# SURROGATE CONSTRAINT DUALITY AND EXTENSIONS IN INTEGER PROGRAMMING

A THESIS
Presented to
The Faculty of the Division of Graduate Studies
by
Mark Henry Karwan

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in the School of Industrial and Systems Engineering

Georgia Institute of Technology
December, 1976

# SURROGATE CONSTRAINT DUALITY AND EXTENSIONS IN INTEGER PROGRAMMING

Approved:

R. L. Rardin, Chairman

J. J. Jarvis

R. G. Parker

V. E. Unger

W. R. Smythe, Jr.

Date approved by Chairman: 11/15/76

# ACKNOWLEDGMENTS

Many persons have been instrumental in helping and guiding me through my years of graduate studies which have culminated in this dissertation. My initial interest in the field of operations research was kindled at Johns Hopkins University by Dr. Charles ReVelle and Dr. Jack Elzinza. At Georgia Tech, Drs. McGinnis, Graves, and Parker offered invaluable friendship and guidance. Dr. Jarvis provided useful advice and encouragement as teacher and academic advisor. As members of my dissertation committee, Drs. Unger and Smythe gave their time and efforts towards the successful completion of the dissertation. No satisfactory amount of thanks and appreciation could ever be given to my dissertation advisor, Dr. Ron Rardin. The many hours he spent helping to formulate, guide and review my research are incalculable. His guidance, advice, and friendship certainly composed the single largest factor in making my studies at Georgia Tech a rewarding and enjoyable experience.

Finally, I truly owe the successful termination of my graduate studies to my wife Sabina. I wish to thank her for her patience, hard work and understanding over the past three years, and for typing the first draft of this dissertation from a maze of mathematical symbols, inserts and hieroglyphic handwriting.

# TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

# LIST OF TABLES

# LIST OF ILLUSTRATIONS

# SUMMARY

This dissertation investigates a number of important methodological questions arising in the use of the surrogate dual and its extensions in integer programming. Many of these important results are derived from extensions of lagrangian dual results and thus lead to a comparison of the various dual problems. A number of algorithmic procedures are presented for solving the surrogate dual and the most promising of these procedures is efficiently integrated into a primal branch-and-bound procedure.

The proposed surrogate multiplier search procedures are implemented in computerized algorithms and tested on a set of randomly generated 0-1 integer programming test problems. Computational results indicate the viability of employing the surrogate dual for providing bounds in a branch-and-bound procedure.

# CHAPTER I

# INTRODUCTION

The general integer linear programming problem* can be stated as:

$$\text{(P)} \quad \text{Min } cx \qquad \text{subject to } Ax \leq b$$
$$x \in S$$

where

$$S = \{x \geq 0: \; Gx \leq h, \; x \text{ satisfies some discrete constraints}\}$$

Here, as usual, $A$ is an $m \times n$ matrix, $x$, $b$, $c$ are vectors of the appropriate dimension, and $G$ is a $q \times n$ matrix with $h$ a $q \times 1$ vector. The set $S$ is assumed to have some computationally convenient structure not possessed by the entire problem (P). To avoid many pathological cases it will be assumed throughout this dissertation that $S$ is bounded and (P) is feasible.

The <u>surrogate relaxation</u> of the problem (P) associated with any $v \geq 0$ is

---

[*] Many of the results of this investigation can be shown to hold for much more general versions of (P) with nonlinear constraints and objective functions. However, only the above formulation of (P) will be considered because it conforms with much of the theoretical and applied literature in integer programming.

$$(P^v) \quad \text{Min } cx \quad \text{subject to } v(Ax - b) \leq 0$$
$$x \in S$$

Clearly, any x which is feasible for (P) is feasible for $(P^v)$, but the reverse need not be true, i.e. $(P^v)$ is a relaxation of (P) with the single constraint $v(Ax - b) \leq 0$ acting as a surrogate for the system $Ax \leq b$.

Define the function

$\nu(\cdot)$ = the value of an optimal solution to problem $(\cdot)$

Then clearly $\nu(P^v)$ provides a lower bound on $\nu(P)$ for any $v \geq 0$. The best such bound is achieved by the surrogate dual

$$(D_S) \quad \text{Max}_{v \geq 0} \{\nu(P^v)\}$$

The more widely known lagrangian relaxation is:

$$(P_u) \quad \text{Min } cx + u(Ax - b) \text{ subject to } x \in S$$

with the corresponding lagrangian dual

$$(D_L) \quad \text{Max}_{u \geq 0} \{\nu(P_u)\}$$

Again, any x feasible for (P) is feasible for $(P_u)$. If x is feasible for (P), $u \geq 0$ implies that $u(Ax - b) \leq 0$. It follows that $\nu(P_u) \leq \nu(P)$ for any $u \geq 0$, so that $\nu(P_u)$ and $\nu(D_L)$ also provide lower bounds on $\nu(P)$.

By combining the ideas of lagrangian and surrogate relaxation it is also possible to form a composite relaxation:

$$(P^v_u) \quad \text{Min } cx + u(Ax - b) \quad \text{subject to}$$
$$v(Ax - b) \leq 0, \; x \in S$$

with corresponding composite dual

$$(D) \quad \underset{u,v\geq 0}{\text{Max}} \quad \{\nu(P^v_u)\}$$

Finally, one can define a multiple surrogate constraint relaxation associated with $k \geq 2$ nonnegative multipliers $v^1$, $v^2,\ldots,v^k$:

$$(P^{v^1},v^2,\ldots,v^k) \quad \text{Min } cx \quad \text{subject to } x \in S,$$
$$\begin{aligned} v^1(Ax - b) &\leq 0 \\ v^2(Ax - b) &\leq 0 \\ \vdots \quad & \quad \vdots \\ v^k(Ax - b) &\leq 0. \end{aligned}$$

The corresponding multiple surrogate dual is

$$(D_Sk) \quad \underset{v^1,v^2,\ldots,v^k \geq 0}{\text{Max}} \quad \{\nu(P^{v^1},v^2,\ldots,v^k)\}$$

In a like manner, any combination of lagrangian treatment of constraints in the objective function and surrogate combining of constraints can form a valid relaxation from which one can define a corresponding dual problem. However, it will be sufficient for this investigation to restrict attention to the four relaxations defined above.

To date, most research on these relaxation duals in integer programming has centered on the lagrangian dual. The important theoretical issues are documented in Geoffrion (20), and Bazaraa and Goode (5); and applications have been presented by Held and Karp (31), Geoffrion (18), Balas (2), Fisher (13), Fisher, Northrup and Shapiro (14) and Rardin (39). The principal areas of research have included

- Properties of $\nu(P_u)$ as a function of u, including concavity, subdifferentiability, ascent and steepest ascent directions
- The existence and characterization of duality gaps, i.e. differences between $\nu(P)$ and $\nu(D_L)$
- The use of the dual as a lower bound in branch and bound schemes as a means of fathoming branches and deciding on which variable to branch
- The issue of integrality, or when the value of the lagrangian relaxation could just as well be found by dropping all integer requirements in S and solving the lagrangian as a linear program
- Tactics for generating lagrange multipliers which solve $(D_L)$, including general direct search methods based on the use of subgradients and Dantzig-Wolfe decomposition-based outer-linearization schemes

·Identification of classes of integer programs whose special structure permits particularly efficient use of lagrangian relaxation techniques.

Many of the corresponding areas have not been well developed in surrogate duality, and very little research has been directed to studying extensions such as multiple surrogate constraints and the composite dual. The successful applications of lagrangian duality and the fundamental result* that $\nu(P) \geq \nu(D_S k) \geq \nu(D) \geq \nu(D_S) \geq \nu(D_L)$ (i.e. that the lagrangian is actually the least successful for bounding the value of the primal) lead one to suspect that further development of the surrogate dual and its extensions should prove beneficial. This dissertation undertakes such a development.

## Surrogate Duality Literature

The idea of surrogate constraints was first introduced by Glover (24) in 1965 in conjunction with 0-1 integer programming. Glover's definition of the strength of a surrogate constraint leads to the choice of the multiplier, v, which maximizes the value of the surrogate relaxation, $\nu(P^V)$. However, he only shows how to obtain the strongest surrogate constraint when restricting attention to two inequalities. Glover also introduces the idea of using multiple surrogate constraints.

By 1967, Balas (3) and Geoffrion (17) had demonstrated

---

*See Chapter II for a proof.

the practical usefulness of using surrogate constraints in 0-1 integer programming under certain relaxed assumptions. However, the assumptions utilized by these two researchers had the effect of replacing the original integer programming problem by a linear programming problem whose structure was sufficiently simple that the distinction between the surrogate constraint approach and the lagrangian approach vanished.

In 1968 Glover (25) further developed the ideas of strongest surrogate constraints, comparing his definitions with those of Balas (3) and Geoffrion (17). He presented the idea of replacing only a portion of the original problem constraints with a single surrogate constraint, leaving the others to be explicitly enforced, as a scheme for producing stronger surrogate duals. This concept corresponds to the definition in this dissertation with explicitly enforced constraints in the set S.

The first major theoretical treatment of surrogate constraints in mathematical programming is that of Greenberg and Pierskalla (28) in 1970. Greenberg and Pierskalla show that $\nu(P^v)$ is quasi-concave in v. Thus all level sets of the function are convex, but a sequence of "plateaus" or "flat spots" may be encountered so that a local maximum is not necessarily a global one. Greenberg and Pierskalla also show that the surrogate duality gap, $\nu(P) - \nu(D_S)$, is at least as small as the lagrangian duality gap and give an example showing a strictly smaller gap. They also give sufficient conditions for no gap to exist between the surrogate dual and the primal. The composite idea of combining both dual formulations by putting some of the constraints in the objective function and leaving some of them in the surrogate constraint is also first introduced in this paper as a means of further closing any gap which might exist.

The approach by Glover (26) in 1975 was to develop a duality theory which encompassed both surrogate and lagrangian duality, as well as their composite, in a single framework through the concepts of parametric and relative subgradients. Optimality conditions were developed which demonstrated again that the surrogate approach often yields a smaller duality gap than the lagrangian.

The only algorithmic development in surrogate duality has been by Banerjee (4) in 1971, who suggested a Benders (7) type approach for an integer linear program. This algorithm (to be reviewed in Chapter IV) involves iteratively solving an expanding master problem as a linear program and a surrogate relaxation of (P) as a subproblem. Banerjee's work centered on lagrangian duality so that he did not fully develop, test or implement his surrogate concept.

## Purpose of the Dissertation

It is easily seen by the brevity of the surrogate literature that there are a large number of unanswered theoretical and practical questions with regard to surrogate duality and its extensions. In the remainder of this dissertation, the following issues will be addressed:

- Characterization of the composite and multiple surrogate dual functions
- Generalization of the integrality property to the surrogate dual and its extensions
- Theoretical investigation of the occurrence of gaps between the various duals
- Development of general search procedures for optimal surrogate dual multipliers
- Application of surrogate duality in a primal branch-and-bound scheme
- Testing the search procedures and the bound improvement from surrogate duality on a set of randomly generated 0-1 integer linear programming problems

## The Branch-and-Bound Context

Only in rare integer programs would one expect any of the dual problems defined above to directly produce a solution to (P). Thus the importance of the duals in integer programming centers on their ability to produce bounds for a branch-and-bound procedure. By careful partitioning of the constraints of a problem into relaxed ones $Ax \leq b$ and enforced ones $x \varepsilon S$, one can create problems $(P_u)$, $(P^V)$, etc. which are much easier to solve than (P). Thus the bound $\nu(P_u)$ or $\nu(P^V)$ is relatively easy to obtain, and searches over $u \geq 0$ or $v \geq 0$ will produce improved bounds. To establish this branch-and-bound context for the ensuing chapters, a generalized branch-and-bound algorithm and an example of the use of relaxation duality will be briefly outlined here.

Figure 1 presents an outline of a general branch-and-bound procedure. To facilitate the discussion, define P(T) to be the same as (P) except that x is restricted to $x \varepsilon T$.

The set of feasible solutions to (P) can be partitioned into independent subsets by an enumeration which places additional constraints on each integer variable. The unenumerated portion of (P) can be represented by a list of candidate problems, each of which is simply (P) with certain additional constraints $x \varepsilon T$ appended. Each additional constraint stipulates that the value of one of the integer variables must lie in a certain closed interval or take on a certain fixed value. The best currently known feasible solution to (P) is called the incumbent solution with incumbent solution value $\nu^*$. In Step 1, some candidate problem, P(T), which could still produce an optimal solution to (P), is chosen to be explicitly explored. One or more relaxations are then solved in Step 2. If it is determined that P(T) could not yield a feasible solution to (P) which is better than the incumbent solution, then P(T) is fathomed, i.e. eliminated from further consideration in Step 8. If a solution to a relaxation of P(T) is found to be feasible for (P) then a new incumbent is saved at Step 3, and P(T) is fathomed

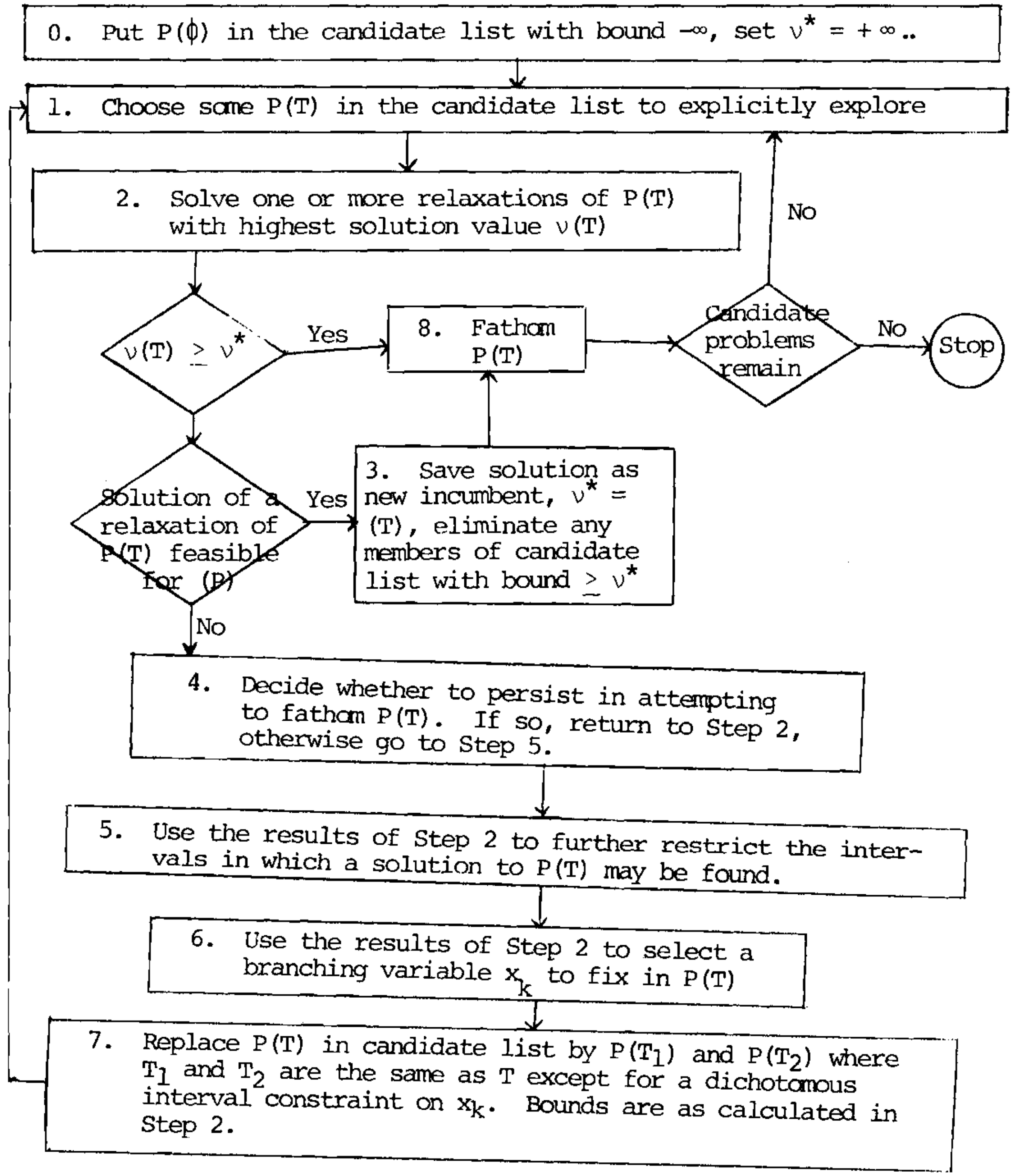

0. Put P(ϕ) in the candidate list with bound −∞, set ν* = +∞..
1. Choose some P(T) in the candidate list to explicitly explore
2. Solve one or more relaxations of P(T) with highest solution value ν(T)
ν(T) ≥ ν*
Yes
8. Fathom P(T)
Candidate problems remain
No
No
Stop
Solution of a relaxation of P(T) feasible for (P)
Yes
3. Save solution as new incumbent, ν* = (T), eliminate any members of candidate list with bound ≥ ν*
No
4. Decide whether to persist in attempting to fathom P(T). If so, return to Step 2, otherwise go to Step 5.
5. Use the results of Step 2 to further restrict the intervals in which a solution to P(T) may be found.
6. Use the results of Step 2 to select a branching variable x_k to fix in P(T)
7. Replace P(T) in candidate list by P(T1) and P(T2) where T1 and T2 are the same as T except for a dichotomous interval constraint on x_k. Bounds are as calculated in Step 2.

Figure 1. Flow Chart of General Branch-and-Bound Approach

since no solution to P(T) can produce a lower cost.

When a condidate problem is examined and not fathomed, the results of Step 2 may be used to further restrict the variable intervals in which a solution to P(T) with $\nu(P(T)) < \nu^*$ might be found. This procedure (Step 5) is termed range restriction. The candidate problem is next separated into two simpler candidate problems. In Step 6, the branching variable is chosen and in Step 7, a dichotomous interval constraint on the branching variable is added to produce the two new candidate problems. Returning to Step 1, the procedure is repeated until no candidate problems remain.

