Chapter 4
Outline of an Algorithm for Integer Solutions to Linear Programs *and* An Algorithm for the Mixed Integer Problem

Ralph E. Gomory

Introduction by *Ralph E. Gomory*

Later in 1957, as the end of my three-year tour of duty in the Navy was approaching, Princeton invited me to return as Higgins Lecturer in Mathematics. I had been a Williams undergraduate and a then a graduate student at Cambridge and Princeton while getting my Ph.D. I had published 4 papers in non-linear differential equations, a subject to which I had been introduced by two wonderful people whose support and encouragement made an unforgettable and wonderful difference in my life: Professor Donald Richmond of Williams College and Professor Solomon Lefschetz of Princeton.

Because of my interest in applied work I had planned to look for an industrial position rather than an academic one on leaving the Navy, but I decided instead to accept this attractive offer and spend a year or two at Princeton before going on. When I returned to Princeton late in the fall of 1957, I got to know Professor A. W. Tucker, then the department head, who was the organizer and prime mover of a group interested in game theory and related topics. This group included Harold Kuhn and Martin (E. M. L.) Beale.

As the Navy had kept me on as a consultant I continued to work on Navy problems through monthly trips to Washington. On one of these trips a group presented a linear programming model of a Navy Task Force. One of the presenters remarked that it would be nice to have whole number answers as 1.3 aircraft carriers, for example, was not directly usable.

I thought about his remark and determined to try inventing a method that would produce integer results. I saw the problem as clearly important, indivisibilities are everywhere, but I also thought it should be possible. My view of linear programming was that it was the study of systems of linear inequalities and that it was closely anal-

Ralph E. Gomory
Alfred P. Sloan Foundation, New York, USA
e-mail: gomory@sloan.org

M. Jünger et al. (eds.), *50 Years of Integer Programming 1958-2008*,
DOI 10.1007/978-3-540-68279-0_4, © Springer-Verlag Berlin Heidelberg 2010

ogous to studying systems of linear equations. Systems of linear equations could be solved in integers (Diophantine equations), so why not systems of linear inequalities? Returning to the office I shared with Bob Gunning (later Dean of the Faculty at Princeton), I set to work and spent about a week of continuous thought trying to combine methods for linear Diophantine equations with linear programming. This produced nothing but a large number of partly worked out numerical examples and a huge amount of waste paper.

Late in the afternoon of the eighth day of this I had run out of ideas. Yet I still believed that, if I had to, in one-way or another, I would always be able to get at an integer answer to any particular numerical example. At that point I said to myself, suppose you really had to solve some particular problem and get the answer by any means, what would be the first thing that you would do? The immediate answer was that as a first step I would solve the linear programming (maximization) problem and, if the answer turned out to be 7.14, then I would at least know that the integer maximum could not be more than 7. No sooner had I made this obvious remark to myself than I felt a sudden tingling in two of my left toes, and realized that I had just done something different, and something that certainly was not a part of classical Diophantine analysis.

How exactly had I managed to conclude, almost without thought, that, if the LP answer was 7.14, the integer answer was at most 7? As I was working with equations having integer coefficients and only integer variables, it did not take long to conclude that the reasoning consisted of two simple steps. First that the objective function was maximal on the linear programming problem and therefore as large or larger than it could ever be on the integer problem. Second that the objective function was an integer linear form and therefore had to produce integer results for any integer values of the variables, including the unknown integer answer. Therefore the objective function had to be an integer less than 7.14. Clearly then it was legitimate to add an additional constraint that confined the objective function to be less than or equal to 7. I thought of this as "pushing in" the objective function. It was also immediately clear to me that there would always be many other integer forms maximal at that vertex in addition to the given objective function and that they could be "pushed in" too.

Greatly excited I set to work and within a few days had discovered how to generate maximal integer forms easily from the rows of the transformed simplex matrix. It became clear rapidly that any entry in a given row of the tableau could be changed by an integer amount while remaining an integer form, that these changes could be used to create a form that was maximal, as that simply meant that all the row entries had to become negative (in the sign convention I was then using). It also was clear that, once an entry became negative, it strengthened the new inequality if the entry was as small as possible in absolute value; so all coefficients were best reduced to their negative fractional parts. This was the origin of the "fractional cut."

