AD 742691

AD 742691

Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) UNIVERSITY OF FLORIDA	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP Unclassified

3. REPORT TITLE

ON THE INTEGER SOLUTION OF THE HYPERBOLIC PROGRAMMING PROBLEM

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

5. AUTHOR(S) (Last name, first name, initial)

GRUNSPAN, Marcel, THOMAS, Michael E.

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
February 1972	85	36

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
DAH C04 68C0002 b. PROJECT NO. 2T0 14501B81B c. d.	Technical Report No. 61 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. AVAILABILITY/LIMITATION NOTICES

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	ARO-Durham

13. ABSTRACT

The problem considered is that of finding an optimal integer solution for the hyperbolic programming problem. A geometrical framework for viewing the problem is developed and a general algorithm for finding an optimal integer solution is proposed. This algorithm reduces to solving a finite sequence of linear integer programs when the number of feasible integer points is finite. It is shown that when the integer restriction is removed, the general algorithm reduces to an algorithm proposed by Isbell and Marlow to solve the continuous hyperbolic program. It is also shown that the group theoretic approach to integer programming can be used for hyperbolic integer programming. Solutions for a hyperbolic programming problem with bounded integer variables only and a hyperbolic knapsack problem are also given. It is shown that using the general algorithm to solve these problems makes it possible to reduce the number of variables at each iteration.

ON THE INTEGER SOLUTION OF THE HYPERBOLIC PROGRAMMING PROBLEM

Technical Report No. 61

SUBMITTED BY:

Marcel Grunspan
Research Associate

M. E. Thomas
Principal Investigator

APPROVED BY:

B. D. Sivazlian
Editor, Project THEMIS

Department of Industrial and Systems Engineering
The University of Florida
Gainesville, Florida 32601

February 1972

This research was performed under Project THEMIS, ARO-D Contract No. DAH C04 68C0002.

ABSTRACT

The problem considered is that of finding an optimal integer solution for the hyperbolic programming problem. A geometrical framework for viewing the problem is developed and a general algorithm for finding an optimal integer solution is proposed. This algorithm reduces to solving a finite sequence of linear integer programs when the number of feasible integer points is finite. It is shown that when the integer restriction is removed, the general algorithm reduces to an algorithm proposed by Isbell and Marlow to solve the continuous hyperbolic program. It is also shown that the group theoretic approach to integer programming can be used for hyperbolic integer programming. Solutions for a hyperbolic programming problem with bounded integer variables only and a hyperbolic knapsack problem are also given. It is shown that using the general algorithm to solve these problems makes it possible to reduce the number of variables at each iteration.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

CHAPTER 1

INTRODUCTION

1.1 The Hyperbolic Programming Problem and Its Origins

Mathematical programs whose objective functions can be expressed as ratios of functionals are commonly called fractional programs. A special case belonging to this class of problems is the hyperbolic programming problem.[1] It has an objective function which is the ratio of linear functionals and is of the following form:

$$\begin{aligned} &\max \{f(\underline{x}') = (\underline{c}'\underline{x}' + \alpha)/(\underline{d}'\underline{x}' + \beta)\} \\ &\text{s.t.} \quad A'\underline{x}' \leq \underline{b} \\ &\qquad \underline{x}' \geq 0 \end{aligned} \tag{P_C'}$$

To avoid pathological cases, it will be assumed that (P_C') is defined over the field of rationals. The constraint matrix A' has dimension $(m \times n)$, $\underline{x}'$ is an n dimensional column vector, $\underline{c}'$ and $\underline{d}'$ are n dimensional row vectors, $\underline{b}$ is an m dimensional column vector and α and β are fixed constants.

Applications of (P_C') arise in a wide variety of contexts. Almogy and Levin [1] note that a fractional objective function can represent a time rate of earnings and is therefore a true measure of economic performance. Martos [2] recognizes the appropriateness of

---

[1] This problem is also known as the linear fractional programming problem.

the structure of (P'_C) when the purpose of optimization is finding the most favorable ratio of revenues and allocations.

Dantzig et al. [3] construct a hyperbolic program when solving a ship routing problem. Their solution requires finding a cycle in a graph which has a minimal cost to time ratio. This problem was also studied by Fox [4].

Production models in which the manufacturing processes generate scrap material also suggest problems that resemble (P'_C). For example, in their discussion of the cutting stock problem, Gilmore and Gomory [5] minimize a rational objective function which represents percentage waste. Wagner [6] considers the related problem of finding the production schedule which maximizes the fraction of usable raw material.

Charnes and Cooper [7] generate (P'_C) when dealing with system evaluation and repricing problems. In this context, the objective function is the ratio of the total change in cost to changes in volume that accompany possible variations of a particular cost coefficient.

Other applications arise in the areas of optimal maintenance of equipment [8], Markov renewal programming [9], and routing problems [1].

The solution to (P'_C) is well documented and is briefly reviewed in Chapter 2. Although the more general fractional programming problem is not considered here, it has also received considerable attention. Dinkelbach [10] studied convex-concave ratios, Bector [11] recognized those ratios which are convex, and Manganarian [12]

noted that under certain restrictions the Frank and Wolfe algorithm can be used to optimize a convex-concave ratio. A bibliography on fractional programming can be found in Grunspan [13].

1.2 Hyperbolic Programming With Integer Variables

Integer variable restrictions are just as important in hyperbolic programming formulations as in linear programming.[1] For example, consider an investor faced with the problem of determining how many shares in each of n investments he should purchase at the beginning of a time period so that at the end of the time period, the ratio of his holdings to his cost is a maximum. Assuming that the constraints on the funds available for investment are linear inequalities and letting

c_i-value of one share of investment i at end of time period

d_i-cost of one share of investment i at beginning of time period.

α-guaranteed return

β-fixed cost

x_i-number of shares of investment i

the optimal investment policy can be obtained by solving the following problem:

---

[1] Economic analysis problems which deal with indivisible commodities are always more realistically formulated as integer programs. A wealth of other examples can be found in Balinski [14].

$$\max \{f(\underline{x}') = (\sum_{i=1}^{n} c_i' x_i' + \alpha)/(\sum_{i=1}^{n} d_i' x_i' + \beta)\}$$

$$\text{s.t.} \quad A'\underline{x}' \leq \underline{b}$$

$$\underline{x}' \geq 0 \quad \text{and integer.}$$

Certain routing problems where the objective is to minimize the cost to time ratio can also be formulated as hyperbolic integer programs [1].

Dantzig [15] has shown that integer variables also arise indirectly. For example, the condition that a variable x can only assume one of a discrete set of values can easily be expressed with the aid of (0-1) variables. Integer variables of the (0-1) type are also useful when it is necessary to express the condition that only k out of m constraints must be satisfied.

The importance of the integer restriction in hyperbolic programming leads to problem (P_C') with the additional restriction that the solution vector $\underline{x}'$ be all integer. This problem will be denoted by (P_I'). Since (P_C') is defined over the field of rationals, there is no loss in generality in assuming that in (P_I') the entries in A', $\underline{c}'$ and $\underline{d}'$ and $\underline{b}$ as well as the fixed constants α and β are all integers.

An alternate formulation for (P_I') is obtained by adding slack variables to the constraint equations. This leads to a problem in a space of higher dimension which is of the following form:

$$\max \{f(\underline{x}) = (\underline{c}\,\underline{x} + \alpha)/(\underline{d}\,\underline{x} + \beta)\}$$

$$\text{s.t.} \quad A\underline{x} = \underline{b}$$

$$\underline{x} \geq 0 \quad \text{and integer} \qquad (P_I)$$

In this formulation A is an m x (m + n) integer matrix, $\underline{x}$ is an (m + n) dimensional integer column vector, $\underline{c}$ and $\underline{d}$ are (m + n) dimensional integer row vectors, $\underline{b}$ is an m dimensional integer column vector and α and β are fixed integers.

Problems (P_I') and (P_I) are alternate statements of the general hyperbolic integer programming problem. Solving (P_I) is equivalent to solving (P_I'). The objective of this dissertation is to develop a class of algorithms for finding the optimal solutions for (P_I) or (P_I').

Throughout the remainder of this paper, it will be assumed that the set of feasible solutions for (P_I') is bounded and that the value of the denominator in the objective function is strictly greater than zero for all feasible solutions. These assumptions are quite realistic in an economic context. Respectively they imply that the levels of activities being programmed cannot be unbounded and that the objective function value cannot become infinite.

CHAPTER 2

CONTINUOUS HYPERBOLIC PROGRAMMING

2.1 The Isbell-Marlow Algorithm

The general algorithm discussed in Chapter 3 can be viewed as a generalization of an algorithm originally proposed by Isbell and Marlow [16] for solving the continuous hyperbolic program. To motivate the presentation which follows, it will be useful to review the central idea behind this algorithm. Let $\overline{X}' = \{\underline{x}' | A'\underline{x}' \leq \underline{b}, \underline{x}' \geq 0\}$, and let $\underline{x}'_a$ be any point such that $(\underline{d}'\underline{x}'_a + \beta) \neq 0$. First suppose that the point $\underline{x}'_a$ is feasible for (P'_C) and consider the following linear programming problem:

$$\max \{z(\underline{x}') = (\underline{d}'\underline{x}'_a + \beta)(\underline{c}'\underline{x}' + \alpha) - (\underline{c}'\underline{x}'_a + \alpha)(\underline{d}'\underline{x}' + \beta)\}$$

$$\text{s.t.} \quad \underline{x}' \in \overline{X}'$$

Since the point $\underline{x}'_a$ is feasible for this problem, it follows that $\max_{\underline{x}' \in \overline{X}'} z(\underline{x}') \geq 0$. Thus if $\underline{x}'_1$ is an optimal solution for the linear program, the condition

$$z(\underline{x}'_1) = (\underline{d}'\underline{x}'_a + \beta)(\underline{c}'\underline{x}'_1 + \alpha) - (\underline{c}'\underline{x}'_a + \alpha)(\underline{d}'\underline{x}'_1 + \beta) \geq 0$$

together with the assumption that $(\underline{d}'\underline{x}' + \beta) > 0 \ \forall \underline{x}' \in \overline{X}'$ implies that

$$f(\underline{x}'_1) = \frac{(\underline{c}'\underline{x}'_1 + \alpha)}{(\underline{d}'\underline{x}'_1 + \beta)} \geq \frac{(\underline{c}'\underline{x}'_a + \alpha)}{(\underline{d}'\underline{x}'_a + \beta)} = f(\underline{x}'_a)$$

Hence given a feasible point $\underline{x}'_a$ for (P'_C), it is always possible to formulate a linear program whose solution yields an objective function value for (P'_C) which is at least as good as $f(\underline{x}'_a)$.

Interestingly enough, even if $\underline{x}'_a$ is not feasible for (P'_C), the solution to the linear program is still feasible for (P'_C). In this case however, it is possible that $f(\underline{x}'_1) < f(\underline{x}'_a)$. Since $\underline{x}'_a$ is not feasible, this is of no consequence. The important thing is that if a feasible point is not immediately available, one can always be obtained by solving a linear program. Once a feasible point is at hand, it is possible to formulate a second linear program which yields an objective function value for (P'_C) which is at least as good as the current value. Independent of the feasibility of $\underline{x}'_a$, the existence of a finite solution for the linear program is insured if $\overline{X}'$ is assumed bounded.

The result of this analysis is the Isbell-Marlow algorithm. It is a procedure for solving (P'_C) which reduces to solving a finite sequence of linear programming problems.

2.2 An Extension of the Isbell-Marlow Algorithm

In this section, it will be shown that hyperbolic programming problems other than (P'_C) can be solved using an approach similar to the one described in Section 2.1.

Consider the problem

$$\max \{f(\underline{x}') = (\underline{c}'\underline{x}' + \alpha)/(\underline{d}'\underline{x}' + \beta)\}$$

$$\text{s.t. } \underline{x}' \in F \qquad (P'_F)$$

where the feasible set F consists of a finite number of points. Repeating the argument used to develop the Isbell-Marlow algorithm for (P'_C), if $\underline{x}'_a$ is any feasible point for (P'_F), then an improved solution for (P'_F) can be obtained by considering the following problem:

$$\max \{z(\underline{x}') = (\underline{d}'\underline{x}'_a + \beta)(\underline{c}'\underline{x}' + \alpha) - (\underline{c}'\underline{x}'_a + \alpha)(\underline{d}'\underline{x}' + \beta)\}$$
$$\text{s.t.} \quad \underline{x}' \in F \qquad (LP'_F)$$

The next theorem gives a necessary and sufficient condition for the point $\underline{x}'_a$ to be optimal for (P'_F).

Theorem 1

The point $\underline{x}'_a \in F$ maximizes the hyperbolic objective function in (P'_F) iff $\underline{x}'_a \in F$ maximizes the objective function in (LP'_F).

Proof:

First suppose that

$$\max_{\underline{x}' \in F} f(\underline{x}') = f(\underline{x}'_a) \qquad (2.1)$$

Assuming that $(\underline{d}'\underline{x}' + \beta) > 0 \ \forall \underline{x}' \in F$, (2.1) implies that

$$(\underline{d}'\underline{x}'_a + \beta)(\underline{c}'\underline{x}' + \alpha) - (\underline{c}'\underline{x}'_a + \alpha)(\underline{d}'\underline{x}' + \beta) \leq 0 \quad \forall \underline{x}' \in F \qquad (2.2)$$

The left-hand side in (2.2) is clearly the objective function in (LP'_F). By inspection, it achieves its upper bound at the point $\underline{x}'_a$. The point $\underline{x}'_a$ is therefore optimal for (LP'_F).

To prove the second part of the Theorem, suppose that

$$\max \{z(\underline{x}')\} = 0$$

$$\text{s.t.} \quad \underline{x}' \in F$$

This implies that

$$z(\underline{x}') \leq 0 \quad \forall \underline{x}' \in F$$

Since $(\underline{d}'\underline{x}' + \beta) > 0 \quad \forall \underline{x}' \in F$, the above inequality implies that

$$f(\underline{x}') \leq f(\underline{x}'_a) \quad \forall \underline{x}' \in F$$

The point $\underline{x}'_a$ is therefore an optimal solution for (P'_F).

Corollary 1.1

The point $\underline{x}'_a$ which maximizes (LP'_F) is optimal for (P'_F) iff $z(\underline{x}'_a) = 0$.

Proof:

This result is a direct consequence of the proof given for Theorem 1.

Theorem 1 and Corollary 1.1 together with the observation that $\max z(\underline{x}') \geq 0$ suggests that the solution to (P'_F) can be obtained by $\underline{x}' \in F$ solving a finite sequence of linear optimization problems.

2.3 Alternate Solution Approaches for (P'_C)

Given that $(\underline{d}'\underline{x}' + \beta) > 0 \quad \forall \underline{x}' \in \overline{X}'$, Martos [2] proved that the solution to (P'_C) must lie on at least one extreme point of the

feasible set $\bar{X}'$. Based on this result, he developed an extreme point search very similar to the simplex algorithm of linear programming. The Martos algorithm is fundamental to the cutting plane methods discussed in Chapter 3. It will be reviewed later as part of the development for the cutting plane algorithms.

Another approach for solving (P_C') which is in the spirit of Lemke's dual simplex algorithm is due to Dorn [17]. In this algorithm it is important that the original problem be treated as the dual problem. An extreme point of the dual feasible set (the original feasible set) is chosen and the current values of the primal variables (the Lagrange multipliers) are computed. If they all have the proper sign, the current solution is optimal. If they do not, an adjacent extreme point of the dual feasible set which improves the objective function value is chosen and the procedure is repeated. Since the dual feasible region has a finite number of extreme points and since a local maximum for (P_C') is also a global maximum, the algorithm terminates in a finite number of steps. Dorn also points out that the problems of finding an initial extreme point solution and of degeneracy are identical with those same problems for linear programs. Remedies for these problems can be found in the appendices of [18].