The successful application of duality in a branch-and-bound scheme can be seen to depend on the quality of the bound and the ease of computing the bound, since one must repeat the procedure over and over with different candidate sets. Taking advantage of special problem structure has led to the successful application of the lagrangian dual in Step 2. With the travelling salesman problem, for example, Held and Karp (31) show that with the proper formulation (choice of set S) the lagrangian relaxation can essentially be solved as a minimal spanning tree problem for which there is an extremely fast greedy algorithm. More precisely, Held and Karp realized that an optimal travelling salesman tour is composed of a minimal 1-tree spanning system of links containing exactly one cycle, constrained to have two links or arcs incident to each city or node. To find a 1-tree, one first removes one node from consideration, finds a spanning tree among the remaining nodes, then appends the remaining node to the spanning tree with two links. By choosing the set S to be the collection of minimal 1-trees (say with node 1 removed) and placing the two-links-per-node constraints in the objective function via lagrange multipliers, each lagrangian relaxation is easily solved. After each relaxation solution, a new set of multipliers is obtained which will improve the bound $\nu(P_u)$.

Note that one solves relaxations repeatedly in order to solve any of the dual problems for any candidate problem. Thus a formulation which allows an easy solution to each relaxation is very important. Hansen and Krarup (30) report excellent computational times in solving the lagrangian dual for the travelling salesman problem and the bounds in this formulation have been shown to be very good with small if any duality gap (6). Thus the travelling salesman problem satisfies all of the conditions for a successful application of lagrangian duality in integer programming.

The above discussion illustrates how relaxation duals are actually used to calculate bounds for candidate problems (P(T)). However, the process of obtaining bounds is the same for all T. Thus the T reference will be omitted in all ensuing discussions, and duals will be discussed in terms of the original problem (P).

# CHAPTER II

# FUNDAMENTAL PROPERTIES

## Value Relationships

As discussed in Chapter I, the importance of the surrogate dual and its extensions is based on the ability to provide good bounds on $\nu(P)$. Geoffrion (20), Glover (26), and Greenberg and Pierskalla (28) prove or imply all of the following value relationships:

$$\nu(P) \geq \nu(D_S k) \geq \nu(D) \geq \nu(D_S) \geq \nu(D_L) \geq \nu(\bar{P})$$

Here $(\bar{P})$ is the linear programming relaxation of the problem (P). In this section their proofs are summarized in Lemma 2.1 and Theorem 2.2, and an example is provided which shows strict inequality throughout the above expression.

Lemma 2.1

$$\nu(P) \geq \nu(P^{u,v}) \geq \nu(P_u^v) \text{ for all } u, v \geq 0 \text{ and}$$

$\nu(P^v) \geq \nu(P_v)$ for all $v \geq 0$.

Proof:

$$\nu(P) \overset{\text{(less constrained)}}{\geq} \nu\begin{pmatrix}\text{Min } cx & & \\ \text{s.t} & v(Ax - b) \leq 0 & \\ & u(Ax - b) \leq 0, & x \in S\end{pmatrix}$$

$$= \nu(P^{u,v}) \overset{\begin{pmatrix}\text{adding nonpositive}\\ \text{to objective function}\end{pmatrix}}{\geq} \nu\begin{pmatrix} \text{Min } cx + u(Ax-b) & \\ \text{s.t. } \; v(Ax-b) \leq 0 & \\ \qquad\quad u(Ax-b) \leq 0, & x \varepsilon S \end{pmatrix}$$

$$\overset{\text{(less constrained)}}{\geq} \nu\begin{pmatrix} \text{Min } cx + u(Ax-b) & \\ \text{s.t. } \; v(Ax-b) \leq 0, & x \varepsilon S \end{pmatrix} = \nu(P_u^v).$$

The second portion of the Lemma follows from the first by choosing $u = 0$.

Q.E.D.

Theorem 2.2

$$\nu(\bar{P}) \overset{(i)}{\leq} \nu(D_L) \overset{(ii)}{\leq} \nu(D_S) \overset{(iii)}{\leq} \nu(D) \overset{(iv)}{\leq} \nu(D_S k) \overset{(v)}{\leq} \nu(P)$$

Proof:

(i) A well known result of linear programming is that the value of a linear program is equal to the value of its lagrangian dual. Thus, $\nu(\bar{P}) = \underset{u \geq 0}{\text{Max}}\ \nu(\bar{P}_u) \overset{\text{(more constrained)}}{\leq} \underset{u \geq 0}{\text{Max}}\ \nu(P_u) = \nu(D_L)$.

(ii) Let $u^*$ be an optimal dual multiplier for $(D_L)$. $\nu(D_L) = \nu(P_u{}^*) \overset{\text{(Lemma 2.1)}}{\leq} \nu(P^{u^*}) \leq \underset{v \geq 0}{\text{Max}}\ \nu(P^v) = \nu(D_S)$.

(iii) Let $v^*$ be an optimal surrogate multiplier for $(D_S)$. $\nu(D_S) = \nu(P^{v^*}) \leq \underset{u \geq 0}{\text{Max}}\ \nu(P_u^{v^*}) \leq \underset{u,v \geq 0}{\text{Max}}\ \nu(P_u^v) = \nu(D)$.

(iv) Now let $u^*$, $v^*$ solve (D).

$$\nu(D) = \nu(P^{v^*}_{u^*}) \overset{\text{(Lemma 2.1)}}{\leq} \nu(P^{u^*,v^*}) \leq \underset{u,v \geq 0}{\text{Max}} \; \nu(P^{u,v}) =$$

$$\nu(D_S 2) \leq \nu(D_S k), \; k \geq 2.$$

$$\text{(v)} \quad \nu(D_S k) = \underset{v^1, v^2, \ldots, v^k \geq 0}{\text{Max}} \; \nu(P^{v^1, \ldots, v^k})$$

$$\overset{\text{(more constrained)}}{\leq} \nu(P).$$

Q.E.D.

Figure 2 presents an example which shows that it is possible for $\nu(P) > \nu(D_S 2) > \nu(D) > \nu(D_S) > \nu(D_L) > \nu(\bar{P})$; that is, strict improvement is obtained each time a 'more difficult' relaxation is considered, and in each case there is a duality gap to the primal.

The problem considered is

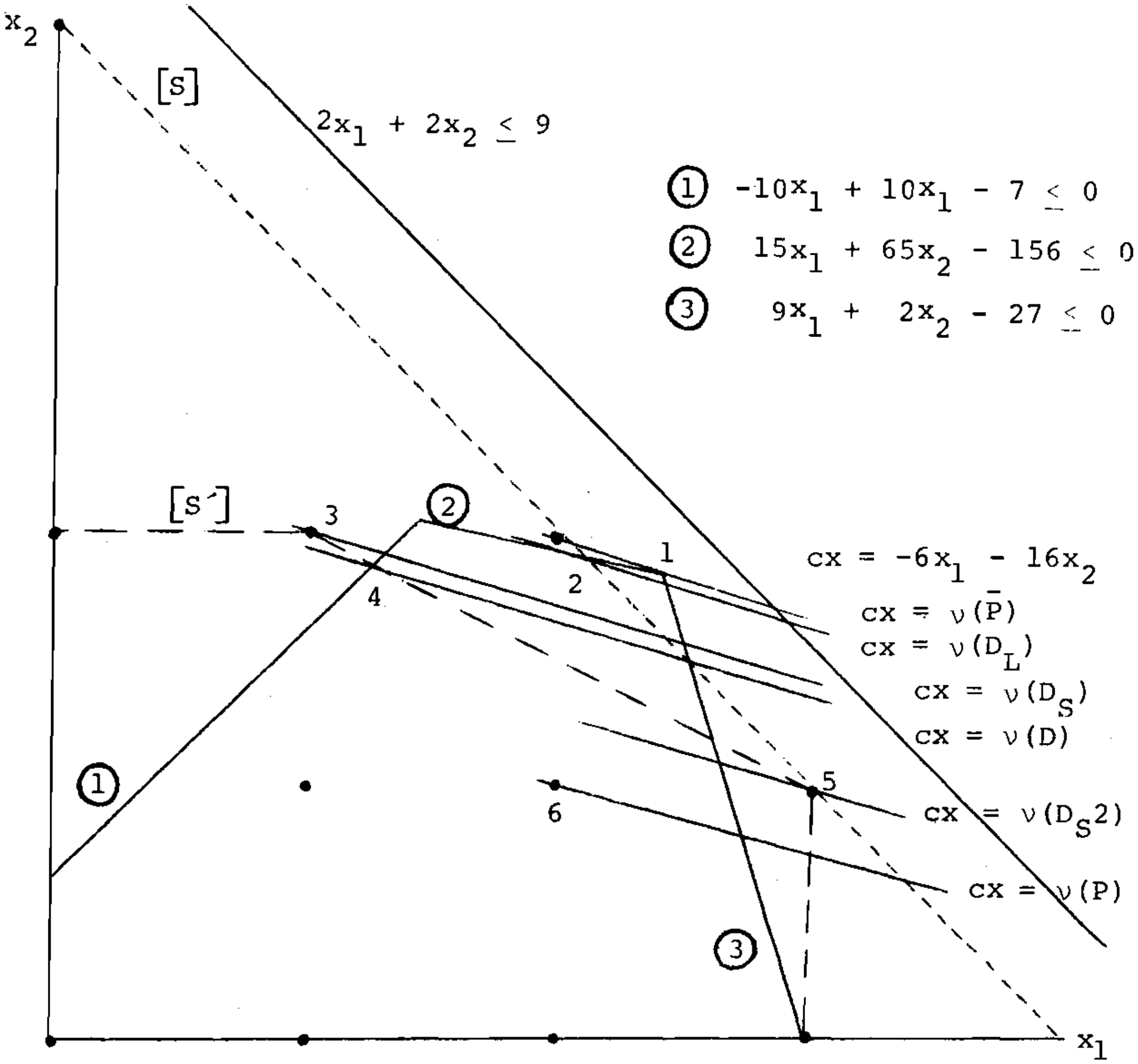
$$\begin{aligned} \text{Minimize} \quad & -6x_1 - 16x_2 \\ \text{s.t} \quad & -10x_1 + 10x_2 \leq 7 \\ & 15x_1 + 65x_2 \leq 156 \\ & 9x_1 + 2x_2 \leq 27 \end{aligned}$$

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \varepsilon \; S = \{\text{integer} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} : 2x_1 + 2x_2 \leq 9\}$$

Referring to Figure 2, it is clear that the optimal solution to the linear programming relaxation $(\bar{P})$ occurs at the point labeled 1. This point corresponds to $(x_1,x_2) = (2\ 3/5,\ 1\ 4/5)$ implying $\nu(\bar{P}) = -44.4$. Geoffrion (20) shows that the optimal value of the lagrangian dual, $\nu(D_L)$, can be obtained by solving the linear program

$$\begin{aligned} &\text{Min } cx \\ &\text{s.t. } \ x \varepsilon \{Ax - b \leq 0 \cap [S]\} \end{aligned}$$

where $[S]$ is the convex hull of S. This solution occurs at point 2 with $(x_1,x_2) = (2\ 2/25,\ 1\ 23/25)$ and $\nu(D_L) = -43.2$. The value of the surrogate dual, $\nu(D_S)$, can be found by trying various combinations of constraints 1, 2, and 3. One optimal surrogate constraint is constraint 2 itself, which 'cuts off' the superoptimal points $(x_1,x_2) = (0,4)$, $(0,3)$, $(1,3)$ and $(2,2)$. The optimal surrogate solution then occurs at $(x_1,x_2) = (1,2)$ with $\nu(D_S) = -38$. The optimal value of the composite dual may be found by trying various surrogate constraints and solving the corresponding linear program over the constraint set $\{Ax - b \leq 0 \cap [S']\}$ where $S' = \{S \cap v(Ax - b) \leq 0\}$. An optimal choice is again constraint 2 by itself. That surrogate constraint results in $[S']$ as shown in Figure 2. The optimal value of the composite dual occurs at point 4, $(x_1,x_2) = (1\ 1/5,\ 1\ 9/10)$ and $\nu(D) = -37.6$. The optimal solution to the multiple surrogate, $(D_S 2)$, can be found by trying all pairs of surrogate constraints. It



$S = \{x : 2x_1 + 2x_2 \le 9,\ x \text{ integer}\}$

Figure 2. Example of Duality Gaps

soon becomes obvious that no two surrogate constraints can cut off all three of the points $(x_1,x_2) = (1,2)$, $(2,2)$ and $(3,1)$. Cutting off points $(1,2)$ and $(2,2)$ leaves $(x_1,x_2) = (3,1)$ with $\nu(D_S2) = -34$. The optimal primal solution is clearly at $(x_1,x_2) = (2,1)$ with $\nu(P) = -28$. Thus the example demonstrates that each successive relaxation results in an improved bound, but none avoids a duality gap.

## Searchability of the Dual Function

The value of the optimal solution to various dual relaxations can be viewed as functions of their dual multipliers. That is, $\nu(P_u)$, $\nu(P^v)$, $\nu(P_u^v)$, and $\nu(P^{u,v})$ are functions of u, v, and (u,v). The characteristics of these functions may determine whether there exist good search procedures to find optimal or near optimal dual multipliers for each relaxation. It is well known (see for example Lasdon (34)) that the value of the lagrangian relaxation, $\nu(P_u)$, is a piecewise linear concave function of u. This property has led to a number of successful search procedures for optimal lagrange multipliers (for a good review see Bazaraa and Goode (5)). Greenberg and Pierskalla (28) prove that the value of the surrogate relaxation, $\nu(P^v)$, is a quasi-concave function of v. This property leads to the development of search procedures for optimal surrogate multipliers in Chapter IV. Before investigating similar results for the composite and multiple surrogate constraint relaxations, proofs of the above two properties are presented for completeness.

Theorem 2.3

$\nu(P_u)$ is a concave function of u.

Proof:

Let $u^1 \geq 0$, $u^2 \geq 0$, and $\bar{x}$ solve $(P_{\bar{u}})$ where $\bar{u} = \lambda u^1 + (1 - \lambda)u^2$ for some $0 \leq \lambda \leq 1$. Then $\nu(P_{\bar{u}}) =$ $c\bar{x} + (\lambda u^1 + (1 - \lambda)u^2)(A\bar{x} - b) =$ $\lambda\left[c\bar{x} + u^1 (A\bar{x} - b)\right] + (1 - \lambda) \left[c\bar{x} + u^2 (A\bar{x} - b)\right] \geq$ $\lambda\nu(P_u 1) + (1 - \lambda) \nu(P_u 2)$ because $\bar{x}$ is just one feasible solution to $(P_u 1)$ and $(P_u 2)$.

Q.E.D.

Theorem 2.4

$\nu(P^v)$ is a quasi-concave function of v.

Proof:

Let $v^1 \geq 0$, $v^2 \geq 0$, $x^1$ solve $(P^{v^1})$, $x^2$ solve $(P^{v^2})$ and $\bar{x}$ solve $(P^{\bar{v}})$, where $\bar{v} = \lambda v^1 + (1 - \lambda)v^2$ for some $\lambda$, $0 \leq \lambda \leq 1$. $\nu(P^v)$ is a quasi-concave function of v if $\nu(P^{\bar{v}}) \geq \text{Min}\ \{\nu(P^{v^1}), \nu(P^{v^2})\}$ for all $\lambda$ such that $0 \leq \lambda \leq 1$. Suppose $\nu(P^{\bar{v}}) < \nu(P^{v^1})$ and $\nu(P^{v^2})$. In this case $\bar{x}$ must be infeasible in both $(P^{v^1})$ and $(P^{v^2})$, and $v^1(A\bar{x} - b) > 0$, $v^2(A\bar{x} - b) > 0$. But then $(\lambda v^1 + (1 - \lambda)v^2)(A\bar{x} - b) =$ $\bar{v}\ (A\bar{x} - b) > 0$ which is a contradiction because $\bar{x}$ is feasible for $(P^{\bar{v}})$.

Q.E.D.

The composite and multiple surrogate duals are suggested by Greenberg and Pierskalla (28) and Glover (26) as a means of helping to further close any lagrangian or surrogate duality gap. Again, an important question is whether there exist good search procedures to find optimal or near optimal multipliers for these two dual problems.

Quasi-concavity is almost a minimal requirement for convenient searching of a surface. Unfortunately the next two examples will show that $\nu(P^{u,v})$ and $\nu(P_u^v)$ lack this property as functions of (u,v). A quasi-concave function may possess "flat spots" which imply local maxima, but a monotone nondecreasing path (monotone increasing if one could ignore the "flat spots") may be found in maximizing the function. A function which lacks quasiconcavity may have much 'stronger' or 'strict' local maxima in the sense that a strict decrease in the function value must precede any increase. These results make impractical simple direction search procedures for optimal composite or multiple surrogate multipliers.

Turning first to the composite dual, consider the problem

$$\begin{array}{ll} \text{Min} & -4x_1 - 5x_2 \\ \text{s.t.} & x_1 + 5x_2 - 4 \le 0 \\ & 5x_1 + x_2 - 4 \le 0 \\ & x \in S \end{array}$$

where $S = \{x:\ 0 \le x \le 2,\ x \text{ integer}\}$.

Let $u^1 = (0,0)$, $v^1 = (1,1)$, $u^2 = (4/5,\ 0)$ and $v^2 = (0,2)$. If $\nu(P^{v}_{u})$ is a quasi-concave function of $(u,v)$ then $\nu(P^{\bar{v}}_{\bar{u}}) \geq$ Min $\{\nu(P^{v^1}_{u^1}),\ \nu(P^{v^2}_{u^2})\}$ where $(\bar{u},\ \bar{v}) = \lambda(u^1,v^1) + (1-\lambda)(u^2,v^2)$, for any $0 \leq \lambda \leq 1$.

$$\nu(P^{v^1}_{u^1}) = \nu\begin{pmatrix} \text{Min} & -4x_1 - 5x_2 \\ \text{s.t.} & 6x_1 + 6x_2 - 8 \leq 0 \end{pmatrix} = -5 \text{ with solution}$$
$$x \in S$$

$(x_1,x_2) = (0,1)$.

$$\nu(P^{v^2}_{u^2}) = \nu\begin{pmatrix} \text{Min} & -\frac{16}{5}x_1 - x_2 - \frac{16}{5} \\ \text{s.t.} & 10x_1 + 2x_2 - 8 \leq 0 \end{pmatrix} = -5\ 1/5 \text{ with solution}$$
$$x \in S$$

$(x_1,x_2) = (0,2)$.