Within a very few days, I had worked out a complete method using the fractional cuts. I thought of this method as the "The Method of Integer Forms." With it I was steadily solving by hand one small numerical example after another and getting the right answer. However, I had no proof of finiteness. I also observed that the fractional rows I was creating seemed to have a lot of special properties, all of which were explained later in terms of the factor group.

Just at this time I ran into Martin Beale in the hall. He was looking for a speaker for the seminar we had on game theory and linear programming. I said I would be glad to give a talk on solving linear programs in integers. Martin said "but that's impossible." That was my first indication that others had thought about the problem. During the exciting weeks that followed, I finally worked out a finiteness proof and then programmed the algorithm on the E101, a pin board computer that was busy during the day but that I could use after midnight. The E101 had only about 100 characters of memory and the board held only 120 instructions at one time, so that I had to change boards after each simplex maximization cycle and put in a new board that generated the cut, and then put the old board back to re-maximize. It was also hard work to get the simplex method down to 120 E101 instructions. But the results were better and more reliable than my hand calculations, and I was able to steadily and rapidly produce solutions to four- and five-variable problems.

During these weeks I learned that others had thought about the problem and that George Dantzig had worked on the traveling salesman problem and had applied special handmade cuts to that problem. Professor Tucker, who was enormously helpful to me during my entire stay at Princeton, gave me the time he had for himself on the program of a mathematical society meeting. There early in 1958 I made the first public presentation of the cutting plane algorithm. This produced a great deal of reaction, many people wrote to me, and Rand Corporation invited me to come out to California for the summer.

In the summer of 1958 I flew west to Los Angeles, where Rand was located, carrying the first edition of the manual for Fortran, then a brand new language. I spent one month at Rand and succeeded in producing a working Fortran version of the algorithm for the IBM 704. During my stay at Rand, I renewed my acquaintance of graduate student days with Lloyd Shapley and with Herb Scarf and met for the first time George Dantzig, Dick Bellman, and Phil Wolfe. Phil, already well known for his work on quadratic programming, generously took on the assignment of orienting me during my visit at Rand. He helped me in every conceivable way.

The Fortran program seemed to be debugged about two days before I left Rand so I was able to do larger examples. Larger meant something like ten to fifteen variables. Most of these problems ran quickly, but one went on and on and producing reams of printout but never reaching a final answer. I thought at the time that perhaps there were still bugs left in the program, but in fact it was the first hint of the computational problems that lay ahead.

It seems likely that it was during that summer that I worked out the mixed integer method, which I never sent in to a journal but appeared later as a Rand report. At the time I regarded it as a pretty straightforward extension of the original cutting plane method. Having done so many hand problems I was aware that, despite its obvious strengths in some of its computational detail it lacked some attractive properties of the all integer calculation. However at this late date I am quite reconciled to the mixed integer cut by (1) its computational success in a world of larges scale computing and (2) a rather recent result in which I have shown that it provides the only facet for the one dimensional corner polyhedron problem that is a facet both for the continuous and for the integer variables case. This finally locates the mixed cutting plane in its proper theoretical setting.

The following article originally appeared as:

R.E. Gomory, *Outline of an Algorithm for Integer Solutions to Linear Programs*, Bulletin Of the American Mathematical Society 64 (1958) 275–278.

RESEARCH ANNOUNCEMENTS

The purpose of this department is to provide early announcement of significant new results, with some indications of proof. Although ordinarily a research announcement should be a brief summary of a paper to be published in full elsewhere, papers giving complete proofs of results of exceptional interest are also solicited.

OUTLINE OF AN ALGORITHM FOR INTEGER SOLUTIONS TO LINEAR PROGRAMS

BY RALPH E. GOMORY[1]

Communicated by A. W. Tucker, May 3, 1958

The problem of obtaining the best integer solution to a linear program comes up in several contexts. The connection with combinatorial problems is given by Dantzig in [1], the connection with problems involving economies of scale is given by Markowitz and Manne [3] in a paper which also contains an interesting example of the effect of discrete variables on a scheduling problem. Also Dreyfus [4] has discussed the role played by the requirement of discreteness of variables in limiting the range of problems amenable to linear programming techniques.