Another alternative for solving (P_C') is proposed by Charnes and Cooper [19]. By means of a simple transformation, the hyperbolic program is transformed to a linear program. If the sign of the objective function value is known at optimality, only one linear program need be solved. Otherwise, it is necessary to solve two

linear programs. Unfortunately, due to the nature of the transformation, finding the integer solution to the Charnes and Cooper linear program does not guarantee that the integer solution to (P'_C) is directly available.

2.4 The Geometry of the Hyperbolic Program

This section is devoted to the geometrical aspects of hyperbolic programming. The ideas developed here will also be useful in describing geometrically the algorithms in Chapter 3.

Corresponding to any sequence of hyperbolic functional values $\{f(\underline{x}'_1),\ldots,f(\underline{x}'_n)\}$ with the property that $\underline{x}'_i \in \overline{X}'$ $\forall i$ and $f(\underline{x}'_i) < f(\underline{x}'_{i+1})$ $(i=1,\ldots,n-1)$ there is a unique sequence of hyperplanes $\{T^o_1,\ldots,T^o_n\}$ where

$$T^o_i = \{\underline{x}' \mid (\underline{d}'\underline{x}'_i + \beta)(\underline{c}'\underline{x}' + \alpha) - (\underline{c}'\underline{x}'_i + \alpha)(\underline{d}'\underline{x}' + \beta) = 0\}$$

$$i=1,\ldots,n \qquad (2.3)$$

The hyperplanes T^o_i can be seen to intersect about the solution set of the following set of equations:

$$\begin{aligned} \underline{c}'\underline{x}' + \alpha &= 0 \\ \underline{d}'\underline{x}' + \beta &= 0 \end{aligned} \qquad (2.4)$$

The solution set for (2.4) is also the set

$$\ker(T) = \{\underline{x}' \mid (\underline{c}'\underline{x}' + \alpha = 0,\ \underline{d}'\underline{x}' + \beta = 0,\ \underline{x}' \in E^n\}$$

which is the kernel of the map defined by

$$T(\underline{x}') = (\underline{c}'\underline{x}' + \alpha\ ,\ \underline{d}'\underline{x}' + \beta) \qquad \underline{x}' \in E^n$$

The first component of the image of $\underline{x}'$ under the map T corresponds to the numerator and the second component corresponds to the denominator of the hyperbolic objective function.

Having introduced the set, kern(T), a foundation for the geometric description of the behavior of the hyperbolic objective function in E^n is developed.

A trivial hyperbolic program occurs when the functionals $(\underline{c}'\underline{x}' + \alpha)$ and $(\underline{d}'\underline{x}' + \beta)$ are linearly dependent. It is easy to see that for such a problem, the objective function value remains constant $\forall \underline{x}' \in E^n$. To avoid this case, it will always be assumed that $(\underline{c}'\underline{x}' + \alpha)$ and $(\underline{d}'\underline{x}' + \beta)$ are linearly independent. This assumption also insures that the set kern(T) is nonempty.

Lemma 1

The set kern(T) $\neq \phi$ if the functional $(\underline{c}'\underline{x}' + \alpha)$ is linearly independent of the functional $(\underline{d}'\underline{x}' + \beta)$.

Proof:

This result is a consequence of the fact that the map T is onto if $(\underline{c}'\underline{x}' + \alpha)$ and $(\underline{d}'\underline{x}' + \beta)$ are linearly independent. Suppose $(y_1, y_2) \in E^2$ is chosen arbitrarily. The map T is onto if there exist $\underline{x}'_* \in E^n$ such that

$$y_1 = \underline{c}'\underline{x}'_* + \alpha$$
$$y_2 = \underline{d}'\underline{x}'_* + \beta$$

The existence of a point $\underline{x}'_*$ follows from the linear independence

assumption. A point $\underline{x}'_{\#}$ can be obtained by arbitrarily setting the levels of (n-2) of the variables and solving for the remaining two. The map T is therefore onto and it follows that kern(T) $\neq \Phi$.

Thus far, based on the assumptions made about (P'_C), kern(T) $= \Phi$ and $\overline{X}' \neq \Phi$. With the additional assumption that $(\underline{d}'\underline{x}' + \beta) > 0$, it also follows that kern(T) $\cap$ $\overline{X}' = \Phi$.

Lemma 2

When $(\underline{d}'\underline{x}' + \beta) > 0 \quad \forall \underline{x}' \in \overline{X}'$, the set kern(T) together with any $\underline{x}'_a \in \overline{X}'$ define a hyperplane T^o_a in E^n.

Proof:

A well known result from linear algebra states that if $T: E^n \to E^m$, dim kern(T) + dim range (T) = n. In particular when m=2, kern(T) is an (n-2) dimensional subspace of E^n. With the assumptions made thus far, kern(T) $\cap$ $\overline{X}' = \Phi$. Hence any point $\underline{x}'_a \in \overline{X}'$ is linearly independent of any point belonging to kern(T) and any basis for kern(T) together with $\underline{x}'_a$ span an (n-1) dimensional subspace of E^n. By definition, this is a hyperplane in E^n. Recalling the definition of T^o_a given in (2.3), it is clear that in addition to the point $\underline{x}'_a$, every point in the set kern(T) lies on the hyperplane T^o_a. Hence kern(T) together with $\underline{x}'_a \in \overline{X}'$ define T^o_a. This proven the Lemma.

Lemma 2 provides the tool for showing that a change in the hyperbolic objective function value can be achieved by a rotation of T^o_1 about kern(T). To see that this is indeed the case, recall that given a

sequence $\{f(\underline{x}'_1),\ldots,f(\underline{x}'_n)\}$ with $\underline{x}'_i \in \overline{\underline{X}}'$ $\forall i$ and $f(\underline{x}'_i) < f(\underline{x}'_{i+1})$ $(i=1,\ldots,n-1)$ there exist a unique corresponding sequence $\{T^o_1,\ldots,T^o_n\}$. Appealing to Lemma 2, note that $\underline{x}'_i$ together with kern(T) define T^o_i and $\underline{x}'_{i+1}$ together with kern(T) define T^o_{i+1}. Furthermore, by definition of the hyperplane T^o_i

$$T^o_i \cap T^o_{i+1} = \text{kern}(T)$$

Since the functional values $f(\underline{x}'_i)$ and $f(\underline{x}'_{i+1})$ are achieved with points that lie on intersecting hyperplanes, and since it was originally assumed that $\underline{x}'_i \in \overline{\underline{X}}'$ $\forall i$ and $f(\underline{x}'_i) < f(\underline{x}'_{i+1})$, the desired conclusion follows.

To clarify the exposition in this section, consider the following problem:

$$\max \{f(x'_1,x'_2) = (c'_1x'_1 + c'_2x'_2 + \alpha)/(d'_1x'_1 + d'_2x'_2 + \beta)\}$$

$$\text{s.t.} \quad a'_{11}x'_1 + a'_{12}x'_2 \leq b_1$$

$$a'_{21}x'_2 + a'_{22}x'_2 \leq b_2$$

$$x'_1, x'_2 \geq 0$$

To avoid problems which need special attention, it is assumed that the feasible region is bounded, that numerator and denominator in the objective function are linearly independent and that $(d'_1x'_1 + d'_2x'_2 + \beta) > 0$ for all feasible points.

Based on the geometrical aspects of this problem, elements of the set kern(T) are also solutions of the system

$$c_1'x_1' + c_2'x_2' + \alpha = 0$$

$$d_2'x_2' + d_2'x_2' + \beta = 0 \qquad (2.5)$$

From the linear independent assumption, the solution to (2.5) is unique and therefore kern(T) is a singleton set in E^2. The assumption that $(d_1'x_1' + d_2'x_2' + \beta) > 0$ for all feasible points implies that kern(T) and the feasible region are mutually exclusive sets. Appealing to Lemma 2, kern(T) together with any feasible point $\underline{x}_a'$ define a hyperplane T_a^o. This is illustrated in Figure 1. The optimal objective function value for this sample problem is obtained by rotating T_a^o about kern(T). Examining Figure 1, it can be seen that the optimal hyperbolic solution for this problem must lie on at least one of the extreme points of the feasible set. (See Theorem 2).

Before proceeding to the integer solution of the hyperbolic programming problem, it will be useful to examine the Isbell-Marlow algorithm within the framework developed here. First a point $\underline{x}_a'$ such that $(\underline{d}'\underline{x}_a' + \beta) \neq 0$ is chosen and a feasible point $\underline{x}_1'$ which maximizes

$$(\underline{d}'\underline{x}_a' + \beta)(\underline{c}'\underline{x}' + \alpha) - (\underline{c}'\underline{x}_a' + \alpha)(\underline{d}'\underline{x}' + \beta)$$

is obtained by solving a linear programming problem. If the hyperplane T_1^o is a supporting hyperplane to the feasible region, the point $\underline{x}_1'$ is optimal for (P_C'). If $\underline{x}_1'$ is not optimal, a point $\underline{x}_2'$ which maximizes

$$(\underline{d}'\underline{x}_1' + \beta)(\underline{c}'\underline{x}' + \alpha) - (\underline{c}'\underline{x}_1' + \alpha)(\underline{d}'\underline{x}' + \beta)$$

is obtained. If T_2^o is a supporting hyperplane to the feasible region,

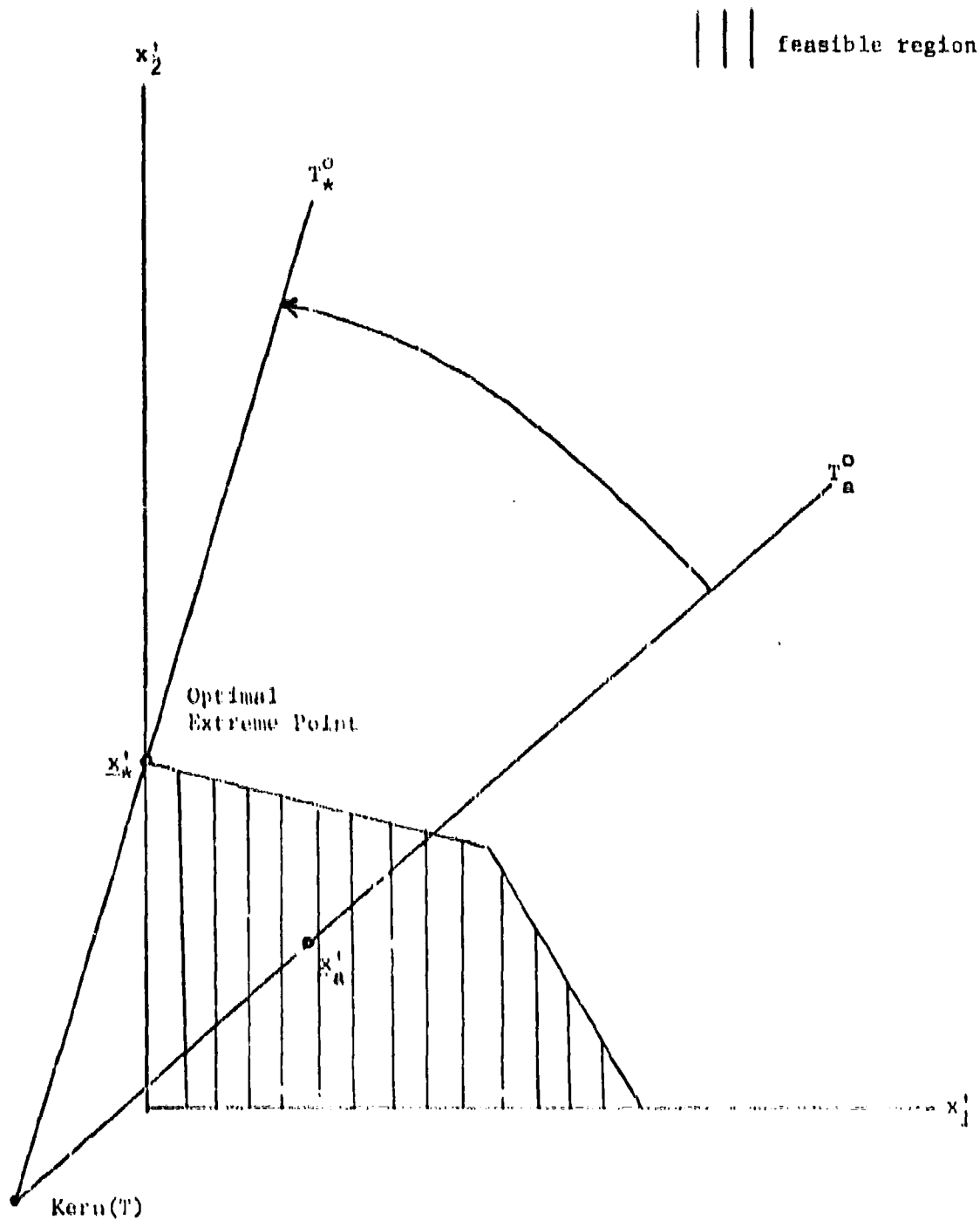
feasible region
x_2'
T^o_*
T^o_a
Optimal
Extreme Point
$\underline{x}'_*$
x'_a
x'_1
Kern(T)

Figure 1.

$\underline{x}^1_2$ is optimal. If it is not, the procedure is repeated. The workings of the Isbell-Marlow algorithm are demonstrated with an example in two variables in Figure 2.

Based on the geometrical analysis of (P^1_C), the most immediate way of obtaining an optimal solution is to rotate T^0_a about kern(T). Although the Isbell-Marlow does not implement a rotation of T^0_a, it accomplishes the same end. It sweeps across the feasible region in a direction towards the optimal solution. The procedure can be described as sequentially translating hyperplanes of changing slopes until the optimal extreme point for (P^1_C) is obtained.

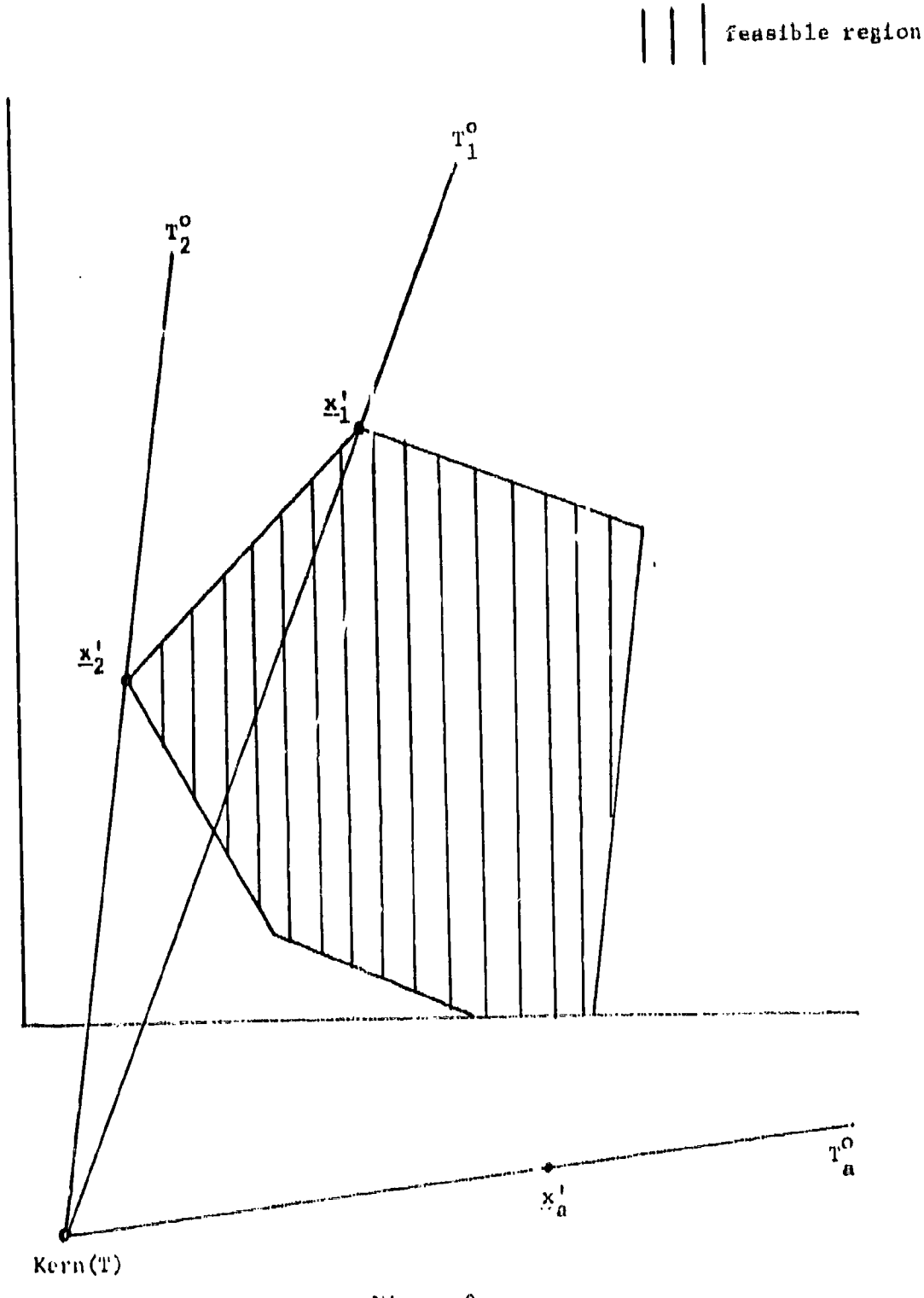
feasible region
T_1^o
T_2^o
$\underline{x}_1'$
$\underline{x}_2'$
T_a^o
$\underline{x}_a'$
Kern(T)

Figure 2.