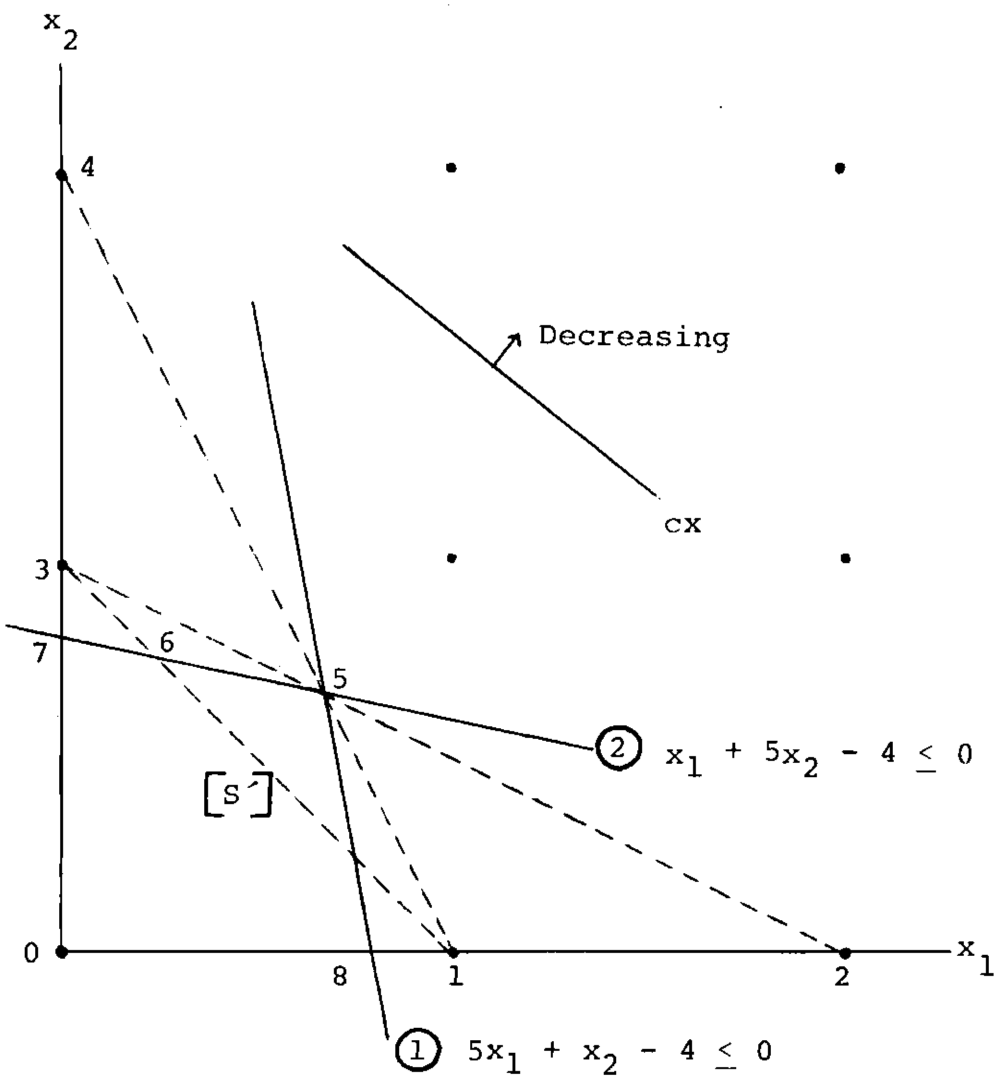
Let $\lambda = \frac{1}{2}$, then $\bar{u} = (\frac{2}{5},\ 0)$ and $\bar{v} = (\frac{1}{2},\ \frac{3}{2})$.

$$\nu(P^{\bar{v}}_{\bar{u}}) = \nu\begin{pmatrix} \text{Min} & -\frac{18}{5}x_1 - 3x_2 - \frac{8}{5} \\ \text{s.t.} & 8x_1 + 4x_2 - 8 \leq 0 \end{pmatrix} = -7\ 3/5 \text{ with solution}$$
$$x \in S$$

$(x_1,x_2) = (0,2)$. $\nu(P^{\bar{v}}_{\bar{u}}) \not\geq$ Min $\{\nu(P^{v^1}_{u^1}),\ \nu(P^{v^2}_{u^2})\}$ so that $\nu(P^{v}_{u})$ is not a quasi-concave function of $(u,v)$.

The above example may also be used to show that $\nu(P_u^v)$ may have 'strict' local maxima. Consider Figure 3 which is a graphical representation of the above example. Choose $v^1 = (1,1)$ and find the corresponding $u^*$ which maximizes $\nu(P_u^{v^1})$. Since $v^1$ only allows the points 0, 1 and 3 to be feasible (following the analysis given for Figure 2) the optimal solution value for $\underset{u \geq 0}{\text{Max}}\ \nu(P_u^{v^1})$ occurs at the point labeled 6 in Figure 3. Thus $u_1^* = 0$, $u_2^* > 0$, and $\nu(P_{u^*}^{v^1}) = cx^1 + u^*(Ax^1 - b) = cx^3 + u^*(Ax^3 - b)$. To improve on this value it is necessary to have v make either points 1 and 2 or points 3 and 4 infeasible, with corresponding optimal choices for u giving solution values at the points 7 and 8 respectively. In rotating $v^1$ to accomplish this task and to keep $\nu(P_u^{v^1})$ from decreasing, u must remain fixed at $u^*$ as long as both points 1 and 3 are feasible. In attempting to cut off points 1 and 2, point 4 must first be made feasible. However this implies a decrease in $\nu(P_{u^*}^v)$ since the best choice of u (if both 1 and 4 are feasible) gives a solution value occurring at point 5. A similar result is found in attempting to cut off both points 3 and 4. So a strict local maximum is found at $(u^*, v^1)$.

The next example shows that the two-surrogate function, $\nu(P^{u,v})$, is also not a quasi-concave function of (u,v). Consider the problem



$S = \{x : 0 \le x_1,\ x_2 \le 2,\ x \text{ integer}\}$

Figure 3. Example of Strict Local Maxima in $\nu(P_u^V)$

$$\begin{aligned} \text{Min} \quad & -x_1 - x_2 \\ \text{s.t.} \quad & 2x_1 + x_2 - 2 \leq 0 : (Ax - b)_1 \\ & -x_1 + x_2 - 1 \leq 0 : (Ax - b)_2 \\ & x \varepsilon S \end{aligned}$$

where $S = \{x: \ 0 \leq x \leq 1, \ x \text{ integer}\}$.

Let $u_1 = (1,0)$, $v_1 = (0,2)$, $u_2 = (0,2)$, $v_2 = (1,0)$. Then $\hat{x} = (1,1)$ is infeasible for both $(P^{u^1,v^1})$ and $(P^{u^2,v^2})$ since:

$$(A\hat{x} - b)_1 = 1, \ (A\hat{x} - b)_2 = -1$$

$$u^1(A\hat{x} - b) = 1, \ v^1(A\hat{x} - b) = -2$$

$$u^2(A\hat{x} - b) = -2, \ v^2(A\hat{x} - b) = 1$$

Clearly $\nu(P^{u^1,v^1})$ and $\nu(P^{u^2,v^2})$ are both $> c\hat{x} = -2$ since $\hat{x}$ is infeasible to both problems and no other $x \varepsilon S$ has a value less than -1.

Letting $\lambda = \frac{1}{2}$ gives $\bar{u} = \bar{v} = (\frac{1}{2},1)$ and $\bar{u}(A\hat{x} - b) = -\frac{1}{2}$. Therefore $\hat{x}$ is feasible for $(P^{\bar{u},\bar{v}})$ and $\nu(P^{\bar{u},\bar{v}}) = -2 \ngeq \text{Min}\ \{\nu(P^{u^1,v^1}), \ \nu(P^{u^2,v^2})\}$.

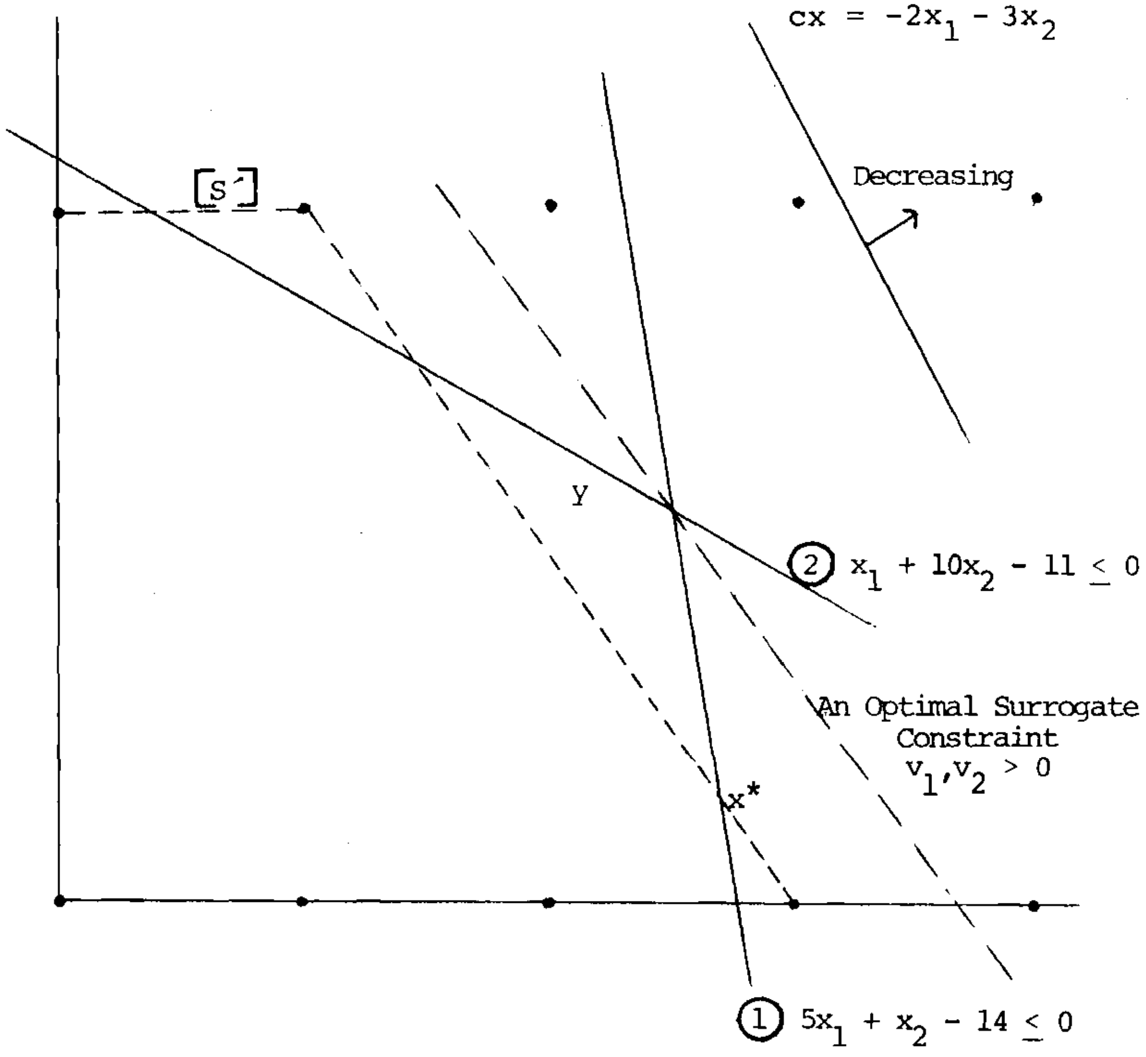
The above examples apparently preclude simple searches for optimal dual multipliers in (D) and $(D_Sk)$. Glover (26) suggests that it may be appropriate to proceed by keeping $u \cdot v = 0$, that is, not putting more than one type of multiplier on each constraint. The example below, however, shows that this restriction may prevent finding the optimal solution for the composite dual. Consider the problem

$$\begin{aligned} \text{Min} \quad & -3x_1 - 2x_2 \\ \text{s.t.} \quad & 5x_1 + x_2 - 14 \leq 0 \\ & x_1 + 10x_2 - 11 \leq 0 \\ & x \in S \end{aligned}$$

where $S = \{x\colon 0 \leq x_1 \leq 4,\ 0 \leq x_2 \leq 1,\ x \text{ integer}\}$.

As in the example of Figure 2, a solution is obtained by trying various surrogate constraints, determining the resulting convex hull of $S'$, where $S' = \{S \cap v(Ax - b) \leq 0\}$ and finding the solution value to the corresponding linear program. Figure 4 shows an optimal surrogate constraint for (D), the resulting convex hull of $S'$, and the point $x^*$ which determines the optimal solution value for (D). Note that $v_1 > 0$ in any optimal surrogate constraint.

As will be shown in Chapter 3, the optimal lagrange multipliers u will be the dual multipliers for the problem $(P_u)$ with S replaced by $[S']$. Since constraint 1 is binding at the optimal solution, $u_1 > 0$. Thus both $u_1 > 0$ and $v_1 > 0$, so that the optimal solution is not complementary. Similar examples may be used to show that complementary u and v do not lead to optimal solutions even in the case of block



$cx = -2x_1 - 3x_2$
Decreasing
[S']
y
x*
② $x_1 + 10x_2 - 11 \le 0$
An Optimal Surrogate Constraint
$v_1, v_2 > 0$
① $5x_1 + x_2 - 14 \le 0$

Figure 4. Example of Optimal (u,v) not complementary

diagonal structure of the constraint matrix A.

The above example can also be used to show the futility of a search procedure for (D) which finds an optimal u for a given v, then tries to find a new v, iteratively searching for lagrangian and surrogate multipliers. The lack of quasi-concavity leads to a local optimal solution. Consider Figure 4 with an initial choice of v which cuts off, i.e. makes infeasible, only the points x = (4,0), (3,1) and (4,1). Then the optimal choice of u is u = (0, 1/9) with solution value -6 8/9 as determined by the point labelled y. Keeping u = (0, 1/9) and trying to find a better v terminates in failure. The points x = (3,0) and (2,1) cannot both be made infeasible by any choice of v, yet cx + u(Ax - b) = -6 8/9 for both of these points when u = (0, 1/9). As shown earlier, the optimal solution value occurs at the point $x^*$ and is clearly greater than -6 8/9.

The above counter-examples seem to preclude any successful search procedures for optimal dual multipliers in the composite and multiple surrogate duals. Thus search procedures for those duals are not further pursued in this dissertation. However, multiplier search procedures for the surrogate dual $(D_S)$ are fully explored in Chapter IV.

# CHAPTER III

## GAPS AND INTEGRALITY PROPERTIES

In the previous chapter it was shown that each dual relaxation can provide a strictly better bound on the primal problem than the preceding "simpler" relaxation; i.e., there are gaps between the duals. Duality gaps are often considered in research on mathematical programming. However, with the exception of some important work by Geoffrion (20), interest in the past has centered on conditions under which there are no primal-to-dual gaps. Since primal-to-dual gaps are almost always present in integer programming, such conditions are of little practical interest. The more important question for integer programmers is, "Under what conditions will particular duals provide a strictly better bound than other, 'simpler' duals?" The chief intent of this chapter is to investigate conditions characterizing when strict differences do or do not result as one moves from dual to dual. If no improvement is to be gained, then it would be best to work with the "simpler" relaxation, since each potentially stronger relaxation is more difficult to solve.

In the following sections, each dual relaxation will be discussed with respect to primal-dual gaps, and the gaps between the dual in question and simpler duals. In order to conveniently present these ideas, the following notation will be employed where necessary. Let $P(R)$ be the version of $P$ with the constraint $x \varepsilon S$ replaced by $x \varepsilon R$. Corresponding relaxation and dual problems are $P_u(R)$, $P^v(R)$, $D_L(R)$, and $D_S(R)$. Let $\left[R\right]$ represent the convex hull of the set $R$. Also let a bar over the name of any constraint set represent the same set with all integrality requirements relaxed. Thus $P(\bar{S})$ is the linear programming relaxation of $P$. Finally, define the function $\Omega(\cdot)$ to represent the set of optimal solutions to the problem $(\cdot)$.

## Lagrangian Dual

As noted in Chapter I, the integer programming dual which has received greatest attention in the literature is the lagrangian dual. Many important questions have been researched and some results are well known. However, for completeness and motivation of later sections, the lagrangian results will be reviewed in this section.

### Primal-Dual Gap

Although the emphasis in this chapter is on gaps between the dual relaxations themselves, it is instructive to review the primal-dual gap characteristics which may give some insight into later results. In the lagrangian case, the well known primal-dual gap property is as follows:

Theorem 3.1

Let $u \varepsilon \Omega(D_L)$ and let $x \varepsilon \Omega(P_u)$. Then $\nu(D_L) = \nu(P)$

and $x \varepsilon \Omega(P)$ if and only if $Ax - b \leq 0$ and $u(Ax - b) = 0$.

Proof:

See, for example, Lasdon (34).

Q.E.D.

Observe that for no primal-dual gap to exist a solution to the lagrangian dual must be both primal feasible and satisfy complementarity; i.e., $u(Ax - b) = 0$. Since most integer programming problems of interest do not satisfy these conditions, one would normally expect a primal-to-dual duality gap.

Integrality Property

A key element of Geoffrion's (20) development of lagrangian duality in integer programming is the idea of an integrality property in the lagrangian relaxation. Geoffrion shows that some lagrangian formulations of P possessing this property can just as well be replaced by $P(\bar{S})$, i.e., the additional complexity of obtaining a bound from the lagrangian dual leads to no improvement over the standard bound from the linear programming relaxation of (P). Formally,

Definition

The problem $(P_u)$ has the integrality property if $\nu(P_u(S)) = \nu(P_u(\bar{S}))$ for all $u \geq 0$, i.e., if the lagrangian relaxation can be solved as a linear program for all u.