It is the purpose of this note to outline a finite algorithm for obtaining integer solutions to linear programs. The algorithm has been programmed successfully on an E101 computer and used to run off the integer solution to small (seven or less variables) linear programs completely automatically.

The algorithm closely resembles the procedures already used by Dantzig, Fulkerson and Johnson [2], and Markowitz and Manne [3] to obtain solutions to discrete variable programming problems. Their procedure is essentially this. Given the linear program, first maximize the objective function using the simplex method, then examine the solution. If the solution is not in integers, ingenuity is used to formulate a new constraint that can be shown to be satisfied by the still unknown integer solution but not by the noninteger solution already attained. This additional constraint is added to the original ones, the solution already attained becomes nonfeasible, and a new maximum satisfying the new constraint is sought. This process is repeated until an integer maximum is obtained, or until some argument shows that a nearby integer point is optimal. What has been needed to transform this procedure into an algorithm is a systematic method for generating

[1] This work has been supported in part by the Princeton-IBM Mathematics Research Project.

the new constraints. A proof that the method will give the integer solution in a finite number of steps is also important. This note will describe an automatic method of generating new constraints. The proof of the finiteness of the process will be given separately.

Let us suppose that the original inequalities of the linear program have been replaced by equalities in nonnegative variables, so that the problem is to find nonnegative integers, w, $x_1, \cdots, x_m$, $t_1, \cdots, t_n$, satisfying

$$(1)\qquad \begin{aligned} w &= a_{0,0} + a_{0,1}(-t_1) \cdots a_{0,n}(-t_n),\\ x_1 &= a_{1,0} + a_{1,1}(-t_1) \cdots a_{1,n}(-t_n),\\ &\vdots\\ x_m &= a_{m,0} + a_{m,1}(-t_1) \cdots a_{m,n}(-t_n) \end{aligned}$$

such that w is maximal. Using the method of pivot choice given by the simplex (or dual simplex) method, successive pivots result in leading the above array into the standard simplex form,

$$(2)\qquad \begin{aligned} w &= a'_{0,0} + a'_{0,1}(-t'_1) \cdots a'_{0,n}(-t'_n),\\ x'_1 &= a'_{1,0} + \quad\cdots\quad \cdots a'_{1,n}(-t'_n),\\ &\vdots\\ x'_m &= a'_{m,0} + \quad\cdots\quad \cdots a'_{m,n}(-t'_n) \end{aligned}$$

where the primed variables are a rearrangement of the original variables and the $a'_{0,j}$ and $a'_{i,0}$ are nonnegative. From this array the simplex solution $t'_j=0$, $x'_i=a'_{i,0}$ is read out.

An additional constraint can now be formulated. The constraint which will be generated is not unique, but is one of a large class that can be produced by a more systematic version of the following procedure.

If the $a'_{i,0}$ are not all integers, select some i_0 with $a'_{i_0,0}$ noninteger, and introduce the new variable

$$(3)\qquad s_1 = -f'_{i_0,0} - \sum_{j=1}^{j=n} f'_{i_0,j}(-t'_j)$$

where $f'_{i_0,j}=a'_{i_0,j}-n'_{i_0,j}$, with $n'_{i_0,j}$ the largest integer $\leqq a'_{i_0,j}$. This new equation is added to the Equations (2), obtaining a new set which will be referred to as (2*). A feasible solution to (2*) is a vector, w', $x'_1, \cdots, x'_m$, $t'_1, \cdots, t'_n$, s_1 of nonnegative components. The values of $x'_1, \cdots, x'_m$, $t'_1, \cdots, t'_n$ determine the s_1 value through (3), so there is a natural correspondence between a solution

$x_1', \cdots, x_m', t_1', \cdots, t_n'$ of (2) and the (not necessarily feasible) solution that these values determine for (2*). Clearly any feasible solution to (2*) determines a feasible solution to the equations (2) simply by dropping the s_1.

