficiency which contributes most to the overall effectiveness of the algorithm. A possible cutting plane approach to integer programming fits into this framework as follows.
Group approach to integer programming

Box 1. Solve

\[
\text{maximize } cx \\
\text{s.t. } x \in LP(Q)
\]

where \(LP(Q)\) is the linear programming relaxation of \(Q\) formed by ignoring integrality requirements.

\(u(Q)\) is set to this optimal value.

Box 4.

\[
l(P) = \text{minimize } cx \\
\text{s.t. } x \in LP(P)
\]

Box 9. Generate a constraint which makes the optimal solution in Box 4 infeasible without making any point of \(P\) infeasible. The improved bound is found by solving

\[
l(P) = \text{minimize } cx \\
\text{s.t. } x \in LP(Q) \\
\text{and } dx \geq d_0
\]

where \(dx \geq d_0\) is the cut.

Box 10. If \(P\) was not fathomed or solved, at least one variable in the last optimal solution \(x^*\) must be non-integer, say \(x_1^*\).

Form a partition

\[
P_1 = P \cap \{x_1 \geq \lceil x_1^* \rceil + 1\} \cap \{dx \geq d_0\} \\
P_2 = P \cap \{x_1 \leq \lceil x_1^* \rceil\} \cap \{dx \geq d_0\}
\]

where \([n]\) is the largest integer not greater than \(n\). A more detailed outline of the cutting plane approach in branch and bound may be had by consulting the survey by Baliński [1].

2. The group relaxation: A first lower bound [7]

The linear integer program (IP) is

\[
v = \text{minimize } cx \\
\text{s.t. } Ax = b \\
x = (x_1, x_2, \ldots, x_n, x_{n+1}) \geq 0 \text{ and integer}
\]

(2.1)

where \(A, b\) have integral coefficients and are of compatible dimensions.

Let \(M\) be any matrix of rational numbers then the set of group equations

\[
MAX \equiv Mb \text{ (modulus 1)}
\]

(2.2)

is a relaxation of \(Ax = b\), that is, any \(x\) which satisfies (2.1) must satisfy (2.2)

Example

\[
3x_1 + x_2 + 9x_3 = 11 \\
-12x_1 + 7x_3 + 3x_4 = 11.
\]

(2.3)
If

\[ M = \begin{pmatrix} \frac{1}{7} & 2 \\ \frac{2}{7} & 7 \\ \frac{1}{2} & 0 \end{pmatrix} \]

then the group equations are

\[ \frac{1}{7} x_2 + \frac{2}{7} x_3 + \frac{6}{7} x_4 \equiv \frac{5}{7} \pmod{1} \]

\[ \frac{1}{2} x_1 + \frac{1}{2} x_2 + \frac{1}{2} x_3 \equiv \frac{1}{2} \pmod{1} \]

or equivalently

\[ x_2 + 2x_3 + 6x_4 \equiv 5 \pmod{7} \]

\[ x_1 + x_2 + x_3 \equiv 1 \pmod{2} \]

(2.4)

Solutions to (2.4) include \((0, 0, 3, 1), (0, 1, 2, 0), (0, 5, 0, 0), (1, 0, 6, 0), (1, 0, 0, 2), (0, 1, 0, 3), (2, 5, 0, 0)\) only the last of which is feasible in (2.3).

The problem

\[ l = \text{minimize } cx \]

\[ MAx \equiv Mb \pmod{1} \]

\[ x \geq 0 \text{ integer} \]

(2.5)

is a combinatorial one that is easy to solve, if the determinant of \(M\) is not too large. Its solution consists of finding a shortest route in a network whose nodes correspond to elements of the group generated by the columns of \(MA\) using addition modulo 1 \([7, 8]\). Its value \(l\) is a lower bound for the optimal value \(v\) of (2.1). An important choice of \(M\) is when it is taken to be the inverse of the optimal linear programming basis matrix, for then \(l\) is at least as high a bound as that gained from solving the LP (with an appropriately revised objective function).

3. The Lagrangian: A second lower bound \([9, 5]\)

Note that for any \(u \in R^n\)

\[ L(u) = \text{minimum } cx + u(Ax - b) \]

\[ MAx \equiv Mb \pmod{1} \]

\[ x \geq 0 \text{ integer} \]

(3.1)

is also a lower bound for \(v\) and hence so is

\[ w = \max_u L(u), \quad u \in R^n \]

(3.2)
$L(u)$ is easily shown to be concave and continuous and problem (3.2) may be solved using a primal-dual hill climbing technique [5].

Note how this extension fits in well with the flow chart. Having chosen a problem $P$, suppose that (2.1) is solved (or (3.1) with $u = 0$) and a lower bound found. If the answer to the question in Box 8 (should that bound be improved?) is yes, then it is simple to do so. There is no set up cost, for the group equations and corresponding network have already been calculated and as the cycle continues the lower bound is monotonically increasing. If ever the answer to Box 8 is no, then there is a valid cut available, namely

$$cx + u(Ax - b) \geq L(u).$$  \hspace{1cm} (3.3)

Hence when $P$ is partitioned we may do so into

$$P_1 = P \cap \{L(u, x) \geq L(u)\}$$

and

$$P_2 = P \cap \{L(u, x) < L(u)\}$$

$P_2$ may be removed from the list at once since it is immediately fathomable. The net result is that $P_1$ replaces $P$ in the list.

4. The Supergroup: A third lower bound [3]

What would be useful, in terms of the flow chart procedure is a method by which the lower bound for $P$ may be increased even after $L(u)$ has been maximized and the procedure of Section 3 is no longer applicable without the addition of the cut, which entails large set up costs. This may be achieved by increasing the size of the group and solving

$$A(u) = \min cx + u(Ax - b)$$

$$MAx \equiv Mb \text{ (mod } \delta)$$

$$x \geq 0 \text{ integer},$$  \hspace{1cm} (4.1)

where $\delta$ is a non-negative vector of integers.

As presented this requires a set up cost but when $M = B^{-1}$ the inverse of the optimal LP basis matrix, this set up cost may be avoided [2, 3].

The problem

$$\max_u A(u)$$

may be solved as (3.2) and since $A(u) \geq L(u)$ the bound from (4.2) will always be better. After solution of (4.2) the cut

$$cx + u(Ax - b) \geq A(u)$$

may be used as before or the size of $\delta$ may be increased. It is possible to show that

2 A. Prékopa
there exists some $\delta^*$ such that if $\delta = \delta^*$ in (4.1), (4.2) will solve the original program (2.1) that is

$$v = \max_u A^*(u).$$

5. Summary

This framework for solving integer programs was presented to give an idea to those not closely involved in integer programming where the trade-offs in computation are to be made and where research areas lie. The key is to use information that has already been calculated to improve the current lower bound without having to discard the results of long calculations. It is not especially intended that these methods are the very best that integer programming has to offer although tests at the National Bureau of Economics in the USA [8] are extremely promising.

REFERENCES