Letting $\bar{u}$ denote the optimal dual multiplier vector for the ordinary linear program $P(\bar{S})$, one can express Geoffrion's key results as follows:

Theorem 3.2

(i) $\nu(P(\bar{S})) = \nu(D_L(\bar{S})) \leq \nu(P_{\bar{u}}(S)) \leq \nu(D_L(S)) = \nu(D_L([S])) = \nu(P([S]))$.

(ii) If the problem $(P_u)$ has the integrality property, then $\nu(P(\bar{S})) = \nu(P_{\bar{u}}(S)) = \nu(D_L(S))$.

Proof:

See Geoffrion (20).

Theorem 3.2 has a number of important implications for developing lagrangian strategies for integer programming problems:

(a) $\nu(P(\bar{S})) = \nu(D_L(\bar{S}))$ implies that relaxing the constraint set S when solving the lagrangian dual precludes any potential bound improvement over the bound from the linear programming relaxation $(P(\bar{S}))$.

(b) $\nu(D_L(\bar{S})) \leq \nu(P_{\bar{u}}(S)) \leq \nu(D_L(S))$ implies that the optimal dual multiplier vector for $P(\bar{S})$, although not necessarily an optimal lagrange multiplier for $P_u(S)$, provides at least as good and possibly a better bound than $\nu(P(\bar{S}))$.

(c) The set S can be replaced by the convex hull of S, $[S]$, and no change in lagrangian dual solution value will occur.

(d) $\nu(D_L(S)) = \nu(P([S]))$, i.e., the value of the lagrangian dual may be obtained by solving the linear program formed when S is replaced in (P) by $[S]$. (This result was used in Chapter II to solve graphically for $\nu(D_L)$.)

Most important, part (ii) of Theorem 3.2 indicates that the lagrangian dual does not provide increased bounding power over the linear programming relaxation when the lagrangian relaxation has the integrality property. The significance of this result can be shown by considering the examples in the next section of natural problem formulations having the integrality property.

## Integrality Property Examples

The integrality property for the lagrangian relaxation arises in many of the most natural formulations of the primal problem. The examples listed below exemplify this point by illustrating the potential weakness* of the lagrangian dual. The first two examples are taken from Geoffrion (20).

Example 1. Consider the constraint S to consist of only integer lower and upper bounds, $L_j$ and $U_j$, on some or all of the variables. Clearly the integrality property is present in this case since $\nu(P_u(S)) = \min_{x \varepsilon S} cx + u(Ax - b)$ has a solution obtained by choosing $x_j = L_j$ if $(c + uA)_j > 0$ and $x_j = U_j$ if $(c + uA)_j \leq 0$. The same solution is optimal when S is relaxed to constraints $L_j \leq x_j \leq U_j$, $x_j$ continuous.

Example 2. Consider a case where the constraint set S is like that of Example 1 with $L_j = 0$, but it also includes

---

*Of course, successful applications of lagrangian duality have been exhibited (see for example (13), (14), (18), (31)) but the formulations used were quite innovative.

some generalized upper bound constraints of the form $\sum_{j\varepsilon J_k} x_j \leq 1$, $k = 1, 2, \ldots, K$, where $J_1, \ldots, J_K$ are disjoint subsets of the variable index set I. Such constraints perform a "multiple choice function," for example in the case of mutually exclusive projects in a capital budgeting problem. The identical optimal solutions to $P_u(S)$ or $P_u(\bar{S})$ can again be determined by inspection. It is only necessary to search for the smallest $(c - uA)_j$ coefficient in each subset $J_k$.

Example 3. The integrality property is also present in any problem in which the constraint set S is unimodular. In this case, replacing S by $\bar{S}$ does not change the solution to $P_u(S)$ since $\bar{S}$ has all integer extreme points. An example in which a logical lagrangian formulation would yield a set of constraints S which are unimodular is the integer multicommodity minimum cost flow problem (MCMC). Consider the following formulation.

$$\text{(MCMC)} \qquad \text{Min} \quad \sum_{k=1}^{r} \sum_{(i,j)\varepsilon A} c_{ij}^k \, f_{ij}^k$$

$$\text{s.t.} \quad \sum_{j\varepsilon N} f_{ij}^k - \sum_{j\varepsilon N} f_{ji}^k = \begin{cases} v^k & i = s^k \\ 0 & i \neq s^k,\ t^k \\ -v^k & i = t^k \end{cases}$$

$$\sum_{k=1}^{r} f_{ij}^{k} \leq u_{ij} \qquad \text{for all } (i,j) \varepsilon A$$

$$f_{ij}^{k} \geq 0 \qquad \text{for all } (i,j) \varepsilon A \text{ and all } k$$

$$f_{ij}^{k} \text{ integer} \qquad \text{for all } (i,j) \varepsilon A \text{ and all } k$$

where N is the set of nodes of the network

A is the set of arcs of the network

$f_{ij}^{k}$ is the flow of commodity k from node i to node j

$v^k$ is the total required flow of commodity k from $s^k$ to $t^k$

$u_{ij}$ is the capacity of arc (i,j)

$s^k$ and $t^k$ are the source and sink, respectively, for commodity k.

Placing the capacity constraints, $\sum_{k=1}^{r} f_{ij}^{k} \leq u_{ij}$ for all $(i,j) \varepsilon A$, in the objective function via a lagrange multiplier produces a lagrangian relaxation which separates into r single commodity flows. It is well known (see Ford and Fulkerson (15)) that the single commodity flow problem is unimodular. Thus, the integrality constraint is superfluous in the lagrangian relaxations, and the integrality property holds.

Example 4. A number of nonlinear programming problems can be arbitrarily closely approximated using integer linear programming formulations. One such example is the following integer programming problem with separable concave cost function (IPC).

$$\text{(IPC)} \qquad \text{Min} \quad \sum_j \gamma_j(y)$$

$$\text{s.t.} \quad Ay \leq b$$

$$0 \leq y \leq d, \ y \text{ integer}$$

where $\gamma_j(y)$ is a piecewise-linear concave function of $y_j$ only. With the proper change of variables, one can formulate an integer linear programming problem with the same set of solutions. Let

$$y_j = \sum_k x_j^k$$

$$x_j^k \text{ has cost } = w_j^k f_j^k + v_j^k x_j^k$$

$$S = \{x: 0 \leq x_j^k \leq d_k w_j^k, \ \sum_j w_j^k \leq 1, \ w_j^k \geq 0, \ w_j^k \text{ and } x_j^k \text{ integer}\}$$

Then writing the expression for y simply as y = g(x) yields the fixed charge formulation

$$\begin{aligned} \text{Min} \quad & \sum_j \sum_k (v_j^k x_j^k + f_j^k w_j^k) \\ \text{s.t.} \quad & Ag(x) - b \leq 0 \\ & x \in S \end{aligned}$$

The lagrangian relaxation is

$$\begin{aligned} P_u(S) : \text{Min} \quad & \sum_j \sum_k (v_j^k x_j^k + f_j^k w_j^k) + u(Ag(x) - b) \\ \text{s.t.} \quad & x \in S \end{aligned}$$

The set S is similar to the "multiple choice" constraints of Example 2. Any solution to $P_u(S)$ has $x_j^k = 0$ or $d_k$, and is the same if S is replaced by $\bar{S}$. Therefore, this formulation also has the integrality property.

## Surrogate Dual

The type of results that have been presented for the lagrangian dual may also be developed for the surrogate dual and its extensions. After reviewing the conditions for no primal-dual gap, the integrality property is generalized to the surrogate case and shown to be much less an issue compared to the lagrangian case. Conditions for gaps between the lagrangian and surrogate duals are developed which indicate expected improvement in $\nu(D_S)$ over $\nu(D_L)$ in most problems of interest. These gap conditions also lead to a two phase, lagrangian-surrogate algorithm.

## Primal-Dual Gap

Greenberg and Pierskalla (28) and Glover (26) give conditions under which $\nu(P) = \nu(D_S)$. The following theorem and proof paraphrase their result.

### Theorem 3.3

Let $v \varepsilon \Omega(D_S)$. Then $\nu(P) = \nu(D_S)$ if and only if there exists an $x \varepsilon \Omega(P^V)$ such that $Ax - b \leq 0$.

### Proof:

Any x which is feasible for (P) is also feasible for $(P^V)$, for all $v \geq 0$. Therefore if $\nu(P) = \nu(D_S)$ and $v \varepsilon \Omega(D_S)$, then any x optimal for (P) is optimal for $(P^V)$ and $cx = \nu(P) = \nu(D_S)$.

Conversely, if $Ax - b \leq 0$ for some $x \varepsilon \Omega(P^V)$, then x is feasible for (P). Since the primal is in terms of minimization and in general $\nu(D_S) \leq \nu(P)$, x is an optimal solution for (P) and $\nu(D_S) = \nu(P)$.

Q.E.D.

Note that in comparison with the lagrangian case, the surrogate case does not require complementarity; i.e., $v(Ax - b) = 0$. Thus in some sense the surrogate dual is less restrictive. This important difference will be referred to again when discussing lagrangian-surrogate gaps.

## Integrality Property

In the surrogate case one can also define an integrality property, i.e., $\nu(P^V(S)) = \nu(P^V(\bar{S}))$ for all $v \geq 0$. But it shall be seen that the bounding weakness in the lagrangian cases does not generally hold for surrogate formulations.

The role of the integrality property in the surrogate case revolves around the relationship between $\nu(D_L(S))$ and $\nu(D_S(S))$. The following theorem summarizes these results.

Theorem 3.4

(i) $\nu(D_L(S)) = \nu(D_S([S]))$

(ii) $\nu(P(\bar{S})) = \nu(D_L(\bar{S})) = \nu(D_S(\bar{S}))$

(iii) If $P^V(S)$ has the integrality property then $\nu(P(S)) = \nu(P_{\bar{u}}(S)) = \nu(D_L(S)) = \nu(D_S(S))$.

Proof:

$$\text{(i)} \quad \nu(D_S([S])) = \nu\left(\begin{array}{lll} \underset{v\geq 0}{\text{Max}} & \text{Min} & cx \\ & \text{s.t.} & v(Ax - b) \leq 0 \\ & & x \in [S] \end{array}\right)$$

The inner minimization problem is convex so it equals its lagrangian dual. Thus,

$$\nu(D_S([S])) = \nu\left(\begin{array}{llll} \underset{v\geq 0}{\text{Max}} & \underset{\alpha\geq 0}{\text{Max}} & \text{Min} & cx + \alpha v(Ax - b) \\ & & & \text{s.t.} \quad x \in [S] \end{array}\right)$$

Since $\alpha$ and $v$ appear only as a product inside the maximization, the problem can be reduced to one in $u = \alpha v$ giving

$$\nu(D_S([S])) = \nu\left(\underset{u\geq 0}{\text{Max}} \ \underset{x\in[S]}{\text{Min}} \ cx + u(Ax - b)\right) \underline{\Delta} \, \nu(D_L([S])).$$

By Theorem 3.2 the last quantity is equivalent to $\nu(D_L(S))$.

(ii) The proof is identical to part (i) with $[S]$ replaced by $(\bar{S})$.

(iii) By the integrality property of $P^V(S)$, $\nu(P^V(S)) = \nu(P^V(\bar{S}))$ for all $v \geq 0$ including any optimal v. Thus, $\nu(D_S(S)) = \nu(D_S(\bar{S}))$. Using this result, part (ii), Theorem 3.2, and the definitions of various duals one obtains

$$\nu(P(\bar{S})) = \nu(D_L(\bar{S})) \leq \nu(P_{\bar{u}}(S)) \leq \nu(D_L(S)) \leq \nu(D_S(S)) = \nu(D_S(\bar{S})).$$

It then follows from part (ii) that $\nu(P(\bar{S})) = \nu(P_{\bar{u}}(S)) = \nu(D_L(S)) = \nu(D_S(S))$.

Q.E.D.

There are several implications of Theorem 3.4. First, note that if $P^V(S)$ has the surrogate integrality property, neither $D_L(S)$ nor $D_S(S)$ is of any value in improving on the bound from $P(\bar{S})$. Also, if one relaxes $P^V(S)$, and solves it as a linear program, when it does not have the integrality property, one still only gets $\nu(P(\bar{S}))$ as a dual value. Thus, the problem $P^V(S)$ must be solved optimally for the dual to be helpful.

It seems obvious that it would be harder to find an example for which $P^V(S)$ has the integrality property than one where $P_u(S)$ has the property. The next theorem and corollary will demonstrate that this observation is very much the case. First note that all examples in the literature of the integrality property holding for a problem $P_u(S)$ are based on the structure of the constraint set $Ax \leq b$, $x \in S$, and not on the cost vector c.

Theorem 3.5

If S is finite, $P(\bar{S})$ has a unique optimal solution, and $\nu(P(\bar{S})) = \nu(D_S(S))$ for all cost vectors c, then $\nu(P(S)) = \nu(P(\bar{S}))$.

Proof:

Let $x^{LP}$ be the unique solution to $P(\bar{S})$. Since $x^{LP}$ is unique, one can perturb the cost function in any arbitrary direction d, and still keep $x^{LP}$ unique, for a sufficiently small step $\alpha$ in the direction d.

By assumption $\nu(D_S(S)) = \nu(D_S(\bar{S})) = \nu(P(\bar{S}))$. So there must exist $x \in \Omega(D_S(S))$ such that $cx = cx^{LP}$, i.e., $x \in S$ which lie on the plane L defined by $cx = \nu(P(\bar{S}))$.

Since S is a finite set, there exists an $\alpha > 0$ such that $|cx - cx^{LP}| > \alpha$ for all $x \in S$ with $x \notin L$. That is the $x \in S$ which do not lie in the plane L, must be some strictly positive distance from L. Therefore, for a sufficiently small change $\alpha d$ in c, all such x will not be contained in the plane L′ defined by $(c + \alpha d)x = (c + \alpha d)x^{LP}$. But by assumption $\nu(D_S(S)) = \nu(P(\bar{S}))$ for all c. Thus, there must exist $x \in S$ such that $x \in L$ <u>and</u> $x \in L'$.

Choose $d = e_1$, the first unit vector. Then for $\alpha_1$ sufficiently small, there must exist $x \in S$ such that $x \in L$ and $x \in L_1$ where $L_1$ is the plane defined by $(c + \alpha_1 e_1)x = (c + \alpha_1 e_1)x^{LP}$. But $cx = cx^{LP}$ since $x \in L$, so we must have

$\alpha_1 e_1 x = \alpha_1 e_1 x^{LP}$ which implies $x_1 = x_1^{LP}$. That is, the first component of $x^{LP}$ is the same as the first component of an $x \varepsilon S$.

Now all $x \varepsilon S$ such that $x \not\varepsilon L$ or $x \not\varepsilon L_1$ are at some strictly positive distance from the plane $L_1$. Choose another direction $d = e_2$. Again, for $\alpha_2 > 0$, sufficiently small, there must exist $x \varepsilon S$ with $x \varepsilon L$ and $x \varepsilon L_1$ such that $(c + \alpha_1 e_1 + \alpha_2 e_2)x = (c + \alpha_1 e_1 + \alpha_2 e_2)x^{LP}$. But $x \varepsilon L_1$ implies $x_2 = x_2^{LP}$.

Now continue to interatively construct new costs $c + \alpha_1 e_1 + \alpha_2 e_2 + \ldots + \alpha_j e_j$ such that there must exist an $x \varepsilon S$ with $x_i = x_i^{LP}$, $i = 1, 2, \ldots, j$. Letting $j = n$ implies that $x^{LP} \varepsilon S$. But then $x^{LP}$ is feasible for $P(S)$. Since $\nu(P(\bar{S})) \leq \nu(P(S))$, $x^{LP}$ is an optimal solution for $P(S)$ and $\nu(P(S)) = \nu(P(\bar{S}))$.

Q.E.D.

Theorem 3.5 implies that if surrogate duals for a problem yield no improvement over the linear programming relaxation, then the original integer problem can be solved as a linear program. Since the integrality property is a sufficient condition for $\nu(P(\bar{S})) = \nu(D_S)$, we have the following corollary:

Corollary 3.6

If $(P^V(S))$ has the integrality property for all c, then $\nu(P(S)) = \nu(P(\bar{S}))$.

Proof:

By the integrality property for all c and Theorem 3.4, $\nu(D_S(S)) = \nu(P(\bar{S}))$ for all c. The conclusion now follows via Theorem 3.5.

Q.E.D.

Thus, Corollary 3.6 says that one need not worry about the integrality property pertaining to the surrogate relaxation for most problems. If S is finite, then $\nu(D_S)) = \nu(P(\bar{S}))$ for all c only if P(S) can be solved as a linear program. In general, this would happen only for very specially structured problems under a condition such as unimodularity.

Note that Corollary 3.6 implies that none of the problem formulations in Examples 1, 2, 3 and 4 can have the surrogate integrality property for general cost functions, even though they have the lagrangian integrality property. If they did, then Theorem 3.5 and Corollary 3.6 would indicate that $\nu(P(\bar{S})) = \nu(P(S))$ so that each formulation could be solved using linear programming. Each example, however, is not generally solvable by linear programming; that is, they are truly integer programming problems.

To further see the bounding advantage of $D_S(S)$ note that one can generally improve on the $\nu(P(\bar{S}))$ bound for $\nu(P(S))$ by solving the one subproblem $P^{\bar{u}}(S)$ where $\bar{u}$ is obtained from the linear program $P(\bar{S})$. This case can occur even when the lagrangian relaxation has the integrality property. Figure 5 presents a graphical example of just such a problem (Example 1 of the previous section).

Let $S = \{x : 0 \leq x \leq 1,\ x \text{ integer}\}$

The optimal lagrangian solution takes on the same value as the solution to $P(\bar{S})$, because $P_u(S)$ has the lagrangian integrality property, i.e., $\nu(D_L(S)) = \nu(P(\bar{S}) = cx^0$. The plane $\bar{u}(Ax - b) = 0$ will pass through $x^0$ and be parallel to the cost contours of c. An optimal surrogate solution must occur at points $x^1$, $x^2$, or $x^3$, all lying in higher level sets of the objective function cx. Thus, $\nu(P^{\bar{u}}(S)) > \nu(P_{\bar{u}}(S)) = \nu(P(\bar{S}))$.