It should be noted that if $f_{i_0,0}$ is $\neq 0$, then there is at least one $f_{i_0,j} \neq 0$, with $j \neq 0$, otherwise the equation

$$x_{i_0}' = a_{i_0,0}' + \sum_{j=1}^{j=n} a_{i_0,j}'(-t_j')$$

can have no solution in integers, and the program has no integer solution.

Since the simplex solution to (2), $t_j' = 0$, $x_i' = a_{i,0}'$ determines, through Equation (3), a negative value, $-f_{i_0,0}$ for s_1, the corresponding solution to (2*) is not feasible, i.e. the new restraint cuts off the old maximum. However, any nonnegative *integer* solution to (2) does give rise to a nonnegative integer solution to the equations (2*).

To see this suppose w'', x_1'', $\cdots$, x_m'', t_1'', $\cdots$, t_n'' is any solution in nonnegative integers to (2). The s_1'' determined is

$$s_1'' = -f_{i_0,0}' - \sum_{j=1}^{j=n} f_{i_0,j}'(-t_j'')$$

$$= n_{i_0,0}' + \sum_{j=1}^{j=n} n_{i_0,j}'(-t_j'') - a_{i_0,0}' - \sum_{j=1}^{n} a_{i_0,j}'(-t_j'')$$

which using (2) becomes $s_1'' = n_{i_0,0}' + \sum_{j=1}^{n} n_{i_0,j}'(-t_j'') - x_{i_0}''$. Since the $n_{i_0,j}'$, the t_j'' and the x_{i_0}'' are all integers, the s_1'' determined is also an integer. Furthermore, since the $f_{i_0,j}'$ and the t_j'' are all nonnegative, (3) shows that s_1'' is $\geqq -f_{i_0,0}' > -1$. Since s_1'' is an integer, this shows it must be nonnegative.

This reasoning establishes a one-one correspondence between nonnegative integer solutions w'', x_1'', $\cdots$, x_m'', t_1'', $\cdots$, t_n'' to (2) and the corresponding nonnegative integer solutions w'', x_1'', $\cdots$, x_m'', t_1'', $\cdots$, t_n'', s_1'' to (2*). Since the w value is the same for both solutions, the problem of maximizing w over nonnegative integer solutions to (2) can be replaced by the problem of maximizing w over the nonnegative integer solutions to (2*). The solution to the original problem is obtained by dropping the s_1.

The procedure now is to maximize w over the solutions to (2*). This is done using the dual simplex method because all the $a_{0,j}'$ and $a_{i,0}'$ are already nonnegative, and $-f_{i_0,0}$ is the only negative entry in the zero column of the equations (2*). This fact usually makes remaximization quite rapid. The process is then repeated if the new simplex maximum is noninteger.

Of course the Equations (2*) involve one more equation than the Equations (2), and an equation is added after each remaximization. However, the total number need never exceed $m+n+2$. For if an s-variable, added earlier in the computation reappears among the variables on the left hand side of the equations after some remaximization, the equation involving it can simply be dropped, as the only equations that need be satisfied are the original ones. This limits the total number of s-variables to $n+1$ or less.

It should be noted that even the process just described involves an element of choice, any of the rows i of (2) with $a_{i,0}$ noninteger might be chosen to generate the new relation. Some choices are better than others. A good rule of thumb based on the idea of "cutting" as deeply as possible with the new relation, and borne out by limited computational experience, is to choose the row with the largest fractional part $f_{i,0}$ in the zero column.

The class of possible additional constraints is not limited to those produced by the method described here since it is easily seen that some simple operations on and between rows preserve the properties needed in the additional relations. These operations can be used to produce systematically a family of additional relations from which a particularly effective cut or cuts can be selected. A discussion of this class of possible additional constraints together with a rule of choice of row which can be shown to bring the process to an end in a finite number of steps—thus providing a finite algorithm—require some space and will be given as part of a more complete treatment in another place.