CHAPTER 3

HYPERBOLIC INTEGER PROGRAMMING

3.1 A General Algorithm

Problem (P_I') with $\overline{X}'$ bounded implies that the number of feasible integer points is finite. Thus it is possible to view (P_I') as a special case of (P_F') which is discussed in Section 2.2.

Theorem 1 and Corollary 1.1 together with the observation that $\max_{\substack{\underline{x}' \in \overline{X}' \\ \underline{x}' \text{ integer}}} z(\underline{x}') \geq 0$ provides the basis for an algorithm to solve (P_I'). Such an algorithm is given below.

Algorithm HIP(1)

1. Find any feasible integer point belonging to $\overline{X}'$. Call the point $\underline{x}_0'$ and set $j=1$.
2. Define the following linear integer program:

   $\max \{z_j = (\underline{d}'\underline{x}_{j-1}' + \beta)(\underline{c}'\underline{x}' + \alpha) - (\underline{c}'\underline{x}_{j-1}' + \alpha)(\underline{d}'\underline{x}' + \beta)\}$

   s.t. $\underline{x}' \in \overline{X}'$

   $\underline{x}'$ integer
3. Solve the problem defined in step 2 by any convenient method. Call the solution $\underline{x}_j'$.
4. If $\max z_j = 0$, stop; otherwise let $j = j+1$ and return to step 2.

The convergence of HIP(1) depends solely on having a finite number of feasible integer points as possible optimal solutions for

(P_I'). For convenience, it is assumed that the number of feasible integer points is k. By construction, the algorithm generates a sequence of points $\{\underline{x}_i'\}$ such that $f(\underline{x}_i') \leq f(\underline{x}_{i+1}')$ $\forall i$. If there exist some finite index j such that $f(\underline{x}_j') = f(\underline{x}_{j+1}')$, it follows that $z_{j+1}(\underline{x}_{j+1}') = 0$. Appealing to Corollary 1.1, $\underline{x}_j'$ and $\underline{x}_{j+1}'$ are optimal hyperbolic solutions. In implementing the algorithm, the worst that can happen is that HIP(1) will generate all k feasible integer points such that $f(\underline{x}_1') < f(\underline{x}_2') < \ldots < f(\underline{x}_k')$. By the nature of HIP(1), it must be that at the next iteration $f(\underline{x}_k') = f(\underline{x}_{k+1}')$. Once again, appealing to Corollary 1.1, it follows that $\underline{x}_k'$ and $\underline{x}_{k+1}'$ are optimal hyperbolic solutions. The boundedness of $\overline{X}'$ insures that (P_I') has a finite number of feasible integer points. Thus if HIP(1) is used to solve (P_I'), the optimal solution will be obtained in a finite number of steps.

3.2 Discussion

The choice of the point $\underline{x}_0'$ in HIP(1) merits further discussion. To be sure that an objective function is generated at step 2 when j=1, it is only necessary to impose the restriction that $(\underline{c}'\underline{x}_0' + \alpha)$ and $(\underline{d}'\underline{x}_0' + \beta)$ are not zero simultaneously. In addition, the point $\underline{x}_0'$ need not be feasible and it need not be integer. If $\underline{x}_0'$ is not feasible, $f(\underline{x}_0')$ is not necessarily a lower bound for the maximum of the hyperbolic objective function. It should be clear however that, $\underline{x}_1'$, the solution to the first linear integer program in HIP(1) is feasible for (P_I') and furthermore $\max_{\underline{x}' \in \overline{X}'} f(\underline{x}') \geq f(\underline{x}_1')$. The worst that

can happen if $\underline{x}'_0$ is not feasible is that convergence of HIP(1) is slower.

Choosing $\underline{x}'_0$ with integer components is not necessary but is desirable if an all integer cutting plane method is used. If the constant terms α and β are not both equal to zero, it is convenient to choose $\underline{x}'_0$ as the null vector. If $\alpha = \beta = 0$, a first feasible integer point can be found in one of several ways. For example, a solution to the problem

$$\max \{z = \underline{1}'\underline{x}'\}$$
$$\text{s.t. } \underline{x}' \in \overline{X}'$$
$$\underline{x}' \text{ integer}$$

can always be used as $\underline{x}'_0$. If the hyperbolic integer program is a maximization problem, it is often reasonable to expect the solution to the problem

$$\max \{z = \underline{c}'\underline{x}'\}$$
$$\text{s.t. } \underline{x}' \in \overline{X}'$$
$$\underline{x}' \text{ integer}$$

to yield a near optimal point for (P'_I). Solving this problem is clearly another means for choosing $\underline{x}'_0$ in HIP(1).

One of the interesting aspects of the algorithm HIP(1) is that its convergence depends solely on having a finite number of feasible points. Special cases of (P'_I) fall into this category and have been considered by other authors. For example, when the components of the solution vector $\underline{x}'$ are restricted to be (0-1) variables and $\overline{X}'$ is defined by pseudo-boolean equations, HIP(1) reduces to an algorithm

similar to the one proposed by Robillard and Florian [20]. When the feasible region is defined by a set of linear inequalities and there is no integer restriction on the solution vector, HIP(1) reduces to an algorithm similar to the one proposed by Isbell and Marlow [16]. Thus the same algorithm which is used to solve an integer problem can also be used to solve a continuous problem. This is best explained by the fact that the solution to (P'_C) lies on at least one of the extreme points of the feasible region. Finding the optimal solution to (P'_C) can therefore be limited to a search of the extreme points of $\overline{X}'$. Under the assumptions made here, the number of extreme points is finite and hence in this sense (P'_C) is a special case of (P'_E). Therefore, HIP(1) can be used to solve (P'_C) and finite convergence is insured.

With regard to integer programming, HIP(1) is general enough to solve the problem

$$\begin{aligned} &\max \{f(\underline{x}') = (\underline{c}'\underline{x}' + \alpha)/(\underline{d}'\underline{x}' + \beta)\} \\ &\text{s.t.} \quad \underline{x}' \in \overline{X}' \\ &\qquad \underline{x}' \text{ integer} \end{aligned} \tag{P'_G}$$

where $\overline{X}'$ is an arbitrary bounded convex set with integer points in its interior. The ease with which (P'_G) can be solved depends on how easily each one of the subproblems can be solved. In the case when the feasible region is defined by parabolic constraints, the method of Witzgall [21] can be used to solve the subproblems. If $\overline{X}'$ is an arbitrary convex set with integer points in its interior, the cutting plane method of Kelley [22] can be used to solve the integer subproblems.

The algorithm HIP(1) is clearly much more general than it first appears. It will be shown later that HIP(1) can also be used to solve a hyperbolic group problem.

3.3 Alternate Optimal Solutions

Having obtained an optimal solution, say $\underline{x}'_*$, for (P'_I), the set of alternate optimal solutions coincides with all the feasible integer points which lie on the following hyperplane:

$$\{\underline{x}' \mid (\underline{d}'\underline{x}'_* + \beta)(\underline{c}'\underline{x}' + \alpha) - (\underline{c}'\underline{x}'_* + \alpha)(\underline{d}'\underline{x}' + \beta) = 0\}$$

For the special case when in (P'_I) $\alpha = \beta = 0$, the next result can be useful for generating alternate optimal solutions.

Lemma 3

If in (P'_I) $\alpha = \beta = 0$ and if $\underline{x}'_*$ is an optimal solution then if k is a positive rational number such that $k\underline{x}'_*$ is integer and $k\underline{x}'_* \in \overline{X}'$ then $k\underline{x}'_*$ is an alternate optimal solution for (P'_I).

Proof:

When $\alpha = \beta = 0$, $f(\underline{x}') = \underline{c}'\underline{x}'/\underline{d}'\underline{x}'$. It follows immediately that the optimal objective function value, $f(\underline{x}'_*) = \underline{c}'\underline{x}'_*/\underline{d}'\underline{x}'_*$ remains unchanged if numerator and denominator are respectively multiplied by a positive rational number k. Therefore if $k\underline{x}'_* \in \overline{X}'$ and if $k\underline{x}'_*$ is integer it is also optimal.

3.4 Reducing the Constraint Set

It was noted earlier that if the point $\underline{x}'_j$ is a feasible integer point for (P'_I) then $f(\underline{x}'_j)$ is a lower bound for the maximum of the objective function value in (P'_I). This implies that

$$(\underline{d}'\underline{x}'_j + \beta)(\underline{c}'\underline{x}' + \alpha) - (\underline{c}'\underline{x}'_j + \alpha)(\underline{d}'\underline{x}' + \beta) \geq 0 \qquad (3.1)$$

is a valid constraint for the (j+1)th subproblem in the general algorithm. A natural modification of HIP(1) is therefore to redefine the feasible region for each subproblem. Let

$$\overline{X}'_{j+1} = \overline{X}'_j \cap T_j \cup T_j^0 \qquad j=1,\ldots$$

where $\overline{X}'_1 = \overline{X}'$, T_j^0 is the hyperplane defined in (2.3), and

$$T_j^+ = \{\underline{x}' \mid (\underline{d}'\underline{x}'_j + \beta)(\underline{c}'\underline{x}' + \alpha) - (\underline{c}'\underline{x}'_j + \alpha)(\underline{d}'\underline{x}' + \beta) > 0\}$$

The set described by $\overline{X}'_{j+1}$ is nothing more than the original constraint set with the additional constraint (3.1).

Trauth and Woolsey [23] have reported that bounding constraints such as (3.1) have helped to speed convergence for certain linear integer programs which were not tractable previously. Whether or not bounding constraints such as (3.1) will help to speed the convergence of HIP(1) remains to be seen. It is of some interest to note that (3.1) is implicit in the original statement of HIP(1) and is therefore a redundant constraint.

It is now shown that for the modified version of HIP(1), the

optimality condition $z_j(\underline{x}'_j) = 0$ is equivalent to the condition $\overline{\underline{X}}'_{j+1} = \overline{\underline{X}}'_j$. By construction $\overline{\underline{X}}'_{j+1} \subseteq \overline{\underline{X}}'_j$. Furthermore if $\underline{x}'_j$ is an optimal solution and $z_j(\underline{x}'_j) = 0$, it follows that

$$T_j^o \cup T_j^+ \supseteq \overline{\underline{X}}'_j$$

or

$$T_j^o \cup T_j^+ \cap \overline{\underline{X}}'_j \supseteq \overline{\underline{X}}'_j$$

By definition $\overline{\underline{X}}'_{j+1} = T_j^o \cup T_j^+ \cap \overline{\underline{X}}'_j$ and it follows that

$$\overline{\underline{X}}'_{j+1} \supseteq \overline{\underline{X}}'_j$$

Therefore, if $\underline{x}'_j$ is optimal $\overline{\underline{X}}'_j = \overline{\underline{X}}'_{j+1}$.

3.5 Cutting Plane Algorithms

Cutting plane methods for solving the linear integer programming problem were suggested by Dantzig, Fulkerson and Johnson [24] and formalized by Gomory [25]. The principle behind these methods is to reduce the original feasible region without eliminating any feasible integer points and then to reoptimize the objective function on the reduced feasible set. If the reoptimization phase yields an all integer solution, it is also optimal. For the linear integer program, Gomory [25] accomplishes this by generating new constraints from existing constraints and then reoptimizes the objective function with the aid of the dual simplex algorithm. The finiteness of this algorithm is insured if the cutting planes are generated according to Gomory's procedure.

A desirable characteristic belonging to the Gomory cuts is that they can be generated systematically from an optimal noninteger tableau as well as from a nonoptimal dual feasible tableau.

In the next few sections, a class of algorithms which can be viewed as a synthesis of the Martos [2] algorithm, the Isbell-Marlow [16] algorithm, Gomory cutting planes and Lemke's dual simplex method of linear programming is developed. Alternately, this class of algorithms can be viewed as a special case of HIP(1).

3.6 Gomory Cuts

In the context of linear integer programming, a Gomory cut is an inequality derived from an existing constraint and must be satisfied by any integer solution. When added to the original set of constraints, this inequality reduces the original feasible region without excluding any feasible integer points.[1] The cutting planes can be derived in the following manner. Denote by $[x]$ the largest integer less than or equal to x. If x is any number and λ is any positive number, it follows that

$$x/\lambda = [x/\lambda] + r_{x/\lambda} \tag{3.2}$$

where $0 \leq r_{x/\lambda} < 1$. Equation (3.2) can be rewritten as

$$x = [x/\lambda]\lambda + r_x \tag{3.3}$$

---

[1] Gomory's fractional and all integer cuts reduce the feasible set $\bar{X}'$ by cutting off the current extreme point if it is not integer. Since the optimal solution to the continuous hyperbolic program occurs at an extreme point of $\bar{X}'$, the possibility of developing a cutting plane algorithm becomes evident.

where $0 \leq r_x < \lambda$. At iteration t, the ith basic variable can always be expressed in terms of the nonbasic variables. That is

$$x_i^t = a_{i0}^t + \sum_{j \in N} a_{ij}^t (-x_j^t) \tag{3.4}$$

where N is the index set for the current nonbasic variables. Using (3.3)

$$x_i^t = x_i^t \cdot 1 = x_i^t \{[1/\lambda]\lambda + r_i^t\}$$

$$a_{ij}^t = [a_{ij}^t/\lambda]\lambda + r_{ij}^t \tag{3.5}$$

Substituting (3.5) into (3.4)

$$x_i^t = \{[1/\lambda]\lambda + r_i^t\}\, x_i^t = [a_{i0}^t/\lambda] + r_{i0}^t \sum_{j \in N} \{[a_{ij}^t/\lambda]\lambda$$

$$+ r_{ij}^t\}\, (-x_j^t) \tag{3.6}$$

Collecting and rearranging terms, (3.6) becomes

$$x_i^t\, r_i^t + \sum_{j \in N} x_j^t r_{ij}^t = r_{i0}^t + \lambda\{[a_{i0}^t/\lambda]$$

$$+ \sum_{j \in N} [a_{ij}^t/\lambda](-x_j^t) + [1/\lambda](-x_i^t)\} \tag{3.7}$$

By definition, the left-hand side of (3.7) must be nonnegative. The term in brackets on the right-hand side is required to be an integer and furthermore it must be nonnegative. This can be seen by letting

$$s = [a_{i0}^t/\lambda] + \sum_{j \in N} [a_{ij}^t/\lambda](-x_j^t) + [1/\lambda](-x_i^t) \tag{3.8}$$

Suppose s is a negative integer. A choice of λ such that $\lambda > (r_{i0}^t/-s) > 0$ implies that $(r_{i0}^t + \lambda s) < 0$. This is a contradiction since it implies that the left-hand side in (3.7) is negative. The inequality $s \geq 0$ is therefore an appropriate cutting plane. For the special case when $\lambda=1$, (3.8) reduces to the Gomory fractional cut and when $\lambda > 1$ it reduces to the Gomory all integer cut.