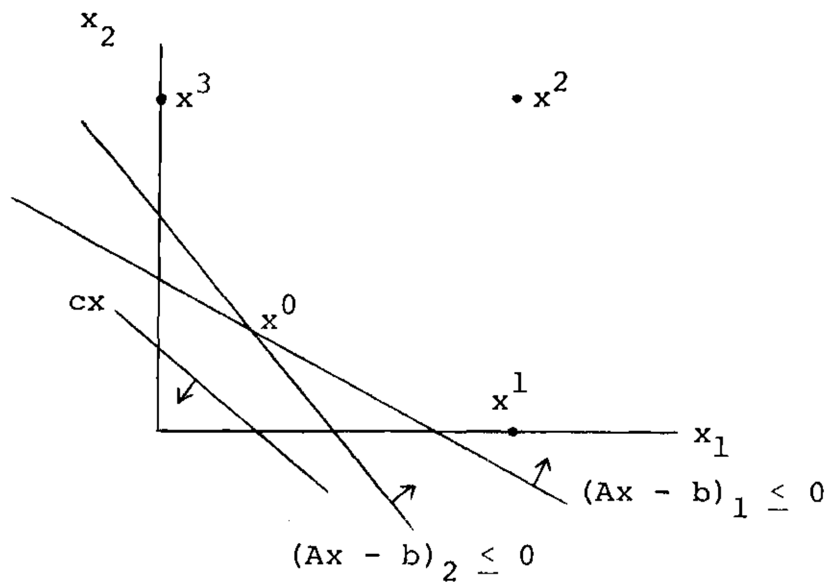


Figure 5. Example of Integrality Property for $(P_u)$, not $(P^u)$

In the next section, further characterization of when gaps may occur between the lagrangian and surrogate duals is developed.

## Lagrangian-Surrogate Gaps

It was shown in the previous section that one condition for no gap to exist between the surrogate and lagrangian duals is that $(P^v)$ have the integrality property. However, Theorem 3.5 demonstrated that the integrality property will not occur in many problems of interest.

Concentration here is on further characterizations of when such a gap does or does not exist, i.e., when $\nu(D_L) =$ or $< \nu(D_S)$. Recall that the function $\Omega(\cdot)$ represents the set of optimal solutions to the problem $(\cdot)$. Consider first the following lemma.

Lemma 3.7

If $\nu(D_L) = \nu(D_S)$ and $u \in \Omega(D_L)$, then $u \in \Omega(D_S)$ and $\Omega(P^u) = \{x \in \Omega(P_u) : u(Ax - b) = 0\}$.

Proof:

To show $u \in \Omega(D_S)$, note that for all $x \in S$, $cx < \nu(D_L)$ implies $u(Ax - b) > 0$. Thus, all $x \in S$ with $cx < \nu(D_L)$ are infeasible in $(P^u)$, and $\nu(P^u) \geq \nu(D_L)$. But then $\nu(D_L) \leq \nu(P^u) \leq \nu(D_S) = \nu(D_L)$ and $u$ solves $(D_S)$.

Now if $x \in \Omega(P^u)$ then $u(Ax - b) \leq 0$. But by $x \in S$ and $u \in \Omega(D_L)$, $\nu(D_L) \leq cx + u(Ax - b) \leq cx \leq \nu(D_S) = \nu(D_L)$. Thus, $x \in \Omega(P_u)$ and $u(Ax - b) = 0$. Conversely, if $x \in \Omega(P_u)$ and

$u(Ax - b) = 0$, then $\nu(D_L) = cx = \nu(D_S)$ and $u(Ax - b) \leq 0$ so that $x \varepsilon \Omega(P^u)$.

Q.E.D.

The observations in Lemma 3.1 lead immediately to the next theorem.

Theorem 3.8

Either $\nu(D_S) > \nu(D_L)$ or for every $u \varepsilon \Omega(D_L)$, there exist $x \varepsilon \Omega(P_u)$ such that $u(Ax - b) = 0$.

Proof:

Lemma 3.7 shows that if $\nu(D_S) = \nu(D_L)$ then the set $\Omega(P^u)$ is identical to $\{x \varepsilon \Omega(P_u): u(Ax - b) = 0\}$. The theorem follows directly from the fact that $\Omega(P^u)$ is nonempty.

Q.E.D.

Theorem 3.8 states that the surrogate and lagrangian dual values can be equal only if there exist complementary x's for every optimal lagrange multiplier u, i.e., x's with $u(Ax - b) = 0$. This result has some value in arguing the merits of the surrogate approach, since it seems unlikely in most integer programs that an exactly complementary x could be found for every optimal lagrangian multiplier u.

Also, recall that complementarity plays a key role in the theory of gaps between $\nu(D_L)$, $\nu(D_S)$ and $\nu(P)$. That is, if an optimal solution to $(P_u)$ is feasible in (P) and satisfies $u(Ax - b) = 0$, then it is optimal in (P). In the surrogate case the corresponding result does not require complementarity. Thus, it is not surprising that complementarity arises here as the key issue in gaps between $\nu(D_L)$ and $\nu(D_S)$. It is also possible to apply Theorem 3.8 in a constructive test to detect a gap between $\nu(D_L)$ and $\nu(D_S)$.

Theorem 3.9

Let u be an optimal multiplier for $(D_L)$ and define the (possibly empty) set $\Lambda = \{x \varepsilon \Omega(P_u): \ u(Ax - b) = 0\}$. If one solves

$$\begin{aligned} \text{Max} \quad & \alpha \\ \text{s.t.} \quad & d(Ax - b) \geq \alpha \quad \text{for all } x \varepsilon \Lambda \\ & d \geq 0 \end{aligned}$$

and the optimal $\alpha = 0$, then $\nu(D_L) = \nu(D_S)$. If S is a finite set and $\alpha \neq 0$, $\nu(D_S) > \nu(D_L)$.

Proof:

Clearly, $\alpha \geq 0$, and when $\Lambda$ is empty, $\alpha$ is arbitrarily large which corresponds to the result of Theorem 3.8 that $\nu(D_S) > \nu(D_L)$. If $\Lambda$ is nonempty, then from Lemma 3.7, $\Omega(P^u) = \Lambda$. If u is not the optimal surrogate multiplier, then there must be $w \geq 0$, such that $w(Ax - b) > 0$ for every $x \varepsilon \Omega(P^u)$, i.e., such that all $x \varepsilon \Omega(P^u)$ are made infeasible in $(P^w)$. If $\alpha = 0$ there is no such w and $\nu(D_L) = \nu(D_S)$. If $\alpha > 0$, then there exists a direction d such that $(u + \beta d)(Ax - b) > 0$ for all $x \varepsilon \Lambda$, for all $\beta > 0$. Moreover, by the optimality of u in $(D_L)$, $u(Ax - b) > 0$ for all $x \varepsilon S$ such that $cx < \nu(D_L)$. Thus, if S is finite, for the direction d chosen above, there must exist some $\beta > 0$ such that $(u + \beta d)(Ax - b) > 0$ for all

$x \in S$ with $cx \leq \nu(D_L)$. This implies that one can improve on $\nu(D_L)$ in $(D_S)$ by using $u + \beta d$ as a surrogate multiplier. That is to say, $\nu(D_L) = \nu(P^u) < \nu(P^{u+\beta d}) \leq \nu(D_S)$, and the theorem follows.

Q.E.D.

Theorem 3.9 suggests a two phase, lagrangian/surrogate algorithm which proceeds as follows:

Phase I: (a) Solve $(D_L)$ for the optimal multiplier u.

(b) Identify all complementary solutions to $(P_u)$, i.e., $x \in \Omega(P_u)$ with $u(Ax - b) = 0$

Phase II: (a) Solve the linear program in Theorem 3.9 to determine if a surrogate dual optimization should be undertaken.

(b) If the optimal $\alpha$ in Theorem 3.9 is positive, attempt to improve further on the surrogate multiplier beginning with u or $u + \beta d$ as a starting surrogate multiplier.

Before further developing this two phase concept, note that the above theorems are trivially satisfied if $(P^u)$ has the integrality property. In that case, $(P_u)$ can be solved as a linear program and complementary $x \in \Omega(P_u)$ which cannot be "cut off" in $(P^u)$ are assured.

The previous discussion concluded that integer programs of interest which have the surrogate integrality problem are probably uncommon. However, as shown earlier, many of the most natural lagrangian formulations of integer programs do have the integrality property. The two phase approach above seems well suited to such cases. If the lagrangian relaxation can be seen to have the integrality property then to find the optimal u one needs only to solve the linear programming problem $P(\bar{S})$, and proceed immediately to Phase II in the above procedure. The complex problem of finding improved multipliers at Step II(b) is addressed in Chapter IV.

## Composite Dual

Extending the results of the previous section to encompass the composite dual is straightforward. The results developed may be seen to be somewhat stronger in the case of integrality properties and weaker in the case of gaps between $(D_S)$ and (D).

### Primal-Dual Gap

Glover (26) shows that the conditions for no gap between the composite dual and the primal are exactly the same as those for the lagrangian-primal case. That is, there must be an optimal solution to (D) which is feasible for (P) and which is complementary with respect to the lagrange multiplier u, i.e., $u(Ax - b) = 0$. It appears that these conditions may be harder to satisfy in comparison with the surrogate case. However, $u = 0$ is a complementary feasible solution for the composite dual, so that the requirements are actually weaker.

## Integrality Property

One may extend all of the previous results with regard to integrality properties to the composite dual. In the composite case the problem $(P_u^v)$ can be said to have the integrality property if $\nu(P_u^v(S)) = \nu(P_u^v(\bar{S}))$ for all u, $v \geq 0$. It will be shown that, as in the surrogate case, the bounding weaknesses of many lagrangian formulations do not generally hold for composite formulations.

The following theorem is an extension of the results of Theorem 3.4.

Theorem 3.10

(i) $\nu(D_L(S)) = \nu(D_S([S])) = \nu(D([S]))$.

(ii) $\nu(P(\bar{S})) = \nu(D_S(\bar{S})) = \nu(D(\bar{S}))$.

(iii) If $P_u^v(S)$ has the integrality property then $\nu(P(\bar{S})) = \nu(D_L(S)) = \nu(D_S(S)) = \nu(D(S))$.

Proof:

(i) $$\nu(D([S])) = \nu\left(\begin{array}{lll} \underset{u,v\geq 0}{\text{Max}} & \text{Min} & cx + u(Ax - b) \\ & \text{s.t.} & v(Ax - b) \leq 0 \\ & & x \in [S] \end{array}\right)$$

The inner minimization problem is convex so it equals its lagrangian dual. Thus,

$$\nu(D([S])) = \nu\left(\begin{array}{llll} \underset{u,v\geq 0}{\text{Max}} & \underset{\alpha\geq 0}{\text{Max}} & \text{Min} & cx + u(Ax - b) + \alpha v(Ax - b) \\ & & \text{s.t.} & x \in [S] \end{array}\right)$$

The problem can be reduced to one in $w = u + \alpha v$ giving

$$\nu(D([S])) = \nu\left(\begin{array}{lll} \underset{w \geq 0}{\text{Max}} & \text{Min} & cx + w(Ax - b) \\ & \text{s.t.} & x \in [S] \end{array}\right) \underline{\Delta} \nu(D_L([S])).$$

By Theorem 3.4 the last quantity is equivalent to $\nu(D_L(S))$ and $\nu(D_S([S]))$.

(ii) The proof is identical to part (i) with $[S]$ replaced by $(\bar{S})$.

(iii) By the integrality property of $P_u^v(S)$, $\nu(P_u^v(S)) = \nu(P_u^v(\bar{S}))$ for all $u,v \geq 0$ including any optimal u,v pair. Thus $\nu(D(S)) = \nu(D(\bar{S}))$. Using this result and the definitions of various duals one obtains

$$\nu(P(\bar{S})) \leq \nu(D_L(S)) \leq \nu(D_S(S)) \leq \nu(D(S)) = \nu(D(\bar{S})).$$

It then follows from part (ii) that equalities are obtained throughout the above expression.

Q.E.D.

Again, note that if $P_u^v(S)$ has the integrality property, none of the dual formulations is of any value in improving on the bound from $P(\bar{S})$. If one relaxes $P_u^v(S)$ and solves it as a linear program when it does not have the integrality property, one still only gets $\nu(P(\bar{S})$ as a dual value.

As in the surrogate case, it would seem difficult to find a problem for which $P_u^v(S)$ has the integrality property. The following theorem is similar to Theorem 3.5. If the theorem is stated in terms of $P_u^v(S)$ having the integrality property for all c, then the proof would be identical to that of Theorem 3.5. However, here one can draw the same conclusions as before with less restrictive assumptions.

Theorem 3.11

If S is finite, $P(\bar{S})$ has a unique optimal solution, and $P_u^V(S)$ has the integrality property, then $\nu(P(\bar{S})) = \nu(P(S))$.

Proof:

Let $x^{LP}$ be the optimal basic solution to $P(\bar{S})$. Since $x^{LP}$ is unique, the cost function can be perturbed in any arbitrary direction d, and still keep $x^{LP}$ unique for a sufficiently small step $\alpha$, in the direction d.

By the integrality property and Theorem 3.10, $\nu(D(S)) = \nu(D_S(S)) = \nu(P(\bar{S}))$. So there must exist $(u,v) \varepsilon \Omega(D(S))$ and $x \varepsilon \Omega(P_0^V)$ such that $u = 0$ and $cx = cx^{LP}$, i.e., $x \varepsilon S$ which lie on the plane L defined by $cx = \nu(P(\bar{S}))$. Since S is a finite set, there exists a $\delta > 0$, such that $|cx - cx^{LP}| > \delta$ for all $x \varepsilon S$ such that $x \not\varepsilon L$. That is, the $x \varepsilon S$ which do not lie on the plane L, must be some strictly positive distance from L. Therefore, for a sufficiently small change $\alpha u$ in u all such x will not be contained in the plane L′ defined by $cx + \alpha u(Ax - b) = cx^{LP} + \alpha u(Ax^{LP} - b)$. But by assumption $\nu(D(S)) = \nu(P(\bar{S}))$ for all $u,v \geq 0$, so there must exist $x \varepsilon S$ such that $x \varepsilon L$ <u>and</u> $x \varepsilon L'$. Choose $u = e_1$, the first unit vector. Then for $\alpha_1$ sufficiently small, there must exist $x \varepsilon S$ such that $x \varepsilon L$ and $x \varepsilon L_1$ where $L_1$ is the plane defined by $cx + \alpha_1 e_1(Ax - b) = cx + \alpha_1 e_1(Ax^{LP} - b)$. But $cx = cx^{LP}$

since $x \in L$, so $\alpha_1 e_1(Ax - b) = \alpha_1 e_1(Ax^{LP} - b)$ which implies $(A_1 x - b_1) = (A_1 x^{LP} - b_1)$ and x is feasible for the first constraint. (Here $A_i$ is the $i^{\underline{th}}$ row of A.)

Now, for $\alpha_1$ sufficiently small, all $x \in S$ such that $x \notin L$ or $x \notin L_1$ are at some strictly positive distance from $L_1$. Choose another direction $u = e_2$. For $\alpha_2 > 0$ sufficiently small, there must exist $x \in S$ with $x \in L$ and $x \in L_1$ such that $cx + (\alpha_1 e_1 + \alpha_2 e_2)(Ax - b) = cx^{LP} + (\alpha_1 e_1 + \alpha_2 e_2)(Ax^{LP} - b)$. But $x \in L_1$ implies that $(A_2 x - b) = (A_2 x^{LP} - b)$ and x is feasible for the first two constraints.

Now iteratively construct new multipliers $\alpha_1 e_1 + \alpha_2 e_2 + \ldots + \alpha_j e_j$, such that there must exist an $x \in S$ satisfying $(A_i x - b_i) = (A_i x^{LP} - b_i)$, $i = 1,2,\ldots,j$. Letting $j = m$ implies that there exists an $x \in S$ which is feasible in $P_u^V(S)$ and optimal in $P(\bar{S})$. But then x is feasible for P(S) and $cx = cx^{LP} = \nu(P(\bar{S}))$. In general, $\nu(P(\bar{S})) \leq \nu(P(S))$, so x is an optimal solution for P(S), and $\nu(P(\bar{S})) = \nu(P(S))$. Q.E.D.

One can draw the same type of conclusions as those for Theorem 3.5 and Corollary 3.6. That is, one need not worry about the integrality property pertaining to the composite relaxation for most problems. Thus, in turn, one would expect strict improvement of $\nu(D)$ over lesser duals. Note that if $(P_u^V)$ has the integrality property then so do $(P_u)$ and $(P^V)$ since one can set u or v equal to zero in $(P_V^u)$. In this case, $\nu(P(\bar{S})) = \nu(D_L(S)) = \nu(D_S(S)) = \nu(D(S))$.

## Surrogate-Composite Gaps

Here again, interest is centered on determining when one can improve on one dual by using the 'next stronger' dual. As noted in the previous section, one condition for no gap to exist between the surrogate and composite duals is the condition that $(P_u^v)$ have the integrality property. However, this case rarely occurs.

It would be beneficial to develop conditions and a gap detection test similar to those shown earlier for the lagrangian-surrogate case. Towards that end, consider the following two lemmas.

Lemma 3.12

If $\nu(D_S) = \nu(D)$ and $v \in \Omega(D_S)$, then $(0,v) \in \Omega(D)$.

Proof:

$$\nu(D_S) = \nu(P_0^v) \leq \nu(D) = \nu(D_S).$$

Q.E.D.

Lemma 3.13

Given $v \in \Omega(D_S)$, if there exists a $u \geq 0$ such that $u(Ax - b) > 0$ for all $x \in \Omega(P^v)$, then $\nu(D) > \nu(D_S)$.

Proof:

$(P^v) = (P_0^v)$ so $\Omega(P_0^v) = \Omega(P^v)$. The u defined above are just the lagrangian ascent directions for $(P_0^v)$.

Q.E.D.

Lemma 3.13 gives a sufficient condition for a gap to exist between $\nu(D)$ and $\nu(D_S)$. In particular, given an optimal multiplier $v \in \Omega(D_S)$, one could solve the following linear program:

$$\begin{array}{ll} \text{Max} & \alpha \\ \text{s.t.} & d(Ax - b) \geq \alpha \text{ for all } x \in \Omega(P^{v}) \\ & d \geq 0 \end{array}$$

If $\alpha \neq 0$ then by Lemma 3.13 one can conclude that $\nu(D) = \nu(D_S)$ and use d as the ascent direction in $(P_0^{v})$. However, the conclusion that $\alpha = 0$ implies $\nu(D) = \nu(D_S)$ does not apply as in the lagrangian-surrogate case. This unfortunate fact results from the existence of strict local maxima in $\nu(P_u^{v})$ noted in Chapter II.