References

1. George B. Dantzig, *Discrete-variable extremum problems*, J. Operations Res. Soc. Amer. vol. 5, no. 2 (1957).

2. G. Dantzig, R. Fulkerson, and S. Johnson, *Solution of a large-scale traveling-salesman problem*, J. Operations Res. Soc. Amer. vol. 2, no. 4 (1954).

3. Harry M. Markowitz and Alan S. Manne, *On the solution of discrete programming problems*, Econometrica vol. 25, no. 1 (1957).

4. Stuart E. Dreyfus, *A comparison of linear programming and dynamic programming*, Rand Report P-885, June, 1956.

Princeton University

The following article originally appeared as:

R.E. Gomory, *An Algorithm for the Mixed Integer Problem*, Research Memorandum RM-2597, The Rand Corporation, 1960.

U. S. AIR FORCE
PROJECT RAND

RESEARCH MEMORANDUM

AN ALGORITHM FOR THE
MIXED INTEGER PROBLEM

Notes on Linear Programming
and Extensions—Part 54

Ralph Gomory

RM-2597

7 July 1960

Assigned to ______________________

This research is sponsored by the United States Air Force under contract No. AF 49(638)-700 monitored by the Directorate of Development Planning, Deputy Chief of Staff, Development, Hq USAF.

The RAND Corporation
1700 MAIN ST. • SANTA MONICA • CALIFORNIA

SUMMARY

An algorithm is given for the numerical solution of the "mixed integer" linear programming problem, the problem of maximizing a linear form in finitely many variables constrained both by linear inequalities and the requirement that a proper subset of the variables assume only integral values. The algorithm is an extension of the cutting plane technique for the solution of the "pure integer" problem.

CONTENTS

SUMMARY... ii

TEXT.. 1

REFERENCES.. 12

LIST OF RAND NOTES ON LINEAR PROGRAMMING AND EXTENSIONS.... 13

RM-2597
1

AN ALGORITHM FOR THE MIXED INTEGER PROBLEM

The problem discussed here is an integer programming problem, i.e., the problem of maximizing

$$z = a_{o,o} + \sum_{j=1}^{j=n} a_{o,j}(-t_j),$$

subject to the inequalities

$$(1) \qquad \sum_{j=1}^{j=n} a_{i,j}t_j \leq a_{i,o}, \qquad i = 1, \ldots, m$$

and subject to the additional condition that some specified subcollection of the variables appearing above should be integers.

If the inequalities above are changed into equations in nonnegative variables by the addition of m "slack" variables, and the whole set is enlarged to form a set in which all the variables are expressed in terms of the independent or "nonbasic" ones, we have

$$z = a_{o,o} + \sum_{j=1}^{j=n} a_{o,j}(-t_j)$$

$$s_i = a_{i,o} + \sum_{j=1}^{j=n} a_{i,j}(-t_j) \qquad i = 1, \ldots, m$$

$$t_j = -1(-t_j) \qquad j = 1, \ldots, n.$$

RM-2597
2

For the sake of a more uniform notation we will rewrite this as

$$(2) \qquad x_i = a_{i,o} + \sum_{j=1}^{j=n} a_{i,j}(-t_j) \qquad i = 0, \ldots, m+n,$$

where the x_i now are <u>all</u> the variables and the $a_{i,j}$ are <u>all</u> the coefficients.

The usual[1] linear programming problem is solved by applying G. B. Dantzig's simplex method. In this method a series of "pivot steps," "Gaussian eliminations," "changes of basis," or "changes to different sets of nonbasic variables" bring the equations (2) into a form in which, denoting the new coefficients in the equations by primes,

$$(i) \quad a'_{i,o} \geq 0 \qquad i = 1, \ldots, m+n$$

and

$$(ii) \quad a'_{o,j} \geq 0 \qquad j = 1, \ldots, n.$$

The first condition is the condition that in the "trial solution" obtained by putting all the nonbasic variables equal to zero, the values that result for all the variables are nonnegative. The second condition makes certain that the objective function is in fact maximal when the variables are given the values they attain in this trial solution. The solution obtained is of course

[1]The usual method terminates when conditions (ii) first hold. It is necessary here that the pivoting continue until all columns $j > 0$ become lexicographically positive. The procedure for doing this is described in [1].