3.7 Some Background

Before proceeding to a discussion of cutting plane algorithms for solving (P_I'), some results regarding the continuous hyperbolic programming problem are reviewed.

Theorem 2 (Martos [2], Dorn [17])

If in (P_C), the feasible region $\bar{X}$ is bounded, and if $(\underline{d}\,\underline{x} + \beta) > 0 \quad \forall \underline{x} \in \bar{X}$, the optimal solution to (P_C) occurs on at least one extreme point of $\bar{X}$.

The algorithm of Martos is based on Theorem 2. The Theorem suggests that the optimal solution can be obtained by performing an extreme point search similar to the simplex algorithm of linear programming.

Geometrically, the simplex algorithm of linear programming is initiated by examining any extreme point of the feasible set $\bar{X}$. If the extreme point is optimal, it is recognized as such and if it is not, rules are available for moving to an adjacent extreme point which also improves the objective function value. If no improvement is possible, the algorithm terminates. To see how these same ends are attained for

the hyperbolic program, let

$$z^1 = \underline{c}_B \underline{x}_B + \alpha$$

$$z^2 = \underline{d}_B \underline{x}_B + \beta$$

The subscript B denotes the current basis and $\underline{x}_B$ denotes the current feasible solution. Via a change of basis, the new values of z^1 and z^2, respectively, become

$$\bar{z}^1 = z^1 - \theta(z_j^1 - c_j)$$

$$\bar{z}^2 = z^2 - \theta(z_j^2 - d_j)$$

where θ is a positive constant.[1] The quantities $(z_j^1 - c_j)$ and $(z_j^2 - d_j)$ are the respective relative costs associated with numerator and denominator of the hyperbolic objective function. Clearly, for an improved objective function value, the following condition must hold:

$$z^1/z^2 < \bar{z}^1/\bar{z}^2 \tag{3.9}$$

To determine whether or not the condition given in (3.9) is satisfied, it is useful to define the following quantity:

$$t_j = z^1(z_j^2 - d_j) - z^2(z^1 - c_j) \tag{3.10}$$

The condition in (3.9) implies that an improvement in the hyperbolic

---

[1] The constant θ is computed exactly as in the simplex algorithm of linear programming. That is $\theta = \min_i \{x_{Bi}/y_{ik},\ y_{ik} > 0\}$.

objective function value can occur only if there exist some nonbasic column of the constraint matrix A for which $t_j > 0$. If in addition $\theta > 0$, a strict improvement occurs. Note that the condition $\theta > 0$ is always true if $\underline{x}_B > 0$ $\forall B$. Clearly, problems of degeneracy in hyperbolic programming are analogous to degeneracy problems in linear programming. This being the case, Martos [2] points out that hyperbolic cycling which might result from hyperbolic degeneracy can usually be resolved with the aid of Charne's perturbation method. To continue with the Martos algorithm, if $t_j \leq 0$ for all nonbasic columns, the current solution cannot be improved by moving to an adjacent extreme point.

Theorem 3 (Dorn [17])

A local maximum for a hyperbolic programming problem is also a global maximum.

Appealing to Theorem 3, if $t_j \leq 0$ $\forall j$, the solution is also optimal.

The algorithm of Martos is demonstrated with an example in Appendix A.

3.8 A Fractional Cutting Plane Algorithm

In this discussion, the fractional cutting plane algorithm for (P_I) is viewed as a special case of the algorithm MIP(1). The first step is to choose a feasible point, not necessarily an integer point, and construct a linear integer program whose solution is feasible for (P_I). In this algorithm, the feasible point is the optimal solution for (P_C) and it is obtained by using the column tableau for Martos' algorithm (see

Appendix A). If the solution displayed in the optimal tableau is all integer, it is also optimal for (P_I). If it is not, a Gomory fractional cutting plane can be generated from the optimal representation of the constraint set. Appending a legitimate cutting plane to the optimal hyperbolic tableau does not exclude any feasible integer points and thus a reoptimization of the enlarged tableau can lead to the optimal integer solution. Unfortunately, the Gomory fractional cutting plane destroys the primal tableau and there is no analog of the dual simplex method for the hyperbolic programming problem which can be used to reoptimize the tableau.

Primal feasibility can however be regained by solving

$$\max \{z = (\underline{d}\,\underline{x}_* + \beta)(\underline{c}\,\underline{x} + \alpha) - (\underline{c}\,\underline{x}_* + \alpha)(\underline{d}\,\underline{x} + \beta)\}$$

$$\text{s.t.} \quad A\underline{x} = \underline{b}$$

$$\underline{x} \geq 0 \text{ and integer} \qquad \text{(P1)}$$

where $\underline{x}_*$ is the extreme point of the feasible set which maximizes the continuous hyperbolic program. (P1) is solved using Gomory's fractional algorithm. This requires first obtaining the optimal tableau for (P1) without the integer restriction. This tableau is available without any additional work if the algorithm of Martos with the column tableau is used to solve (P_C). To see this, let the optimal functional value of the numerator be

$$z^1 = (\underline{c}\,\underline{x}_* + \alpha)$$

and similarly let the optimal functional value of the denominator be

$$z^2 = (\underline{d}\,\underline{x}_* + \beta)$$

Substituting z^1 and z^2 in (P1) and partitioning the vector $\underline{x}$ into a basic and a nonbasic component, the objective function in (P1) becomes

$$z = z^2(\underline{c}_B\underline{x}_B + \underline{c}_N\underline{x}_N + \alpha) - z^1(\underline{d}_B\underline{x}_B + \underline{d}_N\underline{x}_N + \beta)$$

The basic vector $\underline{x}_B$ is eliminated using the relation

$$\underline{x}_B = B^{-1}(\underline{b} - N\underline{x}_N)$$

This yields

$$z = z^2[(\underline{c}_B B^{-1}\underline{b} + \alpha) + (\underline{c}_N - \underline{c}_B B^{-1}N)\underline{x}_N] -$$

$$z^1[(\underline{d}_B B^{-1}\underline{b} + \beta) + (\underline{d}_N - \underline{d}_B B^{-1}N)\underline{x}_N] \qquad (3.11)$$

When B is an optimal basis for (P_C), (3.11) reduces to

$$z = \{z^1(\underline{d}_B B^{-1} - \underline{d}_N) - z^2(\underline{c}_B B^{-1}N - \underline{c}_N)\}\,\underline{x}_N \qquad (3.12)$$

The components of the vector in brackets are easily recognized as the quantities t_j defined in (3.10). Moreover they are the optimal t_j's and hence $t_j \leq 0 \quad \forall j \in N$. This implies that the optimal relative costs for the continuous version of (P1) are exactly equal to the optimal t_j's. An optimal basis for (P_C) is therefore also optimal for the continuous version of (P1). To obtain the complete optimal tableau for the latter, it is only necessary to append a row corresponding to (3.12) to the optimal hyperbolic tableau. A cutting plane is also

appended to the tableau and the linear objective function is reoptimized via the dual simplex algorithm. This procedure is repeated until an all integer solution for (P1) is obtained. It is interesting to note that the noninteger optimal objective function value for (P1) is zero. This implies that the first Gomory cut is never generated from the objective function row.

In addition to the optimality condition given in step 4 of HIP(1), an optimal solution for (P_I) can also be recognized with the aid of the following sufficient condition:

Lemma 4

If the quantities t_j are computed from the optimal tableau for (P1) and $t_j \leq 0$ $\forall j$, the current solution is also optimal for the hyperbolic integer program.

Proof:

After (P1) is solved, the reduced feasible region can be expressed as the intersection of a finite number of hyperplanes. In particular, the feasible region is $\overline{X}_1 \cap S_1$ where S_1 is the region defined by the Gomory cuts used to solve (P1) and $\overline{X}_1 = \overline{X}$. From the Gomory cut properties, it follows that $\overline{X}_1$ and $\overline{X}_1 \cap S_1$ contain exactly the same integer points. Thus appealing to the Martos algorithm, if $t_j \leq 0$ $\forall j$, the solution to (P1) is also optimal for (P_I).

If $t_j \leq 0$ $\forall j$, T_1^0 is a supporting hyperplane to $\overline{X}_1 \cap S_1$ at x_1 and x_1 is also optimal for (P_I). If there exist $t_j > 0$ for some

$j \in N$, it is still possible for the current solution to be optimal for (P_I). To see if this is indeed the case, the following problem is formulated.

$$\max \{z = (\underline{d}\,\underline{x}_1 + \beta)(\underline{c}\,\underline{x} + \alpha) - (\underline{c}\,\underline{x}_1 + \alpha)(\underline{d}\,\underline{x} + \beta)\}$$

$$\text{s.t.} \quad \underline{x} \in \overline{X}_1 \cap S_1$$

$$\underline{x} \text{ integer} \qquad \text{(P2)}$$

The optimal solution to (P2) is again obtained by using Gomory's fractional algorithm. The noninteger solution which is obtained first is called $\underline{x}_1^*$. Note that the hyperplane $z(\underline{x}_1^*) = (\underline{d}\,\underline{x}_1 + \beta)(\underline{c}\,\underline{x}_1^* + \alpha) - (\underline{c}\,\underline{x}_1 + \alpha)(\underline{d}\,\underline{x}_1^* + \beta)$ is parallel to T_1^o. Thus if a cutting plane algorithm is used to solve (P2), a new integer extreme point on the integer hull contained in the original feasible region is obtained or an integer point lying on T_1^o is obtained. If the latter occurs, $z(\underline{x}_2) = 0$ and $\underline{x}_2$ which is the optimal solution to (P2) is also optimal for (P_I). If the solution to (P2) does not lie on T_1^o and there exist some $t_j > 0$, a new linear integer program is defined and the procedure is repeated. A statement of the fractional cutting plane algorithm is now given below. The following definitions will be useful.

$\overline{X}_1 = \overline{X} = \{\underline{x} | A\underline{x} = \underline{b},\ \underline{x} \geq 0\}$

$\overline{X}_j \sim$ feasible region for jth linear integer program

$S_j \sim$ region defined by Gomory cuts used to solve jth linear integer program.

Algorithm HIP(2)

1. Disregard the integer restriction and solve (P_I) using the column tableau version of the Martos algorithm (see Appendix A). Call the solution $\underline{x}_*$. If the solution is all integer stop; otherwise let $j = 1$ and $\underline{x}_0 = \underline{x}_*$.
2. Define the following linear integer program:

   $\max \{z_j = (\underline{d}\ \underline{x}_{j-1} + \beta)(\underline{c}\ \underline{x} + \alpha) - (\underline{c}\ \underline{x}_{j-1} + \alpha)(\underline{d}\ \underline{x} + \beta)\}$

   s.t. $\underline{x} \in \overline{\underline{X}}_j$

   $\underline{x}$ integer

   where $\overline{\underline{X}}_1 = \overline{\underline{X}}$

   $\overline{\underline{X}}_j = \overline{\underline{X}}_{j-1} \cap S_{j-1}$ for $j=2,3,\ldots$
3. Solve the problem in step 2 using Gomory's fractional algorithm. Call the solution $\underline{x}_j$.
4. If $\max z_j = 0$ or if $t_j \leq 0\ \forall j \in N$ stop; otherwise let $j = j+1$ and return to step 2.

Although HIP(2) is clearly a special case of HIP(1), it is also in the same spirit as Gomory's fractional algorithm for the linear integer program. That is, the optimal extreme point for (P_C) is obtained first. If the solution is integer, the procedure terminates. If it is not, a cutting plane which cuts off the current extreme point and which does not delete any feasible integer points is appended to the original feasible region. The objective function is then reoptimized on the reduced feasible set. New cutting planes are generated until the reoptimization phase yields an extreme point of the integer hull

contained in $\bar{X}$. What makes (P_I) more difficult to solve than the linear integer program is that more than one extreme point of the integer hull may have to be generated before the optimality condition is satisfied. Clearly since $\bar{X}$ is assumed bounded, the integer hull has a finite number of extreme points. At most HIP(2) will generate all of these and since $f(\underline{x}_{j+1}) \geq f(\underline{x}_j)$ $\forall j$, HIP(2) will converge in a finite number of steps.

Although conceptually the same, HIP(1) and HIP(2) do have some differences. In HIP(2), the feasible region is reduced at each iteration whereas in HIP(1) the feasible region remains the same at each iteration. Because of the nature of cutting plane algorithms, HIP(2) only searches points which lie within the boundary of the original feasible region and which are outside or on the boundary of the integer hull contained in the original feasible region. This is not necessarily the case with HIP(1) if a branch and bound algorithm is used to solve each of the linear integer programs. A special property of HIP(2) is that the optimal tableau for the jth linear integer problem with some slight modification becomes the initial tableau for the (j+1)th problem. This simply means that when solving the (j+1)th problem, it is not necessary to start from the beginning.

The cutting plane algorithm described here is implemented with the aid of the tableau format shown in Table 1. The entries in the first row are the t_j's ($j \in N$) which determine the optimality of the hyperbolic program. They are computed after each integer solution is obtained according to

TABLE 1

		t_1	t_2	...	t_s	. . . .	t_n
	1	$-x_1$	$-x_2$	...	$-x_s$	. . . .	$-x_n$
N	n_{00}	n_{01}	n_{02}	...	n_{0s}	. . . .	n_{0n}
D	d_{00}	d_{01}	d_{02}	...	d_{0s}	. . . .	d_{0n}
z	a_{00}	a_{01}	a_{02}	...	a_{0s}	. . . .	a_{0n}
x_1	a_{10}	a_{11}	a_{12}	...	a_{1s}	. . . .	a_{1n}
.	.	.	.		.		.
.	.	.	.		.		.
.	.	.	.		.		.
.	.	.	.		.		.
x_n	a_{n0}	a_{n1}	a_{n2}	...	a_{ns}	. . . .	a_{nn}
x_{n+1}	$a_{n+1,0}$	$a_{n+1,1}$	$a_{n+1,2}$	...	$a_{n+1,s}$	. . . .	$a_{n+1,n}$
.	.	.	.		.		.
.	.	.	.		.		.
.	.	.	.		.		.
.	.	.	.		.		.
x_{n+m}	$a_{n+m,0}$	$a_{n+m,1}$	$a_{n+m,2}$	...	$a_{n+m,s}$	. . . .	$a_{n+m,n}$
s_i	$-f_{i0}$	$-f_{i1}$	$-f_{i2}$	...	$-f_{is}$	. . . .	$-f_{in}$

Appealing to Lemma 4, the current solution is optimal.