## Multiple Surrogate Dual

The case of the multiple surrogate dual will be covered briefly since all of the results are simple extensions to those described earlier in this chapter. The primal-dual gap condition is shown by Glover (26) to be the same as in the single surrogate case, that is, for $(v^1,\ldots,v^k) \in \Omega(D_S k)$, there must be an $x \in \Omega(P^{v^1,\ldots,v^k})$ which is feasible for (P). The integrality property results mimic the surrogate case in that if $(P^{v^1,\ldots,v^k})$ has the integrality property for all c, then $\nu(P(S)) = \nu(P(\bar{S}))$ and all dual relaxations are trapped in between.

The conditions for $\nu(D_S 2) = \nu(D)$ are very similar to the conditions for $\nu(D) = \nu(D_L)$. That is, if $(u,v) \in \Omega(D)$, find all $x \in \Omega(P_u^{v})$ such that $u(Ax - b) = 0$. If there exists a vector $d \geq 0$ such that $d(Ax - b) > 0$ for all such x then $\nu(D_S2) > \nu(D)$ (provided, of course, that $\nu(P) > \nu(D)$). The existence of local maxima in $(P_u^v)$ and $\nu(P^{u,v})$, makes this only a sufficient condition for a gap. If there is no vector $d \geq 0$ such that $d(Ax - b) > 0$ for the x defined above, there may still be a gap between $\nu(D)$ and $\nu(D_S2)$.

## Conclusions

The numerous specific results developed in this chapter indicate several broad conclusions. First, they indicate the extreme likelihood of improved bounds when using $\nu(D_S)$ instead of $\nu(D_L)$. It was shown that many natural lagrangian formulations lead to $\nu(D_L) = \nu(P(\bar{S}))$ via the integrality property. For problems in general, surrogate duals can consistently fail to exceed $\nu(P(\bar{S}))$ only if $\nu(P(\bar{S})) = \nu(P)$. Similarly, the existence of no lagrangian-surrogate gap minimally requires complementary solutions to all optimal lagrangian relaxations---a doubtful prospect for general integer programs.

The lack of 'searchability' of the composite and multiple surrogate duals lead to less powerful gap tests and in some cases, comparatively incomplete results for these two duals. Thus, while theoretically interesting, these duals seem of less practical value than the surrogate.

# CHAPTER IV

# SURROGATE MULTIPLIER SEARCH PROCEDURES

As mentioned in Chapter I, the only procedure to date for finding surrogate multipliers was suggested by Banerjee (4) in a dissertation that centered on lagrangian duality. No computational experience was reported and some improvements or refinements are possible on his algorithm. Banerjee's algorithm will be reviewed and refined here, and two more search procedures will be developed. Computational comparisons of these procedures are given in Chapter VI. As much of this dissertation involves the extension of lagrangian dual results to the surrogate case, a section is also included that discusses the subgradient oriented search techniques which have been quite successful in the lagrangian case.

## Subgradient Oriented Procedures

With the exception of the Dantzig-Wolfe or Benders' type procedures, the dual multiplier search schemes in the lagrangian case largely depend on some subgradient-oriented direct search method (see for example the survey in (5)). Bazaraa and Goode (5) present necessary and sufficient conditions for ascent and steepest ascent directions at a given u based on the subgradients, $A\bar{x} - b$, for $\bar{x}$ which are optimal solutions to the lagrangian relaxation $(P_u)$. Similar necessary conditions can be developed for an ascent direction in the surrogate case, but a counterexample will be presented to show such conditions are not sufficient, and thus that subgradient-oriented schemes do not work in the surrogate case.

Lemma 4.1

Let $v \geq 0$ be a current surrogate multiplier and d an arbitrary direction. If there exists an $x \varepsilon \Omega(P^v)$ such that $d(Ax - b) \leq 0$, then $\nu(P^{v+\alpha d}) \leq \nu(P^v)$ for all $\alpha \geq 0$.

Proof:

If $x \varepsilon \Omega(P^v)$ then $v(Ax - b) \leq 0$. Also, $d(Ax - b) \leq 0$. Thus, $(v + \alpha d)(Ax - b) \leq 0$ for all $\alpha \geq 0$, and x is a feasible solution for $(P^{v+\alpha d})$. But then $\nu(P^{v+\alpha d}) \leq cx = \nu(P^v)$.

Q.E.D.

The next theorem follows immediately from Lemma 4.1 and states necessary conditions for d to be an ascent direction. These conditions were derived independently by Banerjee (4).

Theorem 4.2

Any direction d for which there exists $\alpha \geq 0$ satisfying $\nu(P^{v+\alpha d}) > \nu(P^v)$ (i.e., which is an ascent direction for the surrogate problem) must satisfy

$$d(Ax - b) > 0 \text{ for all } x \varepsilon \Omega(P^v) \qquad (4\text{-}1)$$

Corollary 4.3

Any direction d for which there exists $\alpha \geq 0$ satisfying $\nu(P^{v+\alpha d}) > \nu(P^v)$ is an ascent direction for the lagrangian problem

$$(P_0^v) \quad \text{Minimize} \quad cx + 0(Ax - b)$$
$$\text{subject to } x \in S,\ v(Ax - b) \leq 0$$

Proof:

$(P_0^v)$ is identical to $(P^v)$. Thus, $\Omega(P^v) = \Omega(P_0^v)$ and the subgradients for $(P_0^v)$ are the $Ax - b$ for $x \in \Omega(P^v)$. The condition of (4-1) are exactly the ones required for a lagrangian ascent direction (see Bazaraa and Goode (5)). Q.E.D.

Corollary 4.3 implies that (4-1) is not only a necessary condition but is also a sufficient condition for d to be a lagrangian ascent direction in problem $(P_0^v)$. The steepest ascent direction for $(P_0^v)$ can be shown to be the shortest subgradient under an appropriate norm. One might hope that there is an ascent direction in $(P^v)$ corresponding to the shortest subgradient in $(P_0^v)$ with respect to some norm. However, in the following example, the subgradient is unique and is not an ascent direction for the surrogate dual, even though one exists. Therefore, it appears that there cannot be a parallel development in the surrogate case of the subgradient oriented search methods for the lagrangian dual. Consider the following problem:

$$\begin{array}{ll} \text{Min} & x_1 + x_2 - x_3 \\ \text{subject to} & -x_1 + x_2 - 1 \leq 0 \\ & -x_2 + 1 \leq 0 \\ & x_1 + x_2 + 11x_3 - 10 \leq 0 \\ & x \in S \end{array}$$

where $S = \{x\colon\ 0 \leq x_i \leq 2,\ x_i \text{ integer},\ i = 1,2,3\}$

Let $v = (5, 1, 20)$, $v(Ax - b) = 15x_1 + 24x_2 + 220x_3 - 204$. Then $\Omega(P^v) = x^*$ where $x^* = (0, 0, 0)$ with $Ax^* - b = (-1, 1, -10)$ unique. A superoptimal point $\hat{x} = (0, 0, 1)$ is cut off by $v$ since $v(A\hat{x} - b) = 16 > 0$. $A\hat{x} - b = (-1, 1, 1)$. Since $x^*$ is unique, choose $d = Ax^* - b$ as the search direction.

(i) In order to cut off $x^*$ and increase the value of the surrogate dual it is necessary that $(\bar{v} + \alpha d)(Ax^* - b) > 0$. This implies $\bar{v}(Ax^* - b) + \alpha d(Ax^* - b) > 0$ or $-204 + 102\,\alpha > 0$, implying $\alpha > 2$.

(ii) At the same time, however, $\hat{x}$ must remain infeasible. Thus forcing $(\bar{v} + \alpha d)(A\hat{x} - b) > 0$ or $16 - 8\,\alpha > 0$ which implies $\alpha < 2$. It follows that $Ax^* - b$ is not an ascent direction. But, $d = (-5, 11, -19)$ is an ascent direction. With $\alpha = 1$, the new $v = (5, 1, 20) + (-5, 11, -19) = (0, 12, 1)$. $v(Ax - b) = x_1 - 11x_2 + 11x_3 + 2 \leq 0$. The optimal solution to $(P^v)$ has a solution value of one and is given by $x = (0, 1, 0)$ and $(0, 2, 1)$.

## A Benders' Type Procedure, (BTP)

Although subgradient-oriented direct search procedures were shown to be inappropriate in the previous section, one can suggest an algorithm, derived independently by Banerjee (4), for the surrogate dual which parallels the Benders' style procedure in (7) for the lagrangian case. Benders' procedure for the lagrangian dual may be summarized as follows:

Step 0: Let $k = 1$, $u^k = 0$

Step 1: Solve $P_u k$ with optimal solution $x^k$ and $y^k = Ax^k - b$. If $y^k \leq 0$ and $u^k y^k = 0$, then stop, $x^k$ is optimal to the primal problem. Otherwise, go to Step 2.

Step 2: Solve the linear program

Max $z$ subject to

$$cx^j + uy^j \geq z \qquad j = 1, 2, \ldots, k$$

$$u \geq 0$$

If $z = \nu(P_u k)$, then $(u^k, x^k)$ is an optimal solution to $D_L$. Otherwise, denote the optimal u by $u^{k+1}$ and replace k by $k + 1$. Go to Step 1.

The necessary conditions given in Theorem 4.2 to improve the surrogate relaxation $(P^v)$ lead to a similar algorithm, (BTP), for the surrogate dual. To avoid any confusion when comparing and combining various problem solutions and techniques, define

$\nu^*(\cdot)$ = value of an incumbent solution to the problem $(\cdot)$

Step 0: Set $k = 1$, $v^k = v^* = 0$, $\nu^*(D_S) = -\infty$.

Step 1: Solve $(P^{v^k})$ with optimal solution $x^k$ and $y^k = Ax^k - b$.

If $y^k \leq 0$, then stop, $x^k$ is optimal to the primal problem.

If $cx^k > \nu^*(D_S)$, let $\nu^*(D_S) = cx^k$, $v^* = v^k$.

Step 2: Solve the linear program

$$\begin{aligned} &\text{Max} \quad z \quad \text{subject to} \\ &vy^j \geq z \qquad j = 1,2,\ldots,k \\ &1v \leq 1 \\ &v \geq 0 \end{aligned}$$

where $1v \leq 1$ simply normalizes the v's since $(P^v) = (P^{\beta v})$ for all $\beta > 0$. Let $\varepsilon^k = z$, and denote the above linear program by $(LP)_k$.

If $\varepsilon^k \leq 0$, stop; $(v^*, x^*, \nu^*(D_S))$ is optimal for $D_S$. Otherwise, denote the optimal v by $v^{k+1}$ and replace k by k + 1. Go to Step 1.

Clearly, if the set S is a finite discrete set then finite convergence is obtained, since every time Step 2 is successful ($\varepsilon^k > 0$), another member of S from the set of feasible solutions to $(P^{v^k})$ is cut off. If $\varepsilon^k < 0$, stop and an incumbent solution is optimal, since there exist no multipliers $v^{k+1}$ which will cut off all points $x^1, x^2, \ldots, x^k$ previously generated. It is necessary to keep an incumbent solution since the multiplier $v^k$ does not necessarily increase $\nu(P^{v^k})$ monotonically. In other words, $x^k$ satisfying $v^k(Ax^k - b) \leq 0$ in $(P^{v^k})$ have been infeasible in the problem

$(P^{v^{k-1}})$. Thus if $cx^k < \nu(P^{v^{k-1}})$, $\nu(P^{v^k}) < \nu(P^{v^{k-1}})$. Keeping an incumbent solution is not necessary in the lagrangian case unless one plans to stop before reaching optimality by terminating at the iteration k where $z - \nu(P_u k) < \eta$ for some specified $\eta > 0$.

The above arguement for finiteness of the procedure holds when the S finite condition is relaxed to cover the following type of problem.

$$\begin{aligned} \text{Min} \quad & c^1x^1 + c^2x^2 \\ \text{s.t.} \quad & v(A^1x^1 - b^1) \leq 0 \\ & x^2 \epsilon P(x^1) \\ & x^1 \epsilon S \end{aligned}$$

where $S = \{x^1\colon\ x^1$ is integer and possibly more constraints$\}$

$P(x^1) = \{x^2\colon$ given $x^1$ all optimal solutions to $(P^v)$ can be expressed as extreme points of the polyhedral set $P(x^1)\}$

Note the important fact that only $x^1$ figures in the surrogate constraint. Then if the optimal extreme points in $P(x^1)$ for $(P^v)$ are included in Step 2 of (BTP), there are only a finite number of feasible solutions to $(P^v)$ for all $v \geq 0$.

Banerjee presents a proof of convergence for the case of 'infinite S,' i.e., mixed-integer surrogate relaxtions not of the above form. Except for a change in notation the following proof is essentially his.

Theorem 4.3

Assume cx and (Ax - b) are bounded for all $x \varepsilon S$ and assume $\Omega(P) \neq \phi$. Then the algorithm (BTP) either

(a) terminates in Step 1 with an optimal surrogate multiplier $v^*$ and a solution $x^k$ optimal for (P)

or

(b) does not terminate, but the sequence $\{v^k\}$ contains a subsequence converging to $v^* \varepsilon \Omega(D_S)$.

Proof:

(a) If termination occurs in Step 1 then $v^*$ yields a solution to $(P^{v*})$ which is optimal to (P).

(b) Since $v^k$ is in the compact set $V = \{v: 1v \leq 1, v \geq 0\}$, if the algorithm does not terminate $\{v^k\}$ must contain a convergent subsequence $\{v^{k_\ell}\}$ denoted by $\{v^\ell\}$ such that $v^\ell \rightarrow v^* \varepsilon V$. Also since the feasible set for $(LP)_{k+1}$ is contained in the feasible set for $(LP)_k$ for all k, $\varepsilon^{k+1} \leq \varepsilon^k$ for all k. If the algorithm does not terminate, $\varepsilon^k > 0$ for all k. $\{\varepsilon^k\}$ is monotone nonincreasing and bounded below by zero; consequently $\varepsilon^k \rightarrow \bar{\varepsilon} \geq 0$. It will now be shown that $\bar{\varepsilon} = 0$. Consider the subsequence $\{\varepsilon^{k_\ell}\}$ (denoted by $\{\varepsilon^\ell\}$) of $\{\varepsilon^k\}$ corresponding to the subsequence $\{v^\ell\}$. Now $0 < \varepsilon^{k_\ell} =$

$$\underset{v \varepsilon V}{\text{Max}} \ \underset{0 \leq j \leq k_\ell - 1}{\text{Min}} \{v(y^j)\} = \underset{0 \leq j \leq k_\ell - 1}{\text{Min}} \{v^{k_\ell}\ (y^j)\} \leq v^{k_\ell}\ (y^{k_\ell - 1}),$$

i.e., $0 < \varepsilon^{\ell} \leq v^{\ell}(y^{\ell-1})$. Also since $v^{k}(y^{k}) \leq 0$ for all k the following must hold.

$$v^{\ell}y^{\ell} \leq 0 < \varepsilon^{\ell} \leq v^{\ell}(y^{\ell-1}) \tag{4-2}$$

Since the sequence $\{y^{\ell}\}$ belongs to a bounded set it must have a subsequence converging to $y^{*}$. Now let $\ell \to \infty$ and by (4-2)

$$v^{*}\cdot y^{*} \leq 0 < \lim \varepsilon^{\ell} = \bar{\varepsilon} \leq v^{*}\cdot y^{*}$$

$$\therefore \varepsilon^{k} \to 0. \tag{4-3}$$

Now it must be shown that $v^{*} \varepsilon \Omega(D_S)$. If $v^{*}$ is not an optimal surrogate multiplier it is sufficient to show that there does not exist an optimal surrogate multiplier $\bar{v}$. Such will be true if

$$\bigcap_{k=0}^{\infty} \{v \varepsilon V\colon\ v(y^{k}) > 0 \text{ for all } y^{k} = (Ax^{k} - b),\ x^{k} \varepsilon \Omega(P^{v^{k}})\} = \phi$$

Suppose otherwise. Then there exists $\bar{v} \varepsilon V$ such that $v(y^{k}) \geq \delta > 0$ for all k and since $\bar{v}$ is feasible for $(LP)_k$ for all k it must be that $\varepsilon^{k} \geq \bar{v}(y^{k}) \geq \delta > 0$ for all k, implying $\varepsilon^{k} \to \bar{\varepsilon} \geq \delta > 0$ which contradicts (4-3).

Q.E.D.

Note that neither the proof of convergence in the S finite case nor the S not finite case require that $(P^{v^{k}})$ be solved optimally in Step 1. All that is required is the generation of an x which is feasible in $(P^{v^{k}})$ and which must be made infeasible in order to obtain a solution to $(D_S)$. An x which satisfies these conditions is any x which is feasible in $(P^{v^k})$ and which has solution value less than or equal to $\nu^*(D_S)$. If the incumbent solution value $\nu^*(D_S)$ is to be improved upon, then x must be made infeasible by the proper choice of v. Any solution procedure for $(P^v)$ which generates feasible solutions before reaching optimality may generate such x's, at which time the solution procedure for $(P^v)$ may be stopped short of optimality. Such a procedure will be discussed further in Chapter V.

Step 2 in the algorithm (BTP) may involve solving very large linear programming problems as k increases. Note that as long as $\varepsilon^{k+1} \leq \varepsilon^k$, the proof of convergence in Theorem 4.3 follows. Consider dropping some or all of the nonbinding constraints in $(LP)_k$. Clearly solving $(LP)_k$ over just the binding constraints also yields $\varepsilon^k$ and $v^{k+1}$. Solving $(P^{v^{k+1}})$ yields $y^{k+1}$ with $v^{k+1}(y^{k+1}) \leq 0$. This implies that $(LP)_{k+1}$ will be more constrained than $(LP)_k$ and therefore $\varepsilon^{k+1} \leq \varepsilon^k$.