RM-2597
3

$$x_i = a'_{i,o} \qquad i = 0, \ldots, m+n.$$

This solution may very well not satisfy the integer requirement; i.e., some x_i that is required to be an integer is assigned the noninteger value $a'_{i,o}$.

If this occurs we will be able to deduce a new inequality that will be satisfied by any integer solution, i.e., by any solution having integers where they are required, but will not be satisfied by the current trial solution.

Then, just as in [1] and [2], this new inequality will be added to the original set of inequalities, and the new set then remaximized by the simplex method. This remaximization is usually quite rapid, as adding the new inequality maintains dual feasibility and introduces just the one unsatisfied inequality.

If the new maximum solution still contains integer variables which are assigned noninteger values the process is repeated.

To deduce this new inequality we make use of the equation

$$(3) \qquad x_i = a'_{i,o} + \Sigma\, a'_{i,j}(-t_j)$$

where the x_i is an integer variable, $a'_{i,o}$ is noninteger, and the t_j are the current set of nonbasic variables. Since $a'_{i,o}$ is noninteger it can be written uniquely as the sum of an integer $n'_{i,o}$ and a fractional part $f'_{i,o}$, $0 < f'_{i,o} < 1$.

RM-2597
4

We now imagine that we have an integer solution to the problem and use x_i', t_j' to denote the values given to the variables in (3) by this solution. Hence

$$x_i' = a_{i,o}' + \Sigma\, a_{i,j}'(-t_j')$$

and using $a \equiv b$ to mean a and b differ by an integer (equivalence modulo 1), we have, since $x_i' \equiv 0$ and $a_{i,o}' \equiv f_{i,o}'$,

$$(4) \qquad \Sigma\, a_{i,j}' t_j' \equiv f_{i,o}' \; .$$

We will group the constants on the left in (4) according to their sign. Let S^+ be the set of indices j for which $a_{i,j}' \geq 0$, and S^- the set for which $a_{i,j}' < 0$. Then

$$(5) \qquad \sum_{j\epsilon S^+} a_{i,j}' t_j' + \sum_{j\epsilon S^-} a_{i,j}' t_j' \equiv f_{i,o}' .$$

There are now two possibilities to consider. The expression on the left is either (i) nonnegative or (ii) negative.

Case (i). Since the left side is nonnegative and equivalent to f_o, its value can only be f_o', or $1 + f_o'$, or $2 + f_o'$, etc. Hence

$$f_{i,o}' \leq \sum_{j\epsilon S^+} a_{i,j}' t_j' + \sum_{j\epsilon S^-} a_{i,j}' t_j' \leq \sum_{j\epsilon S^+} a_{i,j}' t_j' .$$

Case (ii). If the right-hand side is negative and equivalent to $f'_{i,o}$ it can only be $f_o - 1$, $f_o - 2$, etc. So in every case

$$f'_{i,o} - 1 \geq \sum_{j \in S^+} a'_{i,j} t'_j + \sum_{j \in S^-} a'_{i,j} t'_j \geq \sum_{j \in S^-} a'_{i,j} t'_j ,$$

or, multiplying by $- f'_{i,o}/1 - f'_{i,o}$,

$$f'_{i,o} \leq \sum_{j \in S^-} \frac{f'_{i,o}}{1 - f'_{i,o}} (-a'_{i,j}) t'_j \ .$$

Now either (i) holds or (ii) holds so always

$$(6) \qquad f'_{i,o} \leq \sum_{j \in S^+} a'_{i,j} t'_j + \sum_{j \in S^-} \frac{f'_{i,o}}{1 - f'_{i,o}} (-a'_{i,j}) t'_j ,$$

since the right side is the sum of two nonnegative numbers, one of which is $\geq f'_{i,o}$.

This inequality then is satisfied by any integer solution but not by the present trial solution, since substituting $t_j = 0$ for all j into (6) makes the right-hand side 0.