Lemma 5 implies that if $a_{0j} \geq 0$ $\forall j \in N$ then increasing any one of the original variables to a positive level will decrease the objective function value. If there exist $j \in N$ such that $a_{0j} < 0$, the initial tableau is primal feasible with respect to the relative costs displayed in the z-row. It is not however dual feasible. As in HIP(2), the first step is to find a feasible integer point for (P_I). In addition, an all integer tableau is to be maintained at each iteration. Each of these objectives is achieved if the linear objective displayed in the z-row is maximized using either a primal cutting plane algorithm or Gomory's all integer algorithm. If a primal cutting plane algorithm is used, the first pivot can be made without altering the first tableau. If the Gomory all integer algorithm is used, the current tableau must first be rendered dual feasible with respect to the relative costs of the linear objective function. This is accomplished by appending the constraint

$$x_{n+m+1} = M - \sum_{j \in N} x_j \tag{3.13}$$

to the current tableau and choosing it as the pivot row. The pivot column is chosen as the lexicographically smallest column with respect to the current linear program. The constant M in (3.13) is an arbitrarily large integer such that $x_{n+m+1} \geq 0$. The constraint given by (3.13) simply expresses the fact that the original constraint set is bounded. A pivot at this point will result in an all integer dual feasible tableau. A statement of the all integer algorithm is given below.

$$t_j = n_{00}d_{0j} - n_{0j}d_{00} \qquad \forall j \in N$$

This is the same index t_j defined in (3.10). The entries in the second row identify the nonbasic variables at the current iteration. The N and D rows represent the numerator and denominator, respectively. The entry n_{00} is the current numerator value and d_{00} is the current denominator value. The z row denotes the current linear objective whose integer solution is being sought. Rows x_1 through x_n are the current nonbasic variables and rows x_{n+1} through x_{n+m} are the current basic variables. The last row is for the Gomory cut. The use of HIP(2) is demonstrated with an example in Appendix B.

3.9 An All Integer Cutting Plane Algorithm

In this section, a cutting plane algorithm based on the Gomory all integer cut is developed. The algorithm is a special case of HIP(1) and is started by displaying the problem in the column tableau format (see Table 1). The entries in the z-row are

$$a_{00} = 0$$

$$a_{0j} = -n_{00}d_{0j} + n_{0j}d_{00} \qquad \forall j \in N$$

Lemma 5

The optimal hyperbolic function value is (α/β) and the optimal integer solution is $\underline{x} = 0$ if $a_{0j} \geq 0 \quad \forall j \in N$.

Proof:

By definition $t_j = -a_{0j} \; \forall j$. Thus if $a_{0j} \geq 0 \; \forall j$, $t_j \leq 0 \; \forall j$.

Algorithm HIP(3)

1. Set up the column tableau for the hyperbolic program as in Table 1. The components of the vector $\underline{x}_0$ are $x_{i0} = 0$ $(i=1,\ldots,n)$ and $x_{n+j,0} = b_j$ $(j=1,\ldots,m)$. Set $j=1$.
2. Generate the following linear integer programming problem:

$$\max \{z_j = (\underline{d}\,\underline{x}_{j-1} + \beta)(\underline{c}\,\underline{x} + \alpha) - (\underline{c}\,\underline{x}_{j-1} + \alpha)(\underline{d}\,\underline{x} + \beta)\}$$

$$\text{s.t.} \quad \underline{x} \in \overline{\underline{X}}_j$$

$$\underline{x} \text{ integer}$$

$$\text{where} \quad \overline{\underline{X}}_1 = \overline{\underline{X}}$$

$$\overline{\underline{X}}_j = \overline{\underline{X}}_{j-1} \cap S_{j-1}.$$

3. Option A: Solve the problem defined in step 2 using a primal cutting plane algorithm.

   Option B: Append the constraint given by (3.13) and pivot to render tableau dual feasible with respect to the relative costs in the z-row. Use the Gomory all integer algorithm to maximize the linear objective function.
4. Compute t_j $\forall j \in N$. If $t_j \leq 0$ $\forall j \in N$ or if $z_j(\underline{x}_j) = 0$ stop; otherwise let $j = j+1$ and return to step 2.

Since HIP(3) is a special case of HIP(1), the finiteness proofs are the same. It should be pointed out that when implementing HIP(3) with option B, the constant M in (3.13) must be chosen with some care. It is important to realize that a constraint of the form (3.13) must be appended every time a linear integer program is generated. Since the set of nonbasic variables changes for every linear integer program,

a constant M which is large enough for the first problem may not be large enough for the second problem. This simply means that the constant may have to be changed at each iteration. Certainly, if it is chosen large enough at the outset, it need never be changed.

The use of HIP(3) is demonstrated with the use of an example in Appendix C.

3.10 A Variant of the Fractional Cutting Plane Algorithm

This algorithm is initiated in precisely the same way as HIP(2). The counter index is set equal to one and the first three steps are repeated. If $t_j \leq 0\ \forall j$, the current solution is also optimal for the hyperbolic program. If there exist $t_j > 0$ for some $j \in N$, this algorithm makes use of the fact that $f(\underline{x}_j)$ is a lower bound for the maximum of the hyperbolic objective function. In other words, the integer solution must satisfy the following constraint:

$$(\underline{d}\ \underline{x}_j + \beta)(\underline{c}\ \underline{x} + \alpha) - (\underline{c}\ \underline{x}_j + \alpha)(\underline{d}\ \underline{x} + \beta) \geq 0 \qquad (3.14)$$

The strategy in this algorithm is to append (3.14) to the current tableau and use the Martos algorithm to reoptimize it. This step yields the optimal hyperbolic extreme point for the current feasible region. If the optimal solution has all integer components, the algorithm terminates. If it does not, let $\underline{x}_j^*$ denote the optimal hyperbolic noninteger extreme point and generate the following linear integer program:

$$\max \{z_{j+1} = (\underline{d}\,\underline{x}_j^* + \beta)(\underline{c}\,\underline{x} + \alpha) - (\underline{c}\,\underline{x}_j^* + \alpha)(\underline{d}\,\underline{x} + \beta)\}$$

$$\text{s.t.} \quad \underline{x} \in \overline{X}_{j+1}$$

$$\underline{x} \text{ integer}$$

where $\overline{X}_{j+1} = \overline{X}_j \quad S_j \quad T_j^+ \quad T_j^\circ \quad (j=1,2,\ldots)$. The solution to this problem is either optimal for the hyperbolic integer program or it is an improved lower bound. The algorithm is described below.

Algorithm HIP(4)

1. Solve (P_I) disregarding the integer restriction. If the solution is all integer, stop; otherwise set $\underline{x}_0^* = \underline{x}^*$ and $j=1$.
2. Define the following linear integer program:

   $$\max \{z_j = (\underline{d}\,\underline{x}_{j-1}^* + \beta)(\underline{c}\,\underline{x} + \alpha) - (\underline{c}\,\underline{x}_{j-1}^* + \alpha)(\underline{d}\,\underline{x} + \beta)\}$$

   $$\text{s.t.} \quad \underline{x} \in \overline{X}_j, \ \underline{x} \text{ integer}$$

   where $\overline{X}_1 = \overline{X}$

   $$\overline{X}_j = \overline{X}_{j-1} \cap S_{j-1} \cap T_{j-1}^+ \cup T_{j-1}^\circ \qquad j=2,3,\ldots$$

3. Solve the problem defined in step 2 using Gomory's fractional algorithm. Call the solution $\underline{x}_j$.
4. If $t_j \leq 0 \ \forall j \in N$ or if $f(\underline{x}_{j-1}) = f(\underline{x}_j)$ stop; otherwise go to step 5.
5. Use the Martos algorithm to reoptimize the current tableau. If the resulting solution is integer, stop; otherwise let $\underline{x}_j^*$ be the optimal extreme point and let $j = j+1$ and return to step 2.

The feasible region $\overline{X}_{j+1}$ is defined by appending the constraint

given in (3.14) to the optimal tableau for the jth linear problem. This constraint is only kept on the tableau for the (j+1)th problem and is discarded thereafter. The reason for this is that when

$$(\underline{d}\ \underline{x}_{j+1} + \beta)(\underline{c}\ \underline{x} + \alpha) - (\underline{c}\ \underline{x}_{j+1} + \alpha)(\underline{d}\ \underline{x} + \beta) \geq 0$$

is appended to $\overline{X}_{j+1} \cap S_{j+1}$, the result is that $\overline{X}_{j+2} \subset \overline{X}_{j+1}$.

The algorithm HIP(4) can be helpful when it is desirable to terminate at some near optimal integer point. Note that the quantities $f(\underline{x}_j)$ and $f(\underline{x}_j^*)$ are respectively lower and upper bounds for the maximum of the hyperbolic function. Thus the quantity $\delta = f(\underline{x}_j^*) - f(\underline{x}_j) \geq 0$ is a measure of how close $\underline{x}_j$ is to the optimal integer solution. When $\underline{x}_j$ is optimal $\delta = 0$. A stopping rule is therefore to compute δ and stop when $\delta \leq \delta^*$ where δ^* is user specified.

HIP(4) converges in a finite number of steps because $\overline{X}_{j+1} \subset \overline{X}_j$ $\forall j$ and the number of feasible integer points contained in $\overline{X}_{j+1}$ is at least one less than the number of feasible integer points contained in $\overline{X}_j$. Hence in the worst case there exist some finite iteration K such that $\overline{X}_K$ contains exactly one integer point. Since by virtue of HIP(4) $f(\underline{x}_{j+1}) \geq f(\underline{x}_j)$ it follows that when there is only one integer point left in the feasible region $f(\underline{x}_{j+1}) = f(\underline{x}_j)$. Optimality is therefore achieved in a finite number of steps.

3.11 Group Theoretic Approach to Hyperbolic Integer Programming

In this section, it will be shown that the group theoretic approach used to solve the linear integer program can also be used to

solve (P_I). It will be necessary however to make the additional assumption that when disregarding the integer restriction, the optimal value of the numerator is greater than or equal to zero. This is not restrictive in an economic context since the numerator is usually a cost function. Before proceeding to the hyperbolic group problem, a brief review of the group theoretic aspects of linear integer programming is given.

The linear integer programming problem is of the form

$$\begin{aligned} &\max \{z = \underline{c}\ \underline{x}\} \\ &\text{s.t.} \quad \underline{x} \in \overline{\underline{X}} \\ &\qquad \underline{x} \text{ integer} \end{aligned} \qquad (LP_I)$$

where $\overline{\underline{X}}$ is defined exactly as in (P_I). The group optimization problem associated with (LP_I) is generated by first solving (LP_I) without the integer restriction. This makes it possible to partition the constraint matrix A into a set of optimal basic columns B and a set of optimal nonbasic columns N. This partitioning is always possible if the simplex algorithm is used to solve (LP_C). Partitioning the solution vector accordingly, the constraint equation can be written as

$$B\underline{x}_B + N\underline{x}_N = \underline{b}$$

Expressing the basic vector $\underline{x}_B$ in terms of the nonbasic vector $\underline{x}_N$, and multiplying the objective function by (-1), (LP_I) can be reformulated as follows:

$$\min \{z = -\underline{c}_B B^{-1}\underline{b} - (\underline{c}_N - \underline{c}_B B^{-1}N)\underline{x}_N\}$$

$$\text{s.t.} \quad \underline{x}_B = B^{-1}(\underline{b} - N\underline{x}_N)$$

$$\underline{x}_B, \underline{x}_N \geq 0 \quad \text{and integer} \qquad (LP1_I)$$

The term $(-\underline{c}_B B^{-1}\underline{b})$ which appears in the objective function is constant and therefore the problem

$$\min \{z^* = -(\underline{c}_N - \underline{c}_B B^{-1}N)\underline{x}_N\}$$

$$\text{s.t.} \quad \underline{x}_N \geq B^{-1}(\underline{b} - N\underline{x}_N)$$

$$\underline{x}_B, \underline{x}_N \geq 0 \quad \text{and integer} \qquad (LP2_I)$$

has the same solution as $(LP1_I)$. Furthermore all the costs in the objective function are nonnegative. This is a consequence of the optimality criterion used in the simplex algorithm. If the nonnegativity restriction on $\underline{x}_B$ is dropped, problem $(LP2_I)$ becomes

$$\min \{z^* = -(\underline{c}_N - \underline{c}_B B^{-1}N)\underline{x}_N\}$$

$$\text{s.t.} \quad B^{-1}N\underline{x}_N \equiv B^{-1}\underline{b} \qquad (\text{mod } 1)$$

$$\underline{x}_N \geq 0 \quad \text{and integer} \qquad (LP3_I)$$

It is important to note that if $\underline{x}_N^*$ minimizes z^*, $(B^{-1}(\underline{b} - N\underline{x}_N^*), \underline{x}_N^*)$ need not be feasible for $(LP2_I)$. Although the vector $\underline{x}_B^*$ will be all integer if $\underline{x}_N^*$ is all integer, it need not have all positive components. Gomory [26] shows that if $(\underline{b} - N\underline{x}_N)$ lies in the cone generated by the columns of B, then $\underline{x}_B = B^{-1}(\underline{b} - N\underline{x}_N) \geq 0$. Hence there exists a class of problems such that the solution to $(LP3_I)$ will always generate a feasible solution for $(LP2_I)$.

The algorithms which attempt to solve (LP_I) via ($LP3_I$) neglect the nonnegativity restriction on $\underline{x}_B$. If the solution to ($LP3_I$) generates a feasible solution for (LP_I) it can be shown that the solution is also optimal. If the solution is not optimal, special methods must be used to generate alternate solutions.

The group structure underlying (LP_I) was observed in ($LP3_I$) by Gomory [26]. The main results are summarized here. Denoting the module of all integer points in m-space by M(I) and denoting the module of integer combinations of the columns of B by M(B) Gomory shows that the factor module G = M(I)/M(B) is a finite additive group of D elements where D = |det B|. Furthermore, the positive fractional parts of the columns of the matrix ($B^{-1}N$, $B^{-1}\underline{b}$) generate a finite group F with the group operation addition modulo 1. Similarly, the positive fractional parts of the rows of the same matrix generate a finite additive group F^T. The groups, G, F and F^T are related by the fact that they are all isomorphic to each other. The group structure imbedded in the constraints allows ($LP3_I$) to be viewed as a knapsack problem which must be optimized in the group G. Algorithms which take advantage of the group structure of ($LP3_I$) have been proposed by Gomory [26], White [27], Hu [28], Glover [29], Shapiro [30] and Hefley [31].

It will now be shown that it is possible to proceed as in the case of (LP_I) and generate a hyperbolic group problem. The Martos algorithm is first used to solve (P_C). This makes it possible to partition the constraint matrix into a set of optimal basic columns B and a set of optimal nonbasic column N. Since B is nonsingular

$$\underline{x}_B = B^{-1}(\underline{b} - N\underline{x}_N)$$

Thus the basic vector $\underline{x}_B$ can be eliminated from (P_I) by direct substitution. This yields

$$\max \quad f(\underline{x}_N) = \frac{(\underline{c}_N - \underline{c}_B B^{-1}N)\underline{x}_N + (\underline{c}_B B^{-1}\underline{b} + \alpha)}{(\underline{d}_N - \underline{d}_B B^{-1}N)\underline{x}_N + (\underline{d}_B B^{-1}\underline{b} + \beta)}$$

$$\text{s.t.} \quad \underline{x}_B = B^{-1}(\underline{b} - N\underline{x}_N)$$

$$\underline{x}_B, \underline{x}_N \geq 0 \quad \text{and integer} \qquad (P1_I)$$

As a matter of convenience, the following definitions are made to simplify the statement of $(P1_I)$.

$$z^1 = \underline{c}_B B^{-1}\underline{b} + \alpha$$

$$z^2 = \underline{d}_B B^{-1}\underline{b} + \beta$$

$$\underline{c}_N^* = \underline{c}_N - \underline{c}_B B^{-1}N$$

$$\underline{d}_N^* = \underline{d}_N - \underline{d}_B B^{-1}N$$

Upon proper substitution, $(P1_I)$ becomes

$$\max \quad f(\underline{x}_N) = \frac{\underline{c}_N^*\underline{x}_N + z^1}{\underline{d}_N^*\underline{x}_N + z^2}$$

$$\text{s.t.} \quad \underline{x}_N = B^{-1}(\underline{b} - N\underline{x}_N)$$

$$\underline{x}_B, \underline{x}_N \geq 0 \quad \text{and integer} \qquad (P2_I)$$

Dropping the nonnegativity restriction on $\underline{x}_B$, $(P2_I)$ becomes

$$\max \quad f(\underline{x}_N) = \frac{\underline{c}_N^* \underline{x}_N + z^1}{\underline{d}_N^* \underline{x}_N + z^2}$$

$$\text{s.t.} \quad B^{-1}N\underline{x}_N \equiv B^{-1}\underline{b} \qquad (\text{mod } 1)$$

$$\underline{x}_N \geq 0 \quad \text{and integer} \qquad (P3_I)$$

The group structure present in the constraint of $(LP3_I)$ is also present in the constraint of $(P3_I)$. This follows from the fact that the optimal solution to (P_C) occurs on an extreme point of $\overline{\underline{X}}$ and hence can be characterized by some basis B. The constraint in $(P3_I)$ can therefore be viewed as one constraint which must be satisfied in a finite additive Abelian group. Furthermore, the condition which insures that an optimal solution for $(LP3_I)$ generates a feasible and therefore optimal solution for (LP_I) also holds for the hyperbolic case. That is, if $(\underline{b} - N\underline{x}_N)$ lies in the cone generated by the columns of the optimal basis B then $\underline{x}_B = B_i^{-1}(\underline{b} - N\underline{x}_N) \geq 0$. Thus there exists a class of hyperbolic problems such that the optimal solution to $(P3_I)$ will always generate a feasible and therefore optimal solution for (P_I). This class of problems is analogous to the asymptotic linear integer programming problem.