In the case of S finite, there clearly exists only a finite number of sets of binding constraints. By Theorem 4.3 and the discussion above, there can only be a finite number of iterations in $(LP)_k$ in which $\varepsilon^k$ does not strictly decrease. Each time $\varepsilon^k$ decreases, the previous set of binding constraints cannot be regenerated; thus finite convergence is guaranteed. Since the dimension of v is equal to m, at most m binding constraints will be present in $(LP)_k$ for any k. This should lead to substantial savings in storage and solution time in Step 2 considering the exponential order of the x's which could be feasible for the integer subproblems $(P^v)$.

It is not possible to guarantee that some of the $x^j$ (or $y^j$) will not be regenerated during the solution of the surrogate dual with the (BTP) procedure. However, it will be shown empirically in Chapter VI that a very small percentage of regenerations occur for the test problems considered. Initial empirical evidence indicates that normalizing the $y^j$ produces much faster convergence with fewer regenerations. Initial attempts were made to show that normalization would prevent any regeneration. Counterexamples were difficult to find; however, such examples were easy to construct in the non-normalized case.

In terms of an improved rate of convergence, consider the following hypothetical problem represented in Figure 6 in which $y^j$ has not been normalized.

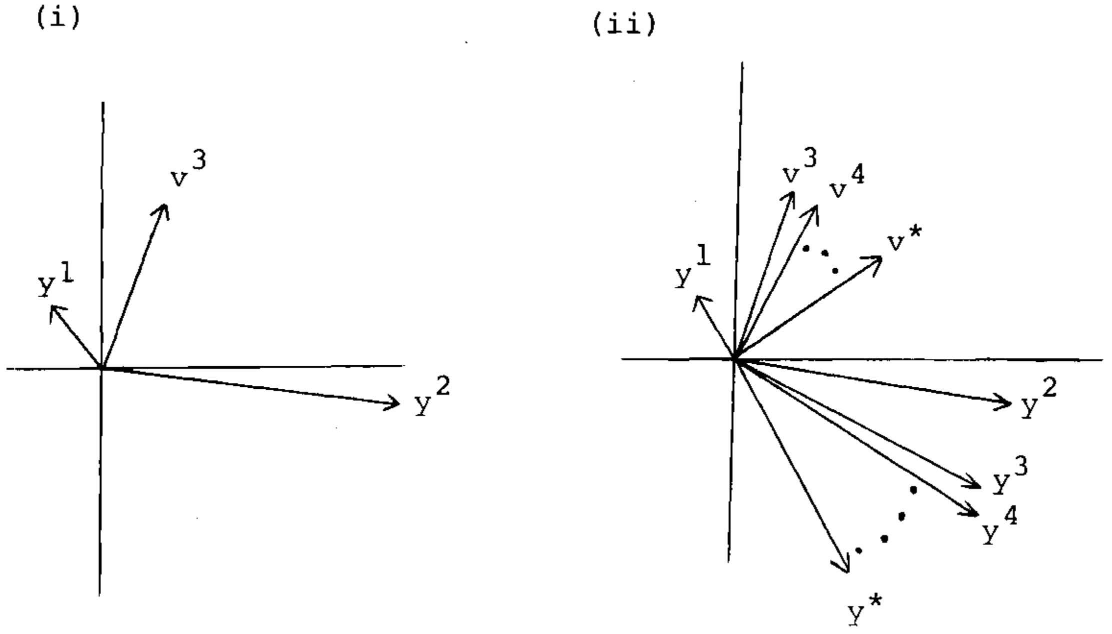


Figure 6. Normalizing Infeasiblities in (BTP)

An initial choice of $v^1 = (1,0)$ yields $x^1$ with $y^1$ as shown in Figure 6. Clearly, to maximize $v(y^1)$ with $v \geq 0$ yields $v^2 = (0,1)$. Now $(P^{v^2})$ yields $y^2$ as shown with considerably larger norm. The small norm of $y^1$ 'biases' $v^3$ to be 'closer' to $y^1$ than to $y^2$. Part (ii) of Figure 6 indicates that this bias yields a sequence of $y^j$, $v^j$ converging to $y^*$ and $v^*$. This convergence would clearly be speeded if the $y^j$ were normalized to prevent the bias on $y^1$ which keeps the sequence in $v^j$ closer in angle to $y^1$.

One can draw upon the similarity of the lagrangian and surrogate algorithms above to see how they could be used effectively in the two phase algorithm suggested in Chapter III. At the termination of the lagrangian Benders' procedure, one has available the optimal lagrangian multiplier $u^*$ which can be used in Step 1 of the surrogate algorithm. Since $\nu(P^u) \geq \nu(P_u)$ for all $u \geq 0$ one might go right into Phase II of the two phase procedure. One could also begin Step 2 of the surrogate algorithm with the set of $x_j$'s such that $cx_j \leq \nu(D_L)$ generated in the lagrangian procedure. (Strict inequality is possible because the $(P_u)$ objective function is $cx + u(Ax - b)$.) This would insure immediate improvement on $\nu(D_L)$ if any gap between $\nu(D_L)$ and $\nu(D_S)$ exists. Further exploitation of the obvious similarities between the two algorithms should lead to a significant savings in storage and computation time for any computer program for implementing the two phase approach. Even if the full procedure were not used, then in the context of a branch-and-bound procedure in which a lagrangian dual has been used successfully, it may prove beneficial to improve the bound on any branch by continuing with the surrogate algorithm for at least a few iterations.

## A Direct Search Procedure, (DSP)

Direct search procedures for optimal lagrange multipliers, generally subgradient oriented search procedures, have been the more successful approach (as compared to Benders' type procedures) in lagrangian duality. One might hope that a direct search procedure could be developed for the surrogate case with similar results. As shown earlier, subgradient oriented procedures are inappropriate; however, it is possible to develop a direct search procedure for optimal surrogate multipliers which strikes a strong resemblance to the subgradient oriented procedures of the lagrangian case. Both procedures have their early roots in works by Agmon (1) and Motzkin and Schoenberg (37). The important results of these two works are summarized in the discussion and theorem below.

The problem studied by Agmon, and Motzkin and Schoenberg, is the following system of linear inequalities for which a solution is to be obtained.

$$\ell_i(x) = \sum_{j=1}^{n} a_{ij}x_j + b_i \geq 0 \qquad i = 1,2,\ldots,m \qquad (4\text{-}4)$$

Consider the following iterative solution procedure for the system in (4-2).

<u>Step 0</u>: Set k = 0. Choose $x^k \varepsilon R^n$ and $\lambda$, $0 < \lambda \leq 2$ arbitrarily.

<u>Step 1</u>: If $x^k$ is a solution to (4-4), stop. Otherwise consider all indices i for which $\ell_i(x^k) < 0$ and choose one, say p, arbitrarily.

<u>Step 2</u>: Set $x^{k+1} = x^k + ta_p$, where $a_p$ is the vector $(a_{p1}, a_{p2},\ldots,a_{pn})$ and $t = -\lambda[\ell_p(x^k)]/|a_p|^2$. Replace k by k + 1 and go to Step 1.

This solution procedure may be referred to as 'linear relaxation' and will be denoted as (LR). The set of solutions to (4-4) are contained in a convex set (denoted here by H) which has dimension $r \leq m$. Agmon shows that at each iteration of (LR), $x^{k+1}$ is point-wise closer than $x^k$ to H. The following theorem due to Motzkin and Schoenberg (37) presents the convergence properties of (LR). The statement of the theorem is taken from Banerjee (4).

Theorem 4.4

Let the sequence $\{x^k\}$ be generated by the process (LR) for the system (4-4). If the system (4-4) is consistent then:

Case 1: $r = m$

(i) If $0 < \lambda < 2$ then the sequence $\{x^k\}$ either terminates or $x^k \to x \in$ boundary of H.

(ii) If $\lambda = 2$ then the sequence $\{x^k\}$ always terminates.

Case 2: $r < m$

If $0 < \lambda < 2$ then $\{x^k\}$ either terminates or $x^k \to x \in H$.

Proof:

See Motzkin and Schoenberg (37).

Q.E.D.

Convergence is based on generating a sequence of points which are monotonically point-wise closer to the solution set H. This property is what leads to the subgradient oriented procedures discussed earlier for the lagrangian dual. Although these procedures are not monotone in incumbent solution value, they are monotone in the above sense of producing dual multipliers which are point-wise closer to optimal dual multipliers. This same type of convergence property will be used to develop a direct search procedure for optimal surrogate dual multipliers.

Consider the surrogate dual for the case of finite discrete S or for the more general version stated earlier of 'finite S.' Then there is a finite set of superoptimal solutions to $(D_S)$, say $(x^1, x^2, \ldots, x^T)$, which must be made infeasible in $(P^v)$ in order to obtain an optimal solution to $(D_S)$. An optimal surrogate multiplier v may be found by solving the following system.

$$v(Ax^i - b) > 0 \qquad i = 1,2,\ldots,T \tag{4-5}$$

$$v \geq 0 \tag{4-6}$$

If v is a solution to (4-5) and (4-6) then clearly $\alpha v$ is also a solution for all $\alpha > 0$. Equation (4-5) may now be replaced by

$$v(Ax^i - b) \geq \varepsilon \qquad i = 1,2,\ldots,T \tag{4-7}$$

for any $\varepsilon > 0$. If the solution to (4-5) has minimal $v(Ax^i - b)$ equal to $\delta > 0$, then $\alpha v$ solves (4-7) where $\alpha = \varepsilon/\delta$.

Now (4-6) and (4-7) form a system of linear inequalities of the form of (4-4). The unknown is now v instead of x, $(Ax^j - b)$ corresponds to $a_j$, and for (4-7), one has $-\varepsilon$ in place of $b_i$.

Consider the following direct search algorithm, (DSP) for optimal surrogate multipliers.

Step 0: Set $k = 0$, $\nu^*(D_S) = -\infty$, $\lambda = 1$. Choose $v^* = v^k \geq 0$, $\varepsilon > 0$ and IT arbitrarily.

Step 1: Solve $(P^{v^k})$ with solution $x^k$. If $cx^k > \nu^*(D_S)$, set $\nu^*(D_S) = cx^k$, $v^* = v^k$. Let $y^k = (Ax^k - b)$. If $k \geq$ IT, stop, and accept current incumbent as an optimal or near optimal surrogate multiplier.

Step 2: Let $v^{k+1} = v^k + ty^k$, where $t = \dfrac{-\lambda[v^k y^k - \varepsilon]}{|y^k|^2}$.

If $v_i^{k+1} < 0$, set $v_i^{k+1} = 0$ for all such i. Replace k by k + 1 and go to Step 1.

Note that the most superoptimal x which is still feasible for $(P^{v^k})$ is generated at each iteration so that one of the violated constraints of (4-5) is found at Step 1, and a corresponding iteration of (LR) occurs at Step 2. If one or more of the constraints (4-6) is violated, say $v_\ell = \delta < 0$, linear relaxation would set $v_\ell = 0$. This is seen by considering (4-4) with $a_{ij} = 1$, $b_i = 0$ and $\lambda = 1$, resulting in $t = -1[\delta]/1^2$ and $v_\ell^{k+1} = v_\ell^k + t \cdot 1 = \delta - \delta = 0$.

Being able to choose the violated constraint arbitrarily in (LR) makes it possible to first choose the most superoptimal x still feasible in $(P^{v^k})$ and then choose any violated nonnegativity constraint on v.

As in the Benders' type procedure outlined earlier, it is possible to stop short in solving $(P^{v^k})$ with an x which is feasible in $(P^{v^k})$ and which has solution value less than or equal to $\nu^*(D_S)$. This will produce an arbitrarily chosen violated constraint, which is all that is required.

If all T of the superoptimal x's are made infeasible, i.e., $v(Ax^i - b) > 0$ $i = 1,2,\ldots,T$, but not necessarily $\geq \varepsilon$, then an optimal surrogate multiplier has been found and the procedure (DSP) may be terminated. That is, it is not necessary to solve for $v(Ax^i - b) \underline{\geq \varepsilon}$ and in fact a superoptimal x, say $x^\ell$, will not be generated in Step 1 if $v(Ax^j - b) > 0$ whether or not $v(Ax^\ell - b) \geq \varepsilon$, $\varepsilon > 0$. So in this sense (DSP) finds an optimal solution faster, or may terminate earlier than suggested by the Benders' type procedure.

For $\lambda = 1$, all cases of Theorem 4.4 indicate finite convergence or convergence to a point on the boundary of H = {Set of solutions to (4-6), (4-7)}. Here, however, it is only necessary to obtain a point in $H^1$ = {Set of solutions to (4-5), (4-6)}. When one is 'close to' the boundary of H one has a point in $H^1$.

An important question not discussed above is how does one know if an optimal solution has been found; i.e., all superoptimal $x^i$, $i = 1,2,\ldots,T$ have been made infeasible. Of course, not knowing these $x^i$ beforehand makes it impossible to be sure, so a heuristic stopping rule must be employed. In (DSP) as outlined above, the quantity IT places a maximum number of iterations on the algorithm. If IT is large enough for a given problem, then the incumbent solution will be optimal at termination. Since the context here is to seek bounds in a branch-and-bound procedure, a perhaps much better stopping rule would be the failure to improve on the incumbent solution value for ($D_S$) after a fixed number of iterations. If the directions for changes in v are 'close to' ascent directions, then such a stopping rule should be appropriate.

Note that if $\varepsilon = 0$, the direct search procedure would not be guaranteed to solve (4-5) and hence ($D_S$). $\varepsilon = 0$ corresponds to the subgradient-type procedure mentioned in the first section of this chapter.

## Linear Relaxation Master Problem Procedure, (LRMP)

The direct search procedure (DSP) developed in the previous section has two obvious shortcomings. First, no use is made of previously generated solutions in the iterative procedure, implying no guarantee that previously generated x's will not be regenerated before a new x is found. Also, the stopping criterion is perhaps too arbitrary and may lead to considerably suboptimal solutions for ($D_S$). By combining the ideas of the Benders' type procedure and the direct search procedure, both of these weaknesses may be overcome.

Consider using the direct search procedure while keeping a list of all x's previously generated in a master problem. Then before a new v is determined and a new relaxation, ($P^V$) is solved, each x can be made infeasible in ($P^V$). The list is checked sequentially taking steps in v corresponding to the linear relaxation or direct search procedure until all x's present in the list satisfy $v(Ax - b) > 0$. If it is impossible to make all of these x's infeasible then the linear relaxation procedure will not terminate. An upper bound based on computational experience can be placed on the number of iterations for solving each master problem. When it is exceeded, the present incumbent solution is accepted as optimal. As shown in Chapter VI, a relatively small upper bound yields very nearly optimal solutions for all problems tested.

Details of such a procedure, denoted (LRMP), are as follows:

<u>Step 0</u>: Set $k = 1$, $\nu^*(D_S) = -\infty$. Choose ITMAX, $v^* = v^k \geq 0$, $\varepsilon > 0$ arbitrarily.

<u>Step 1</u>: Solve $(P^{v^k})$ with solution $x^k$. If $cx^k > \nu^*(D_S)$, set $\nu^*(D_S) = cx^k$, $v^* = v^k$. Let $y^k = (Ax^k - b)$. If $y^k \leq 0$, stop, $x^k$ is optimal for (P).

<u>Step 2</u>: Consider the system

$$uy^i \qquad i = 1,2,\ldots,k \tag{4-8}$$

Set $u = v^k$, IT = 0

<u>Step 3</u>: Sequentially check (4-6) to see if $uy^i > 0$. If $uy^p < 0$, set $u = u + ty^p$, $t = \frac{-[uy^p - \varepsilon]}{|y^p|^2}$. If $u_i < 0$, set $u_i = 0$, $i = 1,2,\ldots,m$. Let IT = IT + 1. If IT > ITMAX, stop. Otherwise repeat Step 3 until

$uy^i > 0$, $i = 1,2,\ldots,k$. Set $v^{k+1} = u$. Replace k by k + 1 and go to Step 1.

As in the previous two algorithms one may stop short in solving Step 1 if an x feasible to $(P^{v^k})$ is found with value less than or equal to $\nu^*(D_S)$.

Observe that the only additional activity in (LRMP) not in (DSP) is the maintenance of a master list of x's. On the other hand, there is considerable gain from eliminating the need to regenerate x's by solving $(P^v)$. Thus, (LRMP) would appear much more attractive computationally, and (DSP) will not be further pursued. Also note that another way to view the (LRMP) procedure is to consider it a Benders' type procedure (BTP) in which only a feasible solution to $(LP)_k$ (i.e., one with $z > 0$) is found at each iteration. The relation between the (BTP) and the (LRMP) algorithms is further investigated in Chapter VI.

# CHAPTER V

# SURROGATE DUALITY IN A BRANCH-AND-BOUND PROCEDURE

Throughout this dissertation it has been assumed that the surrogate dual would be used in providing bounds for a branch-and-bound procedure. In this chapter, the intent is to more fully develop the inner play between the surrogate dual and the primal in a branch-and-bound procedure. When the two are considered conjunctively a number of advantages are gained beyond the providing of a bound by the surrogate dual. A number of general observations will first be made with respect to the surrogate dual. Then specific issues or parts of the general branch-and-bound procedure will be developed in their relationship with the surrogate dual.

## Including Cost in the Surrogate

At any point in a branch-and-bound enumeration of (P) the only solutions of interest are those which can improve on the incumbent solution value $\nu^*(P)$. In (20) Geoffrion shows the value of including the implied constraint $cx < \nu^*(P)$ into the surrogate linear combination with the constraints $Ax - b \leq 0$. However, Geoffrion's surrogate formulations use a different form of the surrogate dual than the one presented in this dissertation. For surrogate duals as defined here, the following theorem shows that the optimal surrogate multiplier on $(cx - \nu^*(P)) < 0$ is zero for any value of $\nu^*(P)$ which is an upper bound on $\nu(P)$. Hence there is no value in including cost in the surrogate constraints.