Of course the inequality (6) can be rewritten as an equation by introducing a nonnegative slack s. Then (6) becomes

$$s = -f'_{i,o} - \sum_{j \in S^+} a'_{i,j}(-t_j) - \sum_{j \in S^-} \frac{f'_{i,o}}{1 - f'_{i,o}} (-a'_{i,o})(-t_j).$$

In obtaining (6) we have used only the fact that x_i was required to be an integer. If some of the nonbasic variables t_j are also integer variables, the inequality (6) can be improved in a manner entirely analogous to the reduction that is always possible in the strictly integer problem. The improvement will take the form of a decrease in the coefficients on the right in the resulting inequality (6). It is clear that for fixed $f'_{i,o}$ the smaller these coefficients, the stronger the inequality.

Let us suppose then that some t_{j_o} is required to be integer and hence is assigned an integer value t'_{j_o} in (5). Changing a'_{i,j_o} by an integer amount then changes the left side of (5) by an integer, and hence preserves the equivalence. Thus we may replace a'_{i,j_o} by any new value $a^* \equiv a_{i,j_o}$ and proceed just as before to deduce an inequality like (6).

If $a^* \geq 0$, the coefficient of t_{j_o} in the resulting inequality is simply a^*. If a^* is < 0, it is $-f'_{i,o}/(1 - f'_{i,o})a^*$. Among $a^* \geq 0$, $a^* = f'_{i,j_o}$ the fractional part[1] of a'_{i,j_o} clearly gives the smallest coefficient to t_{j_o} in the resulting inequality. (This may even be 0.) Among $a^* < 0$, the smallest coefficient is obtained from $a^* = f'_{i,j_o} - 1$, and is

$$\text{(7)} \qquad \frac{f'_{i,o}}{1 - f'_{i,o}} (1 - f'_{i,j_o}).$$

[1] By the fractional part of both positive and negative numbers $a_{i,j}$ we will mean the nonnegative fraction $f_{i,j} < 1$ such that $a_{i,j} = n_{i,j} + f_{i,j}$ with $n_{i,j}$ integer.

To obtain the smallest possible coefficient we choose the smaller of f'_{i,j_o} and (7) which, because an expression of the form $x/1 - x$ increases monotonically with x is seen to be

$$f'_{i,j_o} \text{ if } f'_{i,j_o} \leq f'_{i,o}$$

and

$$\frac{f'_{i,o}}{1 - f'_{i,o}}\left(1 - f'_{i,\ j_o}\right) \text{ if } f'_{i,j_o} > f'_{o,o}.$$

It follows that the strongest inequality is obtained by a simple two-stage process. (i) First replace coefficients of integer variables by their fractional parts if these are less than $f'_{i,o}$, or by the fractional parts less 1 if they are greater than $f'_{i,o}$. (ii) Then deduce the inequality (6) as before. The final result obtained from the equation

$$x_i = a'_{i,o} + \Sigma\, a'_{i,j}(-t_j)$$

by this procedure is the inequality represented by the equation

$$s = -f'_{i,o} - \Sigma\, f^*_{i,j}(-t_j) \tag{8}$$

where the $f^*_{i,j}$, all nonnegative, are given by the following formulae:

RM-2597
8

$$f^*_{i,j} = \begin{cases} a'_{i,j} & \text{if } a'_{i,j} \geq 0 \text{ and } t_j \text{ noninteger variable} \\ \dfrac{f'_{i,o}}{1 - f'_{i,o}}(-a'_{i,j}) & \text{if } a'_{i,j} < 0 \text{ and } t_j \text{ noninteger variable} \\ f'_{i,j} & \text{if } f'_{i,j} \leq f'_{i,o} \text{ and } t_j \text{ integer variable} \\ \dfrac{f'_{i,o}}{1 - f'_{i,o}}(f'_{i,j}-1) & \text{if } f'_{i,j} > f'_{i,o} \text{ and } t_j \text{ integer variable} \end{cases}$$

Equation (8) is then added and the problem is remaximized. It seems sensible to use the dual simplex method at this point as all the $a'_{o,j}$, $j \geq 1$, are nonnegative, and there is only one negative element, $-f'_{i,o}$, in the 0-column.