It will now be shown that an optimal solution for $(P3_I)$ can be obtained by adapting the general algorithm HIP(1) discussed in Chapter 3. It will be useful to first prove the following result.

Lemma 6

Let $\underline{x}_N$ be any feasible solution for $(P3_I)$ such that $(\underline{d}_N^* \underline{x}_N + z^2) > 0$. Then

$$(\underline{c}_N^* \underline{x}_N + z^1)/(\underline{d}_N^* \underline{x}_N + z^2) \leq z^1/z^2$$

Proof:

Suppose the contrary is true. That is

$$(\underline{c}_N^* \underline{x}_N + z^1)/(\underline{d}_N^* \underline{x}_N + z^2) > z^1/z^2$$

Because $(\underline{d}_N^* \underline{x}_N + z^2) > 0$

$$z^2(\underline{c}_N^* \underline{x}_N + z^1) - z^1(\underline{d}_N^* \underline{x}_N + z^2) > 0$$

or equivalently

$$\{z^2(\underline{c}_N - \underline{c}_B B^{-1} N) - z^1(\underline{d}_N - \underline{d}_B B^{-1} N)\}\ \underline{x}_N > 0$$

The components of the vector in brackets are the quantities t_j which are computed in the Marton algorithm to check optimality. The basis B being optimal for (P_C), it must be that $t_j \leq 0\ \forall j \in N$. A contradiction is therefore obtained and the Lemma is proved.

Lemma 6 indicates how a feasible integer point can be found for (PJ_1). It is obtained by solving the following problem:

$$\max\ (z = (z^2 \underline{c}_N^* - z^1 \underline{d}_N^*)\underline{x}_N)$$

$$\text{s.t.}\quad B^{-1} N \underline{x}_N \equiv B^{-1} \underline{b} \pmod 1$$

$$\underline{x}_N \geq 0 \text{ and integer}$$

This problem is equivalent to the problem

$$\min\ (z = \sum_{j \in N} (-t_j) x_j)$$

$$\text{s.t.}\quad B^{-1} N \underline{x}_N \equiv B^{-1} \underline{b} \pmod 1$$

$$\underline{x}_N \geq 0 \text{ and integer} \qquad \text{(GP1)}$$

where t_j is defined in (3.10). Since B is optimal for (P_C), $t_j \leq 0$ $\forall j \in N$. (GP1) is a group knapsack problem and can be solved by any available algorithm. If the Martos algorithm had terminated integer, the solution to (GP1) would be $x_j = 0$ $\forall j \in N$ and the optimal objective function would be zero. If on the other hand, it did not terminate integer, the solution to (GP1) provides a lower bound for the maximum of the hyperbolic objective function. Thus if there exists a better integer point than the one which is optimal for (GP1), it must yield an objective function value which lies in the interval $[(\underline{c}_N^* \underline{x}_N^{(1)} + z^1)/(\underline{d}_N^* \underline{x}_N^{(1)} + z^2),\ z^1/z^2]$ where $\underline{x}_N^{(1)}$ is optimal for (GP1). The existence of such a point can be established by considering the problem

$$\begin{aligned} &\max \{(\underline{d}_N^* \underline{x}_N^{(1)} + z^2)(\underline{c}_N^* \underline{x}_N + z^1) - (\underline{c}_N^* \underline{x}_N^{(1)} + z^1)(\underline{d}_N^* \underline{x}_N + z^2)\} \\ &\text{s.t.} \quad B^{-1} N \underline{x}_N \equiv B^{-1} \underline{b} \qquad \text{(mod 1)} \\ &\qquad \underline{x}_N \geq 0 \quad \text{and integer} \end{aligned} \tag{GP2}$$

An equivalent problem for (GP2) is obtained by multiplying the objective function by (-1). After rearranging terms, the problem becomes

$$\begin{aligned} &\min \{z^2(\underline{c}_N^* \underline{x}_N^{(1)} + z^1) - z^1(\underline{d}_N^* \underline{x}_N^{(1)} + z^2) \\ &\qquad + [(\underline{c}_N^* \underline{x}_N^{(1)} + z^1)\underline{d}_N^* - (\underline{d}_N^* \underline{x}_N^{(1)} + z^2)\underline{c}_N^*]\underline{x}_N\} \\ &\text{s.t.} \quad B^{-1} N \underline{x}_N \equiv B^{-1} \underline{b} \qquad \text{(mod 1)} \\ &\qquad \underline{x}_N \geq 0 \quad \text{and integer} \end{aligned} \tag{GP3}$$

The strategy for finding a sequence of feasible integer points which improve the objective function of $(P3_1)$ is reviewed. First an

upper bound for the objective function is used to generate a group knapsack problem. The solution to this problem provides a lower bound for the maximum of the objective function. It now remains to determine whether or not there exist a feasible integer $\underline{x}_N$ such that

$$\frac{(\underline{c}_N^* \underline{x}_N^{(1)} + z^1)}{(\underline{d}_N^* \underline{x}_N^{(1)} + z^2)} < \frac{(\underline{c}_N^* \underline{x}_N + z^1)}{(\underline{d}_N^* \underline{x}_N + z^2)} < \frac{z^1}{z^2}$$

This can be accomplished by using the lower bound in this last inequality to generate a new group knapsack problem. This problem has the same structure as (GP2). If its optimal objective function value is strictly greater than zero, an improved solution for $(P3_1)$ has been found. If the optimal objective function value is zero, the current solution is also optimal. A statement of the algorithm is given below.

Algorithm GIP(2)

1. Solve the hyperbolic program using the Martos algorithm. If the algorithm terminates with an integer solution, stop; otherwise go to step 2.
2. Use the optimal t_j's and formulate the problem

$$\min \sum_{j \in N} (-t_j) x_j$$

$$\text{s.t.} \quad B^{-1} N \underline{x}_N \equiv B^{-1} \underline{b} \pmod 1$$

$$\underline{x}_N \geq 0 \text{ and integer}$$

Solve using any available algorithm and call the solution $\underline{x}_N^{(1)}$. If $(\underline{d}_N^* \underline{x}_N^{(1)} + z^2) \leq 0$, stop; otherwise set $j=1$ and continue.

3. Formulate the problem

$$\min \{F^{(j+1)} = G^{(j)} + [(\underline{c}_N^* \underline{x}_N^{(j)} + z^1)\underline{d}_N^* - (\underline{d}_N^* \underline{x}_N^{(j)} + z^2)\underline{c}_N^*]\underline{x}_N\}$$

$$\text{s.t.} \quad B^{-1}N\underline{x}_N \equiv B^{-1}\underline{b} \pmod 1$$

$$\underline{x}_N \geq 0 \text{ and integer}$$

where $G^{(j)} = z^2(\underline{c}_N^* \underline{x}_N^{(j)} + z^1) - z^1(\underline{d}_N^* \underline{x}_N^{(j)} + z^2)$. Solve using any available algorithm and call the solution $\underline{x}_N^{(j+1)}$. If $(\underline{d}_N^* \underline{x}_N^{(j+1)} + z^2) \leq 0$, stop; otherwise continue.

4. If $F^{(j+1)} = 0$, the current solution is optimal for $(P3_I)$; otherwise set $j = j+1$ and return to step 3.

An example is solved with HIP(5) in Appendix D.

To prove that HIP(5) converges in a finite number of steps, each linear group problem must have a finite solution. For the time being, this will be assumed to be the case. This assumption will be studied in greater detail later on. Furthermore, it must suffice to consider only a finite number of feasible points. To see that this is the case when each linear group problem is finite, note that each group problem has exactly the same constraint. That is

$$B^{-1}N\underline{x}_N \equiv B^{-1}\underline{b} \pmod 1$$

$$\underline{x}_N \geq 0 \text{ and integer}$$

This constraint can also be written as

$$\sum_{i \in N} g_i x_i = g_0$$

$$x_i \geq 0 \quad \forall i \in N$$

where g_i $\forall i \in N$ and g_0 are elements of a finite Abelian group with addition modulo $D = |\det B|$. If the vector with components x_i, $i \in N$, is a solution for a linear group problem then the vector with components $(x_i + n_i D)$, $i \in N$, is also a feasible solution. In other words

$$\sum_{i \in N} g_i (x_i + n_i D) = g_0$$

Furthermore if the vector with components x_i^*, $i \in N$, is an optimal solution for the group problem then all vectors with components of the form $(x_i^* + n_i D)$ $i \in N$ yield worse objective function values than vectors with components x_i^*. Thus if given a feasible solution vector with some component $x_i > D$, it is always possible to construct a better solution by replacing x_i by x_i' with $0 \leq x_i' \leq (D - 1)$. This can always be accomplished by subtracting the largest possible multiple of D that will keep x_i' positive. It follows therefore that an optimal solution to the linear group problem must have $0 \leq x_i \leq (D - 1)$ $\forall i \in N$ and the feasible region for each linear problem can therefore be bounded.

The discussion thus far implies that if each linear group problem generated by HIP(5) is finite, its solution can be obtained by considering only a finite number of points. Furthermore the finite number of feasible points is exactly the same for each linear group problem. Recalling HIP(5), at any iteration, either there is a strict improvement in the hyperbolic objective function or the solution is repeated. If the solution is repeated, it is also optimal. The algorithm terminates in a finite number of steps because only finitely many feasible points are considered.

The assumption that each linear group problem has a finite solution is now examined in greater detail. Finiteness of every linear group problem is insured if all the components of the vector $[(\underline{c}^*_N \underline{x}^{(j)}_N + z^1)\underline{d}^*_N - (\underline{d}^*_N \underline{x}^{(j)}_N - (\underline{d}^*_N \underline{x}^{(j)}_N + z^2)\underline{c}^*_N]$ are nonnegative at each iteration. To find the conditions when this is true suppose that for some j one of the components is negative. That is

$$(\underline{c}^*_N \underline{x}^{(j)}_N + z^1)d^*_j - (\underline{d}^*_N \underline{x}^{(j)}_N + z^2)c^*_j < 0 \tag{3.15}$$

Since $(\underline{d}^*_N \underline{x}^{(j)}_N + z^1) > 0$, (3.15) can be written as

$$(\underline{c}^*_N \underline{x}^{(j)}_N + z^1)/(\underline{d}^*_N \underline{x}^{(j)}_N + z^2) \cdot d^*_j < c^*_j \tag{3.16}$$

(3.16) is an equivalent statement for (3.15). Thus whenever (3.16) leads to a contradiction, (3.15) must be nonnegative. First suppose $d^*_j = 0$. Appealing to (3.16), $c^*_j > 0$. Recalling that $t_j \leq 0$ at the optimal continuous hyperbolic solution, a contradiction is obtained since $t_j = z^2 c^*_j - z^1 d^*_j > 0$. When in (3.16) $d^*_j < 0$, either $c^*_j \geq 0$ or $c^*_j < 0$. When $d^*_j < 0$ and $c^*_j \geq 0$ a contradiction is obtained since $t_j > 0$. When $d^*_j < 0$ and $c^*_j < 0$, (3.16) implies

$$(\underline{c}^*_N \underline{x}^{(j)}_N + z^1)/(\underline{d}^*_N \underline{x}^{(j)}_N + z^2) > (-c^*_j)/(-d^*_j) \tag{3.17}$$

Lemma 6 together with (3.17) implies that

$$z^1/z^2 > (-c^*_j)/(-d^*_j)$$

Since $z^2 > 0$ and $(-d^*_j) > 0$, it follows that

$$z^2 c^*_j - z^1 d^*_j > 0 \tag{3.18}$$

A contradiction is achieved since (3.18) implies $t_j > 0$.

The last case to consider is when $d_j^* > 0$. First suppose $(\underline{c}_N^* \underline{x}_N^{(j)} + z^1) \geq 0$. This implies $c_j^* > 0$ and (3.16) can be written as

$$(\underline{c}_N^* \underline{x}_N^{(j)} + z^1)/(\underline{d}_N^* \underline{x}_N^{(j)} + z^2) < c_j^*/d_j^* \tag{3.19}$$

Appealing to Lemma 6, (3.19) implies either

a) $$(\underline{c}_N^* \underline{x}_N^{(j)} + z^1)/(\underline{d}_N^* \underline{x}_N^{(j)} + z^2) \leq z^1/z^2 < c_j^*/d_j^*$$

or

b) $$(\underline{c}_N^* \underline{x}_N^{(j)} + z^1)/(\underline{d}_N^* \underline{x}_N^{(j)} + z^2) < c_j^*/d_j^* < z^1/z^2$$

If a) occurs, once again $t_j > 0$ and a contradiction is obtained. If b) occurs, (3.15) is true and hence the linear group problem has an unbounded solution. If $\underline{x}_N^{(j)}$ is feasible for the hyperbolic group problem, feasibility is maintained by replacing the jth component of the solution vector by $(x_{Nj}^{(j)} + nD)$. Therefore b) implies that if n approaches infinity, the hyperbolic function for the group problem approaches $(c_j^*)/(d_j^*)$ in the limit. Letting n get very large implies that the jth nonbasic variable will get very large. Thus there exist some n' such that if $n > n'$ feasibility will be violated in (P_I).

Now suppose $d_j^* > 0$ and $(\underline{c}_N^* \underline{x}_N^{(j)} + z^1) < 0$. This implies $c_j^* < 0$ and (3.19) still holds. Thus the argument given above can be repeated for this case.

The algorithm HIP(5) can therefore terminate in several ways. If all the linear group problems are finite, HIP(5) converges in a finite number of steps. If the solution is feasible for (P_I), it is also optimal. If at some point in the algorithm $(\underline{d}_N^* \underline{x}_N + z^2) \leq 0$, the

denominator in (P_I) will also be less than or equal to zero. This however contradicts the original assumption that $(\underline{d}\ \underline{x} + \beta) > 0$ $\forall \underline{x} \in \overline{\underline{X}}$. Thus $(\underline{d}^*_N \underline{x}_N + z^2) \leq 0$ implies that $(\underline{x}_B, \underline{x}_N)$ is not feasible for (P_I) and the algorithm terminates. If at some iteration a linear group problem with an unbounded solution is generated, feasibility will also be violated in (P_I).

One way to proceed if the group problem leads to infeasibility is to generate faces of the corner polyhedra (see Gomory [26]) and reoptimize the continuous hyperbolic problem. If this leads to an integer solution, it is also optimal. If it is not, a new group problem can be generated and the procedure is repeated.

As of yet there is no evidence to indicate whether or not there exists a class of hyperbolic group problems that will always terminate at step 4 of HIP(5). To determine whether or not HIP(5) is as useful as the group theoretic solution to the linear integer problem will require implementing an algorithm and testing it with a wide variety of problems.