If the dual simplex method is applied, the 0-column is decreased lexicographically at the next step, and furthermore, denoting by double primes the coefficients after the next pivot step and by j_o the column in which the pivot step takes place, we have

$$(9) \qquad \begin{aligned} a''_{i,o} &\leq n'_{i,o} && \text{if } a'_{i,j_o} > 0 \\ a''_{i,o} &\geq n'_{i,o} + 1 && \text{if } a'_{i,j_o} < 0 \end{aligned}$$

where $n'_{i,o}$ is the integer part of $a'_{i,o}$, the index i in (9) is that of the row figuring in equations (3) through (8).

This means that after the next pivot step the value assigned to x_i by the new trial solution is either $\geq$ the next highest integer, or $\leq$ the next lowest integer.

To see this we consider the mechanism of the dual simplex method. The dual simplex method will pick a pivot in the new row represented by (8). If the pivot element is chosen in this row and in the j_o column then the formula for the $a''_{i,o}$ that results after a pivot step is

$$a''_{i,o} = a'_{i,o} - \frac{f'_{i,o} a_{i,j_o}}{f^*_{i,j_o}} .$$

Now the formulas for $f'_{i,j}$ show that if a_{i,j_o} is positive and t_{j_o} noninteger we have

$$a''_{i,o} = a'_{i,o} - \frac{f'_{i,o} a'_{i,j_o}}{a'_{i,j_o}} = n'_{i,o} . \tag{10}$$

If a'_{i,j_o} is negative and t_{j_o} noninteger we have

$$a''_{i,o} = a'_{i,o} - \frac{f'_{i,o} a'_{i,j_o}}{\left(\dfrac{f'_{i,o}}{1 - f'_{i,o}}\right)\left(-a'_{i,j_o}\right)} \tag{11}$$

$$= a'_{i,o} - f'_{i,o} + 1 = n'_{i,o} + 1 .$$

RM-2597
10

To cover the cases when t_{j_o} is an integer variable we need only remember that in this case the f^*_{i,j_o} is deduced by a two-stage process, part (ii) of which is exactly the same as the process used to deduce the f^*_{i,j_o} when t_{j_o} is noninteger. Consequently if part (i) leaves $a'_{i,j}$ unchanged, either (10) or (11) holds just as above. Part (i) will have $a'_{i,j}$ unchanged only if either

$$a'_{i,j} = f'_{i,j}, \qquad \text{and } f'_{i,j} \leq f'_{i,o}$$

or

$$a'_{i,j} < 0,\ a'_{i,j} = f'_{i,j} - 1, \quad \text{and } f'_{i,j} > f'_{i,o}$$

Otherwise part (i) makes a change which results in a strictly smaller final $f^*_{i,j}$. So in these cases we have the strict inequalities

$$a''_{i,o} < n'_{i,o} \qquad \text{if } a'_{i,j_o} > 0$$

$$a''_{i,o} > n'_{i,o} + 1 \qquad \text{if } a'_{i,j_o} < 0.$$

The remaining possibility, $a'_{i,j_o} = 0$, can not occur because $a'_{i,j_o} = 0$ implies $f^*_{i,j_o} = 0$ and so f^*_{i,j_o} can not be the pivot element. Thus (9) holds in all cases.

Now (9) is exactly the property required for a finiteness proof—i.e., a proof that the solution is attained in a finite number of steps—<u>provided that the objective function</u> z <u>is one of the integer variables</u>. To see this we arrange the original equations so that the integer variables on the left in (2) are the first rows following the objective function z. (This means that they rank higher lexicographically in the dual simplex method.) Given property (9), the reasoning in the first finitness proof in [1] (pp. 33–35) now goes through unchanged. Of course one must stop now on attaining the required integer values in the 0-column, as an all-integer matrix is not generally obtained.

RM-2597
12

REFERENCES

1. Gomory, Ralph E., "An Algorithm for Integer Solutions to Linear Programs," Princeton-IBM Mathematics Research Project Technical Report No. 1, November 17, 1958.

2. Beale, E. M. L., "A Method of Solving Linear Programming Problems When Some But Not All of the Variables Must Take Integral Values," Statistical Techniques Research Group, Princeton University (manuscript).