CHAPTER 4

SOME SPECIAL PROBLEMS

4.1 The Hyperbolic Programming Problem with Bounded Integer Variables Only

The problem considered here is

$$\max \{f(\underline{x}) = (\sum_{i=1}^{n} c_i x_i + \alpha)/(\sum_{i=1}^{n} d_i x_i + \beta)\}$$

$$\text{s.t.} \quad 0 \leq x_i \leq m_i \qquad i=1,\ldots,n$$

$$x_i \text{ integer } \forall i \qquad (P'B_I)$$

It is assumed that $\alpha > 0$, $\beta > 0$ and $c_i > 0$, $d_i > 0$ $\forall i$. $(P'B_I)$ has a hyperbolic objective function and its feasible region has a finite number of feasible integer points. This suggests that the general algorithm HIP(1) can be used to solve $(P'B_I)$.

If the feasible region for $(P'B_I)$ does not have all integer extreme points, it must be that some of the upper bounds m_i are not integer. Replacing m_i by the largest integer smaller than m_i will insure that the reduced feasible region has all integer extreme points and furthermore that no feasible integer points for the original problem are eliminated. A special version of HIP(1) which is given below can therefore be used to solve $(P'B_I)$.

Algorithm HIP(6)

1. Let $\alpha^{(0)} = \alpha$ and $\beta^{(0)} = \beta$. Set j=1.
2. Generate the following problem:

$$\max \{z_j = \sum_{i=1} (\beta^{(j-1)}c_i - \alpha^{(j-1)}d_i)x_i^{(j)} + (\beta^{(j-1)}\alpha - \alpha^{(j-1)}\beta)$$

$$\text{s.t.} \quad x_i = 0,1,\ldots,m_i \quad \forall i$$

3. The solution to the problem in step 2 is obtained by letting $x_i^{(j)} = m_i$ if $(\beta^{(j-1)}c_i - \alpha^{(j-1)}d_i) > 0$ and letting $x_i^{(j)} = 0$ if $(\beta^{(j-1)}c_i - \alpha^{(j-1)}d_i) \leq 0$.

4. If $\max z_j = 0$, the current solution is optimal; otherwise compute

$$\alpha^{(j)} = \sum_{i=1}^{n} c_i x_i^{(j)} + \alpha$$

$$\beta^{(j)} = \sum_{i=1}^{n} d_i x_i^{(j)} + \beta$$

Let $j = j+1$ and return to step 2.

Clearly, HIP(6) converges in a finite number of steps because it is a special case of HIP(1).

HIP(6) is of special interest for several reasons. First note the algorithm is still valid if the integer restriction is removed. Second, the ease with which the subproblems can be solved makes the algorithm computationally attractive. Third, and perhaps most important, is that the number of variables in each succeeding subproblem decreases. More particularly once a variable is set to its lower bound at some iteration, it can be eliminated from the problem. To see that this is a valid step suppose x_j is set to zero at iteration k. This implies that

$$(c_j - (\alpha^{(k-1)}/\beta^{(k-1)})d_j) \leq 0$$

By virtue of the algorithm however

$$(\alpha^{(k)}/\beta^{(k)}) \geq (\alpha^{(k-1)}/\beta^{(k-1)}) \qquad k=1,2,\ldots \qquad (4.1)$$

It follows therefore that

$$(c_j - (\alpha^{(k)}/\beta^{(k)})d_j) < 0$$

and the variable x_j remains zero at the (k+1)th iteration. From (4.1) the variable x_j will also remain at its lower bound in the remaining subproblems. The dimension of the subproblems therefore decreases at each iteration. When no more variables can be eliminated from a subproblem, the current solution is also optimal.

Also of interest is that $(P'B_I)$ can be viewed as bivalent problem. That is each variable can assume only one of two values. This becomes apparent once the structure of the solution is known. If $(P'B_I)$ is viewed as a bivalent problem, an algorithm proposed by Hammer and Rudeanu [32] can also be used to solve the problem.

The Hammer and Rudeanu algorithm differs from HIP(6) in that it builds a list of active variables whereas HIP(6) starts with the largest possible list of active variables and reduces it at each iteration.

An example is used to demonstrate HIP(6) in Appendix E.

4.2 The Hyperbolic Knapsack Problem

In this section, the linear fractional or hyperbolic knapsack problem is studied. This problem has the following form:

$$\max \{f(\underline{x}) = (\alpha + \sum_{i=1}^{n} c_i x_i)/(\beta + \sum_{i=1}^{n} d_i x_i)\}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} a_i x_i \leq b$$

$$x_i \geq 0 \text{ and integer} \qquad (P_K)$$

In this discussion, it will be assumed that $\alpha > 0$, $\beta > 0$ and $c_i > 0$, $d_i > 0$ $\forall i$. To insure that (P_K) has a bounded feasible region, it will also be assumed that $a_i \geq 0$ $\forall i$.

(P_K) is of special importance because it arises in many models and Bradley [33] has recently shown how to reduce m linear constraints into one equivalent linear constraint for integer problems.

Since (P_K) has a bounded feasible region, its solution can be found by specializing HIP(1). Such an algorithm is given below.

Algorithm HIP(7)

1. Let $\alpha^{(0)} = \alpha$ and $\beta^{(0)} = \beta$. Set $j=1$.
2. Generate the following problem:

$$\max \{z_j = \sum_{i=1}^{n} (\beta^{(j-1)} c_i - \alpha^{(j-1)} d_i) x_i^{(j)} + (\beta^{(j-1)} \alpha - \alpha^{(j-1)} \beta)\}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} a_i x_i^{(j)} \leq b$$

$$x_i^{(j)} \geq 0 \text{ and integer}$$

3. Solve the linear knapsack problem by any convenient method and call the solution $\underline{x}^{(j)}$.
4. If $\max z_j > 0$ let

$$\alpha^{(j)} = \alpha + \sum_{i=1}^{n} c_i x_i^{(j)}$$

$$\beta^{(j)} = \beta + \sum_{i=1}^{n} d_i x_i$$

Set $j = j+1$ and return to step 2. If $\max z_j = 0$ stop. The current solution is optimal.

HIP(7) will converge to the optimal solution of (P_K) in a finite number of steps because it is a special case of HIP(1). More important, for exactly the same reasons given in Section 4.1, once a variable is set equal to it lower bound it can be eliminated from the rest of the subproblems generated by the algorithm. Thus HIP(7) generates a sequence of linear knapsack problems which decrease in the number of variables. When the number of variables can no longer be decreased, the current solution is also optimal.

The result presented in this section can also be extended to continuous hyperbolic programming problems with positive constraint matrices. That is if the matrix A in (P_C) is positive, the number of variables in each linear problem generated by HIP(1) can be decreased at each iteration.

CHAPTER 5

SOME COMPUTATIONAL RESULTS, SUMMARY AND CONCLUSIONS

5.1 Computational Experience

The computational experience reported here is based on an experimental computer code for the all integer cutting plane algorithm HIP(3). The computer program was run on an IBM 360/65 and is based on R. E. Woolsey's [34] IPSC code.

To determine the difficulty of the test problems, the number of iterations required for the first linear integer problem generated by HIP(3) were compared to the iteration statistic for (P_I). The number of linear integer programs which had to be solved before an optimal solution for (P_I) was recognized, was also recorded.

The test problems were constructed so that the first problem generated by HIP(3) was a linear integer problem of known difficulty. That is if the problem

$$\max \{f(\underline{x}') = \underline{c}'\underline{x}'/(\underline{d}'\underline{x}' + 1)\}$$

$$\text{s.t.} \quad A'\underline{x}' \leq \underline{b}$$

$$\underline{x}' \geq 0 \quad \text{and integer} \qquad (P'T_I)$$

is solved with the aid of HIP(3), the first subproblem is

$$\max \{z_1 = \underline{c}'\underline{x}'\}$$

$$\text{s.t.} \quad A'\underline{x}' \leq \underline{b}$$

$$\underline{x}' \geq 0 \quad \text{and integer}$$

The linear functionals appearing in the numerator of $(P'T_I)$ were taken from Trauth and Woolsey [35]. The components of the vector $\underline{d}'$ were chosen arbitrarily. The test problems and solutions obtained via HIP(3) are given in Appendix F. The results are summarized in Table 2.

Based on a sample of 14 test problems, the average number of linear integer problems which had to be solved to reach optimality was 2.9. Problem 4 did not converge but a lower bound for the maximum of the objective function value was obtained after 62 iterations.(1) In 12 out of 13 test problems which converged, the optimal solution was available before it was recognized. Problem 9 was the exception. When the optimal solution was available before it was recognized, the average number of additional iterations(1) required to identify the solution was 10.9. The average number of iterations to solve the first linear problem is 46.2 and the average number of iterations(1) to solve the hyperbolic problem is 349.3. Using iterations(1) as a criterion, the average test problem is 7.6 times more difficult than the first linear problem. This number is considerably reduced however if problems 2, 3 and 5 are deleted from the sample.

5.2 Summary and Conclusions

In Chapter 2, it was established that the solution to (P'_C) can be obtained by rotating a hyperplane about an (n-2) dimensional subspace in E^n.

---

(1) Iteration as it is used here is in the context of the Gomory All-Integer algorithm.

TABLE 2

	1st Linear Problem		2nd Linear Problem		3rd Linear Problem		4th Linear Problem				
Problem	f(x)	Iter.	f(x)	Iter.	f(x)	Iter.	f(x)	Iter.	Total Time (sec)	Total Iter.	Total Iter. Divided By 1st Prob. Iter.
1.	50/8	45	47/4	59	38/3	32	38/3*	.1	4.58	135	3.14
2.	45/7	74	37/3	960	37/3*	18			17.62	1052	14.25
3.	41/6	68	35/3	1190	35/3*	23			20.98	1281	18.85
4.	37/7	62	-	7050							
5.	33/5	50	30/3	975	30/3*	35			18.66	1058	21.10
6.	28/4	41	20/2	66	20/2*	7			4.40	114	2.78
7.	24/5	47	18/2	144	18/2*	8			5.79	199	4.24
8.	20/6	41	17/2	45	17/2*	11			4.09	97	2.36
9.	10/5	38	8/3*	85					3.69	123	3.24
10.	7/29	35	6/19	106	6/19*	10			2.62	151	4.31
11.	8/33	47	7/22	66	7/22*	1			2.28	109	2.21
12.	10/47	87	9/28	70	9/28*	2			2.52	159	1.83
13.	8/57	26	6/19	19	6/19*	11			1.99	56	2.15
14.	8/33	4	8/33*	3							

*Denotes optimal solution

In Chapter 3, a general algorithm for solving (P_I') (or (P_I)) which requires solving a finite sequence of linear integer programs was proposed. It was also shown that this general algorithm can be used to solve the more general problem (P_G').

Cutting plane algorithms were also studied in detail. These algorithms were viewed as special cases of the general algorithm HIP(1). Because the optimal solution to (P_C') lies on at least one extreme point of the feasible region, the cutting plane algorithms are in the spirit of Gomory's fractional and all-integer algorithms. That is, if the solution to (P_C') is not all integer, a cutting plane which cuts the current extreme point is appended to the constraint set and the objective function is reoptimized on the reduced feasible set. This procedure is repeated until an all-integer solution is obtained. In the worst case, a cutting plane algorithm will generate all the extreme points of the integer hull contained in the original feasible region. Note that if the original matrix is totally unimodular, the solution to (P_C') is all integer and hence also optimal for (P_I'). In this case cuts never have to be added.

Based on the geometry of hyperbolic programming, the solution to (P_I') can also be obtained by rotating a hyperplane about an (n-2) dimensional subspace in E^n. Although the algorithms derived from HIP(1) do not implement rotations directly, they achieve the same thing. An equivalent procedure for rotating a hyperplane T_1^0 about an (n-2) dimensional subspace of E^n until it hits a point $\underline{x}_2'$ is to first move T_1^0 parallel to itself in the direction of $\underline{x}_2'$ until it hits $\underline{x}_2'$. The

second step is to construct a hyperplane T_2^o which passes through the (n-2) dimensional subspace and the point $\underline{x}_2'$. Clearly the end result is a rotation of T_1^o through a positive angle, the final position of the hyperplane being T_2^o. Algorithms derived from HIP(1) selectively determine finite sequences of hyperplanes to be translated. This procedure continues until the optimal solution is recognized.

In Chapter 3, it is also shown that the group theoretic approach can be used to solve (P_1') (or (P_I)). This result depends on the fact that the solution to (P_C) can be expressed in terms of an optimal basis. A hyperbolic group problem is obtained and it is shown that if the solution to (P_I) exists, the group problem has a bounded feasible region. This makes it possible to specialize HIP(1) for solving the hyperbolic group problem. A useful consequence of the group theoretic approach is that the master polyhedra's generated by Gomory [26] for solving the linear integer program can be used for solving (P_I).

In Chapter 4, two special problems were considered. A hyperbolic problem with bounded integer variables only was discussed first. An algorithm for solving the problem which makes it possible to decrease the number of variables at each iteration is proposed. The second special problem is a hyperbolic knapsack problem. It is shown that the solution to this problem can be obtained by solving a finite sequence of linear knapsack problems decreasing in the number of variables at each iteration.

5.3 Further Research and Extensions

The most obvious extension of this work is with regard to

mixed-integer programming. For example, can a partitioning procedure as suggested by Benders [36] for the linear mixed-integer problem be implemented to solve the problem

$$\max \{f(\underline{x},\underline{y}) = (\underline{c}_1\underline{x} + \underline{c}_2\underline{y} + \alpha)/(\underline{d}_1\underline{x} + \underline{d}_2\underline{y} + \beta)\}$$

$$\text{s.t.} \quad A_1\underline{x} + A_2\underline{y} \leq \underline{b}$$

$$\underline{x} \geq 0$$

$$\underline{y} \geq 0 \quad \text{and integer}$$

Another open area with regard to this problem is the cutting plane approach.

The Generalized Lagrange Multiplier approach of Everett also is worthy of consideration with regard to (P_1). This method is useful when the constraints are not binding to the degree indicated by the problem statement. The resulting problem has a very special structure and indicates that special algorithms may be derived.

The structure of the hyperbolic group problem also seems to indicate that special algorithms can be obtained for solving this problem. An interesting question which arises is whether there is a network interpretation for the hyperbolic group problem.

Further research also needs to be done to establish the efficiency of hyperbolic integer programming algorithms. This of course necessitates computer programming efforts to obtain numerical results for a wide variety of test problems.

APPENDICES

APPENDIX A

MARTOS' ALGORITHM: AN EXAMPLE

A column tableau with negative multipliers is adapted for (P_C'). The Martos optimality criterion t_j is easily computed at each iteration according to

$$t_j = n_{00}d_{0j} - d_{00}n_{0j}$$

where n_{00}, d_{0j}, d_{00}, and n_{0j} are defined in Table 1. The format introduced here is useful for implementing cutting plane algorithms for solving integer problems. Consider the following problem:

$$\max \{f(x_1,x_2) = (5x_1 + 3x_2 + 1)/(5x_1 + 2x_2 + 1)\}$$

$$\text{s.t.} \quad 3x_1 + 5x_2 \leq 16$$

$$5x_1 + 2x_2 \leq 11$$

$$x_1, x_2 \geq 0$$

Slack variables x_3 and x_4 are introduced and the constraints are rewritten as

$$x_3 = 16 - 3x_1 - 5x_2$$

$$x_4 = 11 - 5x_1 - 2x_2$$

Table 3 is a statement of the original problem and Table 4 is the optimal tableau. A side calculation shows it was obtained in one iteration. The pivot element in Table 3 is circled.

TABLE 3

t_j		0	1
	1	$-x_1$	$-x_2$
N	1	-5	-3
D	1	-5	-2
x_1	0	-1	0
x_2	0	0	-1
x_3	16	3	5
x_4	11	5	2

TABLE 4

t_j		-83/5	-1/5
	1	$-x_1$	$-x_3$
N	53/5	-16/5	3/5
D	37/5	-19/5	2/5
x_1	0	-1	0
x_2	16/5	3/5	1/5
x_3	0	0	-1
x_4	23/5	19/5	2/5

APPENDIX B

THE FRACTIONAL CUTTING PLANE ALGORITHM: AN EXAMPLE

The example given in Appendix A with the additional restriction that x_1 and x_2 must be integers is used to demonstrate HIP(2). Table 5 is the noninteger optimal tableau with the z-row and Gomory cut row appended. The source row is found to be the x_2 row and the pivot element is circled. Pivoting results in Table 6. Calculating t_j $\forall j \in N$ at this point verifies that the current solution is optimal for (P_I).

TABLE 5

t_j		-83/5	-1/5
	1	$-x_1$	$-x_3$
N	53/5	-16/5	3/5
D	37/5	-19/5	2/5
z	0	83/5	1/5
x_1	0	-1	0
x_2	16/5	3/5	1/5
x_3	0	0	-1
x_4	23/5	19/5	-2/5
s_1	-1/5	-3/5	-1/5

TABLE 6

t_j		-15	-1
	1	$-x_1$	$-s_1$
N	10	- 5	3
z	-1	16	1
x_1	0	-1	0
x_2	3	0	1
x_3	1	3	-5
x_4	5	5	-2
s_1	0	0	-1

APPENDIX C

THE ALL-INTEGER CUTTING PLANE ALGORITHM: AN EXAMPLE

The example used in Appendix A with the additional restriction that x_1 and x_2 must be integers is used to demonstrate HIP(3). Option B in step 3 is followed.

Table 7 is a statement of the original problem with the z-row included. An additional row exhibiting the constraint

$$x_5 = 20 - x_1 - x_2$$

where 20 is an upper bound on the sum of the current nonbasis variables is also appended. Table 8 is the dual feasible tableau with respect to the z-row which results from the first pivot operation. The x_3 row is the source row and s_1 is the first cut appended to the original set of constraints. Reoptimizing z after s_1 is appended yields the optimal tableau which is displayed in Table 9.

TABLE 7

t_j		0	1
	1	$-x_1$	$-x_2$
N	1	-5	-3
D	1	-5	-2
z	0	0	-1
x_1	0	-1	0
x_2	0	0	-1
x_3	16	3	5
x_4	11	5	2
x_5	20	1	1

TABLE 8

t_j			
	1	$-x_1$	$-x_5$
N	61	-5	3
D	41	-5	2
z	20	0	1
x_1	0	-1	0
x_2	20	0	1
x_3	-84	3	-5
x_4	-29	5	-2
x_5	9	1	-1
s_1	-17	0	-1

TABLE 9

t_j		-15	-1
	1	$-\lambda_1$	$-s_1$
N	10	-5	3
D	7	-5	2
z	3	0	1
x_1	0	-1	0
x_2	3	0	1
x_3	1	3	-5
x_4	5	5	-2
x_5	17	1	-1
s_1	0	0	-1

APPENDIX D

A GROUP THEORETIC ALGORITHM

The integer version of the example used in Appendix A is again used here. The first group knapsack problem which must be solved is

$$\min \{(83/5)x_1 + (1/5)x_3\}$$

$$\text{s.t.} \quad \begin{pmatrix} 3/5 \\ 4/5 \end{pmatrix} x_1 + \begin{pmatrix} 1/5 \\ 3/5 \end{pmatrix} x_3 = \begin{pmatrix} 1/5 \\ 3/5 \end{pmatrix} \pmod 1$$

$$x_1, x_3 \geq 0 \text{ and integer}$$

This problem is easily obtained from Table 4. The group constraint is generated by taking the positive fractional parts of the rows corresponding to the basic variable and the objective function costs are $(-t_1)$ and $(-t_3)$. The solution to this simple problem can be found by inspection if it is noted that the right-hand side of the group equation is the same as the group element which multiplies the variable which has the smallest cost in the objective function. The optimal solution is therefore $x_1 = 0$ and $x_3 = 1$. Since $x_2 = 3$ and $x_4 = 5$, this solution is also feasible for the hyperbolic integer program. Proceeding according to the algorithm HIP(5), the linear group problem is found to be

$$\min \{-1/5 + (78/5)x_1 + (1/5)x_3\}$$

$$\text{s.t.} \quad \begin{pmatrix} 3/5 \\ 4/5 \end{pmatrix} x_1 + \begin{pmatrix} 1/5 \\ 3/5 \end{pmatrix} x_3 \equiv \begin{pmatrix} 1/5 \\ 3/5 \end{pmatrix} \pmod 1$$

$$x_1, x_3 \geq 0 \quad \text{and integer}$$

The solution to this problem is found to be identical to the first solution and F = 0. The current solution is therefore also optimal for the hyperbolic integer program.

APPENDIX E

HIP(6): AN EXAMPLE

The ease with which HIP(6) can be implemented is demonstrated with the example below.

$$\max \quad \frac{4x_1 + x_2 + 3x_3 + 2x_4 + 2x_5 + x_6 + 3}{7x_1 + x_2 + 8x_3 + 3x_4 + 5x_5 + 2x_6 + 7}$$

s.t. $x_1 = 0,1,2$ $\quad x_4 = 0,1,2,3,4,5$

$x_2 = 0,1,2,3$ $\quad x_5 = 0,1,2,3,4$

$x_3 = 0,1$ $\quad x_6 = 0,1,2$

Applying the algorithm, the first linear objective to be maximized is

$$\max \{7x_1 + 4x_2 - 3x_3 + 5x_4 - x_5 + x_6\}$$

Following the algorithm, the first solution is $x_1 = 2$, $x_2 = 3$, $x_3 = 0$, $x_4 = 5$, $x_5 = 0$, and $x_6 = 2$. The new constants in the second problem are $\alpha^{(1)} = 26$ and $\beta^{(1)} = 43$. The variables x_3 and x_5 are zero and thus are eliminated from the remaining problems. The second linear objective to be maximized is

$$\max \{-20x_1 + 17x_2 + 8x_4 - 9x_6\}$$

The second solution is therefore $x_1 = 0$, $x_2 = 3$, $x_4 = 5$, $x_6 = 0$ and $x_3 = x_5 = 0$. The new constants in the third problem are $\alpha^{(2)} = 16$ and $\beta^{(2)} = 25$, and the variables x_1 and x_6 are eliminated. The third linear objective to be maximized is

$$\max \{9x_2 + 2x_4\}$$

Since no more variables can be eliminated, the optimal solution is $x_2 = 3$, $x_4 = 5$ and $x_1 = x_3 = x_5 = x_6 = 0$. The optimal objective function value is (16/25).

APPENDIX F

SOME COMPUTATIONAL RESULTS

The computational results summarized in Chapter 5 are based on the following set of problems:

Test Problems

$$\max \quad f(\underline{x}) = \frac{20x_1+18x_2+17x_3+15x_4+15x_5+10x_6+3x_7+3x_8+x_9+x_{10}}{x_1+\ x_2+\ x_3+\ x_4+\ x_5+\ x_6+\ x_7+\ x_8+x_9+x_{10}+\ 1}$$

$$\text{s.t.} \quad x_i = 0,1 \qquad i=1,2,\ldots,10$$

$$30x_1+25x_2+20x_3+18x_4+17x_5+11x_6+5x_7+2x_8+x_9+x_{10} \leq b$$

Problem	1	2	3	4	5	6	7	8	9
b	55	50	45	40	35	30	25	20	10

$$\max \ [f(\underline{A}) = \frac{A_1 + A_2 + A_3}{3A_1 + 7A_2 + 4A_3 + 1}$$

$$\begin{aligned} \text{s.t.} \quad & A_1 + 2A_2 + 2A_3 + 2F_1 + 3F_2 \leq B_1 \\ & 2A_1 + A_2 + 2A_3 + 3F_1 + 2F_2 \leq B_2 \\ & A_1 - R_1F_1 \leq 0 \\ & A_2 - R_2F_2 \leq 0 \end{aligned}$$

$A_1, A_2, A_3, F_1, F_2 \geq 0$ and integer

Problem	R_1	R_2	B_1	B_2
10	6	7	18	15
11	9	7	18	17
12	9	9	21	21
13	6	8	19	15

Problem 14

The objective function for this problem is the same as in problems (10-13). The feasible region is given below.

$$
\begin{aligned}
A_1 + A_2 + \quad & 2F_1 + 2F_2 \quad \leq 10 \\
A_1 \quad + A_3 + \; & 2F_1 \quad + 2F_3 \leq 10 \\
A_2 + A_3 \quad & + 2F_2 + 2F_3 \leq 10 \\
A_1 \quad - \; & 8F_1 \quad \leq 0 \\
A_2 \quad & - 8F_2 \quad \leq 0 \\
A_3 \quad & - 8F_3 \leq 0
\end{aligned}
$$

$A_1, A_2, A_3, F_1, F_2, F_3 \geq 0$ and integer

The solutions to problems (1-9) are displayed in Table 10 and those to problems (10-14) in Table 11.

TABLE 10

Problem	f(x)	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
1.	38/3	1	1	0	0	0	0	0	0	0	0
2.	37/3	1	0	1	0	0	0	0	0	0	0
3.	35/3	0	1	1	0	0	0	0	0	0	0
4.	-	-	-	-	-	-	-	-	-	-	-
5.	30/3	0	0	0	1	1	0	0	0	0	0
6.	20/2	1	0	0	0	0	0	0	0	0	0
7.	18/2	0	1	0	0	0	0	0	0	0	0
8.	17/2	0	0	1	0	0	0	0	0	0	0
9.	8/3	0	0	0	0	0	0	1	1	0	0

TABLE 11

Problem	f(A)	A_1	A_2	A_3	F_1	F_2	F_3
10.	6/19	6	0	0	1	0	-
11.	7/22	7	0	0	1	0	-
12.	9/28	9	0	0	1	0	-
13.	6/19	6	0	0	1	0	-
14.	8/33	0	0	8	0	0	1

REFERENCES

1. Y. Almogy and O. Levin, "The Fractional Fixed-Charge Problem," Operations Research, Statistics and Economics Mimeograph Series No. 57, TECHNION-Israel Institute of Technology, Faculty of Industrial and Management Engineering, December, 1969.

2. B. Martos, "Hyperbolic Programming," Naval Research Logistics Quarterly, Vol. 11, Nos. 2, 3, pp. 135-155 (1964).

3. G. B. Dantzig, W. Blattner, M. R. Rao, "Finding a Cycle in a Graph with Minimum Cost to Time Ratio with Applications to a Ship Routing Problem," pp. 77-83 in Theory of Graphs International Symposium, Dunod, Paris and Gordon and Breach, New York, 1966.

4. B. Fox, "Finding Minimal Cost-Time Ratio Circuits," Operations Research, Vol. 17, No. 3, pp. 546-550 (1969).

5. P. C. Gilmore and R. E. Gomory, "A Linear Programming Approach to the Cutting Stock Problem," Operations Research, Vol. 11, pp. 863-888 (1963).

6. H. M. Wagner, "Principles of Operations Research with Applications to Managerial Decisions," Prentice Hall, Inc., 1969, pp. 559-560.

7. A. Charnes and W. W. Cooper, "Systems Evaluation and Repricing Theorems," Management Science, Vol. 9, No. 1 (1962).

8. M. Klein, "Inspection-Maintenance-Replacement Schedule Under Markovian Deterioration," Management Science, Vol. 9, No. 1, pp. 25-32 (1962).

9. B. Fox, "Markov Renewal Programming by Linear Fractional Programming," SIAM Journal of Applied Mathematics, Vol. 14, pp. 1418-1432 (1966).

10. W. Dinkelbach, "On Nonlinear Fractional Programming," Management Science, Vol. 13, No. 7, pp. 492-498 (1967).

11. C. R. Bector, "Programming Problems with Convex Fractional Functions," Operations Research, Vol. 16, No. 2, pp. 383-391 (1968).

12. O. L. Mangasarian, "Nonlinear Fractional Programming," MRC Technical Summary Report No. 819, Mathematics Research Center, The University of Wisconsin, December, 1967.

13. M. Grunspan, "Fractional Programming: A Survey," Technical Report No. 50, Project THEMIS, Systems Research Center, Industrial and Systems Engineering Department, University of Florida, January, 1971.

14. M. L. Balinski, "Integer Programming: Methods, Uses, Computation," Management Science, Vol. 12, pp. 253-313 (1965).

15. G. B. Dantzig, "On the Significance of Solving Linear Programming Problems with Some Integer Variables," Econometrica, Vol. 28, No. 1, pp. 30-44 (1960).

16. J. R. Isbell and W. H. Marlow, "Attrition Games," Naval Research Logistics Quarterly, Vol. 3, pp. 71-94 (1956).

17. W. S. Dorn, "Linear Fractional Programming," IBM Research Report RC-830 (1962).

18. C. E. Lemke, "The Dual Method of Solving the Linear Programming Problem," Naval Research Logistics Quarterly, Vol. 1, No. 1, pp. 36-47 (1954).

19. A. Charnes and W. W. Cooper, "Programming with Linear Fractional Functionals," Naval Research Logistics Quarterly, Vol. 9, pp. 181-186 (1962).

20. M. Florian and P. Robillard, "Hyperbolic Programming with Bivalent Variables," Department d'Informatique, Universite de Montreal, Publication #41, August, 1970.

21. C. Witzgall, "An All-Integer Programming Algorithm wich Parabolic Constraints," J. SIAM, Vol. 11, No. 4 (1963).

22. J. E. Kelley, Jr., "The Cutting Plane Method for Solving Convex Programs," J. SIAM, Vol. 8, No. 4, pp. 703-712 (1960).

23. C. A. Trauth, Jr. and R. E. Woolsey, "MESA, A Heuristic Integer Linear Programming Technique," SC-RR-68-299 Sandia Laboratories, Albuquerque, July, 1968.

24. G. B. Dantzig, D. R. Fulkerson, and S. M. Johnson, "Solution of a Large Scale Traveling-Salesman Problem,"Operations Research, Vol. 2, No. 4, pp. 393-410 (1954).

25. R. E. Gomory, "An Algorithm for Integer Solutions to Linear Programs," pp. 269-302 in Recent Advances in Mathematical Programming, R. L. Graves and P. Wolfe (Eds.), McGraw-Hill, New York, 1963.

26. _____, "Some Polyhedra Related to Combinatorial Problems," Linear Algebra and Its Applications, Vol. 2, pp. 451-558 (1969).

27. W. W. White, "On a Group Theoretic Approach to Linear Integer Programming," OR 66-27, Operations Research Center, University of California, Berkeley, September, 1966.

28. T. C. Hu, "Integer Programming and Network Flows," Addison Wesley Publishing Company, Inc., 1969.

29. F. Glover, "Integer Programming Over a Finite Additive Group," SIAM Journal of Control, Vol. 7, No. 2, pp. 213-231 (1969).

30. J. F. Shapiro, "Dynamic Programming Algorithms for the Integer Programming Problem-I: The Integer Programming Problem Viewed as a Knapsack Type Problem," Operations Research, Vol. 16, No. 1, pp. 103-121 (1968).

31. G. I. Hefley, "Group Programming Decomposition in Integer Programming," Technical Report No. 52, Project THEMIS, Systems Research Center, Industrial and Systems Engineering Department, University of Florida

32. P. L. Hammer and S. Rudeanu, "Boolean Methods in Operations Research and Related Areas," Springer-Verlag, New York, Inc., 1968.

33. G. H. Bradley, "Transformation of Integer Programs to Knapsack Problems," Report No. 37, Yale University, New Haven, Connecticut (1970).

34. R. E. D. Woolsey and C. A. Trauth, Jr., "IPSC A Machine Independent Integer Linear Program (U)," Sandia Corporation Research Report, SC-RR-66-433, July, 1966.

35. C. A. Trauth, Jr. and R. E. Woolsey, "Integer Linear Programming: A Study in Computational Efficiency," Management Science, Vol. 15, No. 9, pp. 481-493 (1969).

36. J. F. Benders, "Partitioning Procedures for Solving Mixed-Variables Programming Problems," Numerische Mathematik, Vol. 4, pp. 238-252 (1962).