Theorem 5.1

Consider the surrogate relaxation

$$(P^V(w)) \text{ Min } cx \text{ subject to } u(Ax - b) + w(cx - \nu^*(P)) \leq 0$$

where $\nu^*(P) \geq \nu(P)$. Then $w = 0$ in some optimal solution to

$$(D_S(w)) \qquad \underset{u\geq 0, w\geq 0}{\text{Max}} \quad \nu(P^{u,w})$$

Proof:

For any $u \geq 0$ and $w \geq 0$, $x$ optimal in $(P^u(w))$ implies $cx \leq \nu(P)$ and thus $cx \leq \nu^*(P)$. Thus, for such $x$, $w(cx - \nu^*(P)) \leq 0$. It follows that any $x$ optimal in $(P^u(0))$ is feasible in $(P^u(w))$ and so $\nu(P^u(0)) \geq \nu(P^u(w))$. Thus, an optimal solution to $(D_S(w))$ must exist which has $w = 0$.

Q.E.D.

## Surrogate Subproblems

Recall the surrogate relaxation of (P) for any $v \geq 0$ is

$$\begin{aligned} (P^V) \qquad & \text{Min } cx \\ & \text{s.t. } \; v(Ax - b) \leq 0 \\ & \qquad x \in S \end{aligned}$$

Note that $(P^V)$ is itself an integer linear programming problem with a single explicit constraint $v(Ax - b) \leq 0$. Thus it is a knapsack problem with a set of side constraints, $x \in S$. A number of solution techniques have appeared in the literature for the case of $S = \{x: x \geq 0, x \text{ bounded above}\}$. Basically these can be divided into two categories, dynamic programming procedures and branch-and-bound or implicit enumeration procedures. For a good review of the dynamic programming procedures, see Garfinkel and Nemhauser (16). It will soon become evident that a branch-and-bound procedure will be more convenient in solving $(P^V)$, because the relation between the primal and knapsack branch-and-bounds can be exploited. Moreover, Cabot (9), Kolesar (33), Fayard and Plateau (12), and Greenberg and Hegerich (27), among others, have developed branch-and-bound procedures which proved computationally more efficient than the dynamic programming approaches. Finally, recall that in Chapter IV it was shown that each surrogate relaxation need not be solved optimally, but a feasible solution with value less than or equal to the surrogate dual incumbent was sufficient to terminate solving $(P^V)$. By solving $(P^V)$ via a branch-and-bound procedure such solutions will be shown to be obtainable and require no extra computations. In a dynamic programming procedure, however, a feasible solution is generally not available until optimality is obtained so that $(P^V)$ must be solved completely. For these reasons and more to become apparent upon seeing the inner play with (P), the remainder of this chapter assumes surrogate relaxation subproblems are best solved via a branch-and-bound procedure. Such a procedure is defined in the next sub-section for later reference.

Solving Relaxations as Knapsack Problems

The knapsack branch-and-bound procedure for solving surrogate relaxations follows the flowchart given in Figure 1 of Chapter I. A number of observations may be made, however, which are specific to the knapsack problem. First, note that for $S = \{x: \ x \geq 0, \ x \text{ bounded above}\}$ the knapsack problem $(P^v)$ may be written in the following form.

$$\text{(KNP)} \qquad \begin{aligned} \text{Max} \quad & -cx \\ \text{s.t.} \quad & \sum_j w_j x_j \leq w_0 \qquad (5\text{-}1) \\ & 0 \leq x_j \leq u_j \end{aligned}$$

where $u_j$ is an upper bound on $x_j$, $w_j = (vA)_j$, and $w_0 = vb$. Without loss of generality it may be assumed that $-c_j$, $w_j$, and $w_0$ are positive in all relaxations requiring nontrivial solutions. This is seen by the following argument. If $w_0 < 0$, multiply (5-1) by a minus one and let $w_j = -w_j$. Now consider the following cases. If $w_j \leq 0$ and $-c_j \geq 0$, then clearly $x_j = u_j$ in an optimal solution to (KNP) since the objective function will increase and none of the resource w will be used. If $w_j \geq 0$ and $-c_j \leq 0$, then an optimal solution can have $x_j = 0$ since $x_j > 0$ will reduce $\nu$(KNP) and only use up the resource $w_0$. If both $w_j$ and $-c_j$ are less than zero, replace $x_j$ by $u_j - x_j'$. Then $-c_j(u_j - x_j') = -cu_j + cx_j'$ where $-cu_j$ is a constant so that the maximization involves only $c_jx_j'$ with $c_j > 0$. Also $w_j(u_j - x_j) = w_ju_j - w_jx_j$ so that $-w_j > 0$ and $w_ju_j < 0$ may be taken to the right hand side and added to $w_0$.

The following observations precede an outline of a branch-and-bound procedure to solve (KNP). Note that the linear programming relaxation of (KNP), denoted by $(\overline{\text{KNP}})$, is solved trivially. Simply order the variables in non-decreasing "bang-for-buck" or $-c_j/w_j$ ratio. Assume this has been done; now write $-c_1/w_1 \geq -c_2/w_2 \geq \dots \geq -c_n/w_n$. Sequentially set $x_j = u_j$ until (5-1) is violated at, say, $j = r$. The solution to $(\overline{\text{KNP}})$ is then $x_j = u_j$, $j = 1,2,\dots,r-1$ and $x_r = (w_0 - \sum_{j=1}^{r-1} w_ju_j)/w_r$. Of course it is possible that all of the $x_j = u_j$ if w is greater than $\sum_{j=1}^{n} w_ju_j$. The same solution procedure holds for any knapsack candidate problem in a branch-and-bound procedure so that $\nu(\overline{\text{KNP}})$ may be used in Step 2 of Figure 1 to provide an easily obtained bound on the candidate problem KNP(T). Note that in the case of S not completely discrete, if $x_r$ is a continuous variable the optimal solution to $(\overline{\text{KNP}})$ is an optimal feasible solution for (KNP). If $x_r$ is not continuous then setting $x_r = 0$ will provide a feasible solution to (KNP) at any step in the branch-and-bound procedure. Figure 7 provides a flowchart of a branch-and-bound procedure for solving (KNP), assuming that

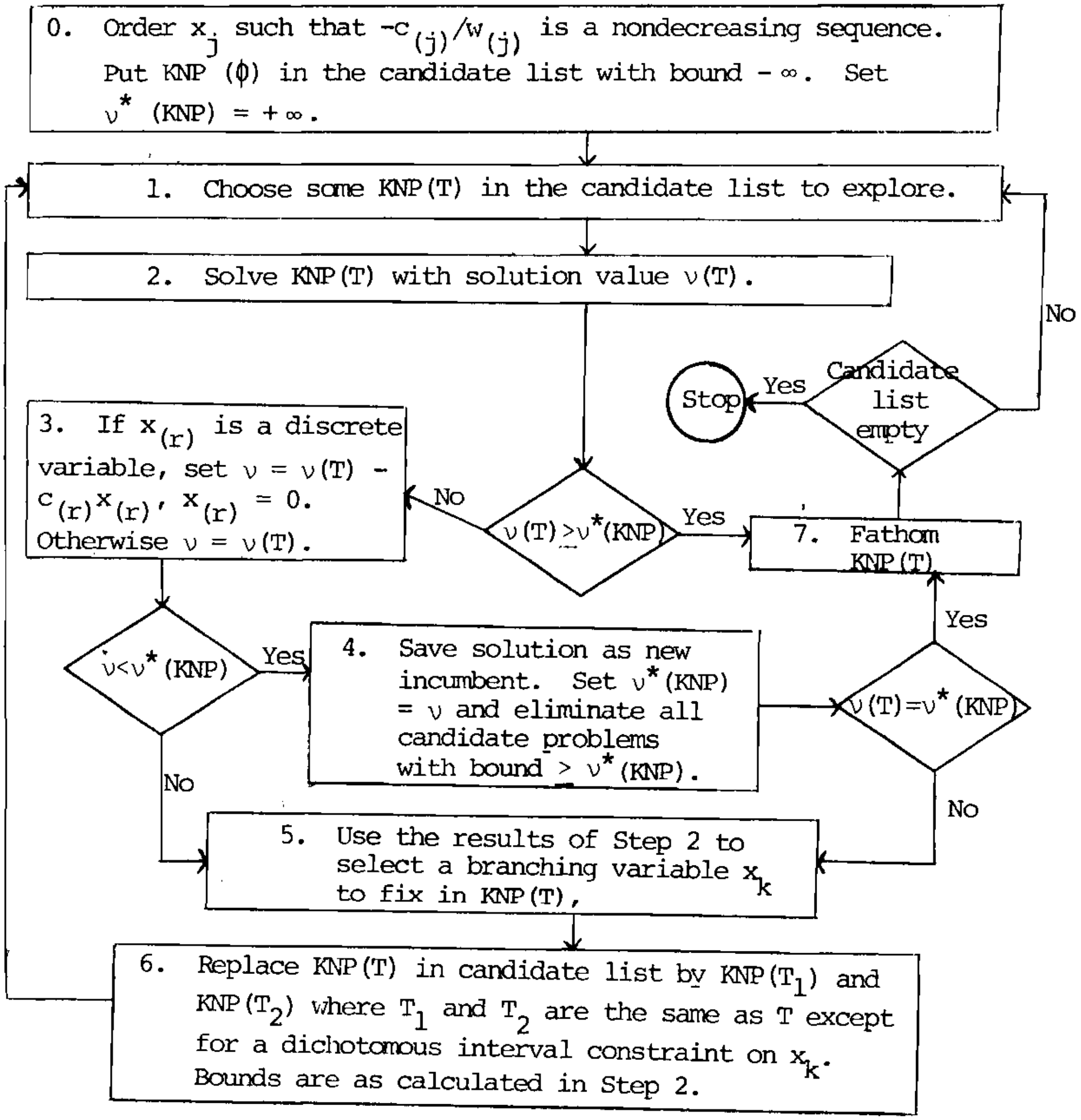

0. Order $x_j$ such that $-c_{(j)}/w_{(j)}$ is a nondecreasing sequence. Put KNP ($\phi$) in the candidate list with bound $-\infty$. Set $\nu^*$ (KNP) $= +\infty$.
1. Choose some KNP(T) in the candidate list to explore.
2. Solve KNP(T) with solution value $\nu(T)$.
$\nu(T) \geq \nu^*(KNP)$
No
Yes
3. If $x_{(r)}$ is a discrete variable, set $\nu = \nu(T) - c_{(r)}x_{(r)}$, $x_{(r)} = 0$. Otherwise $\nu = \nu(T)$.
$\nu < \nu^*$ (KNP)
Yes
No
4. Save solution as new incumbent. Set $\nu^*$(KNP) $= \nu$ and eliminate all candidate problems with bound $\geq \nu^*$(KNP).
$\nu(T) = \nu^*$ (KNP)
Yes
No
5. Use the results of Step 2 to select a branching variable $x_k$ to fix in KNP(T),
6. Replace KNP(T) in candidate list by KNP($T_1$) and KNP($T_2$) where $T_1$ and $T_2$ are the same as T except for a dichotomous interval constraint on $x_k$. Bounds are as calculated in Step 2.
7. Fathom KNP(T)
Candidate list empty
Yes
Stop
No

Figure 7. Branch-and-Bound Procedure for (KNP)

the proper modifications have been made so that $-c_j$, $w_j$, and $w_0$ are all positive. To be consistent with later discussions the problem (KNP) is also assumed to have been converted to a minimization on cx.

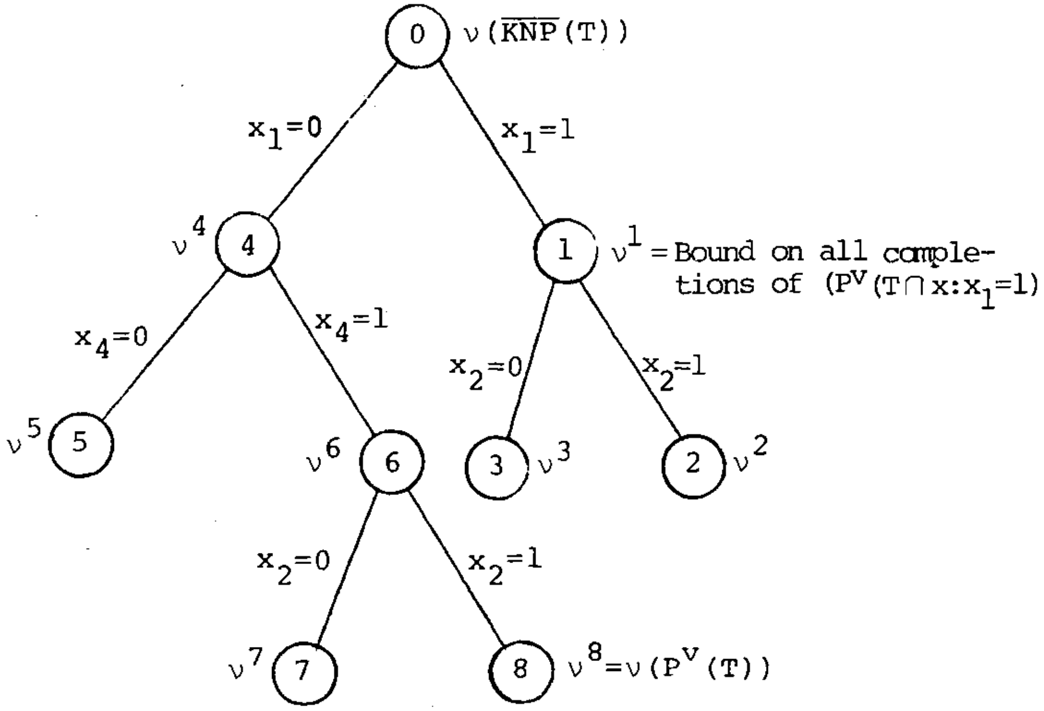
Role of the Primal Incumbent in $(P^V)$

Recall that $\nu(D_S(T))$ is being employed as a bound for some candidate problem P(T) in the primal branch-and-bound procedure. However, $\nu(P^V(T))$ is a valid bound in P(T) for all $v \geq 0$, not just the v which maximizes $\nu(P^V(T))$. Thus $(D_S(T))$ need not be solved optimally if $\nu(P^V(T))$, for some v used on the way to solving $(D_S(T))$, is sufficient to fathom P(T), i.e., $\nu(P^V(T)) \geq \nu^*(P)$. Conversely, the value of the incumbent in the primal, $\nu^*(P)$, may be used as a bound in solving $(P^V)$. That is, if no completion of a candidate problem in $(P^V)$ can produce a solution with value less than $\nu^*(P)$, that candidate problem in $(P^V)$ may be fathomed. If all candidate problems in the knapsack $(P^V)$ fail to produce a solution with value less than $\nu^*(P)$, then it can be concluded that $\nu(P^V(T)) \geq \nu^*(P)$ so that the candidate problem P(T) may be fathomed in the primal. This is just one of the important interactions of the primal and surrogate relaxation branch-and-bound procedures. The next section discusses the role of the dual surrogate in providing conditional bounds and a choice of branching variables in the primal branch-and-bound procedure.

## Conditional Bounds and Branching Variables

The rationale for the interaction between the two branch-and-bound procedures with respect to conditional bounds and branching rules can perhaps best be understood via a 0-1 integer programming example. Later a procedure for the general case will be presented. Consider Figure 8 which presents a branch-and-bound tree for the problem $(P^V(T))$ where $P(T)$ is a given candidate problem from the primal tree. This tree may result from the application of the algorithm described earlier and presented in Figure 7. The optimal solution to $(P^V(T))$ is found at node 8 with value $\nu^8$. Since the full tree is shown and an optimal solution has been found, $\nu^1, \nu^2, \ldots, \nu^7$ must all be $\geq \nu^8$.

Now a number of important observations may be made. If $\nu^8$ is accepted as the optimal solution value for $(D_S(T))$ and the candidate problem $P(T)$ is not able to be fathomed $(\nu^8 < \nu^*(P))$ then a branching variable must be chosen and a conditional bound computed for each of the two new nodes created in the primal tree. Note that if $x_1$ is chosen as the branching variable, then a valid bound on any solution to $(P(T \cap \{x\colon\ x_1 = 1\}))$ is given by $\hat{\nu} = \text{Min}\ (\nu^2, \nu^3)$. Also since $\nu^8$ was the optimal value of $(P^V(T))$, $\hat{\nu} \leq \nu^8$. So even though $\nu^8 < \nu^*(P)$, it is possible that the bound $\hat{\nu} \geq \nu^*(P)$ so that no completion of $P(T \cap \{x\colon\ x_1 = 1\})$ will ever need be considered. Thus $x_1$ is a good candidate for a branching variable in the primal tree. Note that a conditional bound



$S = \{x: 0 \le x \le 1,\ x \text{ integer}\}$

Figure 8. Example of a Branch-and-Bound Tree for $(P^V(T))$

for branching on $x_2$ may be taken as $\mathrm{Min}(\nu^5, \nu^7, \nu^3)$ for $x_2 = 0$ and $\mathrm{Min}(\nu^5, \nu^8, \nu^2)$ for $x_2 = 1$. All of the end nodes for which $x_2$ is a free variable must be included (hence $\nu^5$) in calculating these bounds. $x_1$ is the only variable for which no free end nodes may exist, so it is chosen as the branching variable.

What is required to implement the branching procedure suggested above is the saving of the minimum value or bound on the end nodes for each of the two sides of the tree defined by the first branching variable. An end node may be recognized as one from which a fathoming occurs. Thus before fathoming it is necessary to determine which side of the tree one is on, check to see if the bound on that node is less than the saved bound for that side of the tree, and if necessary, replace that saved bound. Then after solving $(P^V(T))$ one will have $\nu(P^V(T))$ as the bound on one side ($\nu^8$ in Figure 8), and a bound saved for branching on the nonoptimal side of the tree ($\mathrm{Min}(\nu^2, \nu^3)$ in Figure 8). Formally let

$x_{(B1)}$ = initial branching variable

$x^*_{(B1)}$ = optimal value of $x_{(B1)}$ in $P^V(T)$

$T_1$, $T_2$ = dichotomous constraint sets on which $x_{(B1)}$ has been branched

$\nu^{(i)}$ = minimum value of all end nodes for completions of $(P^V(T \cap T_i))$, $i = 1, 2$

$\nu^{(f)}$ = bound on the node which is about to be fathomed in $P^V(T)$

When ready to fathom in a branch-and-bound procedure for $(P^V(T))$ determine whether $x_{(B1)}$ is contained in $T_1$ or $T_2$. If $x_{(B1)} \varepsilon T_i$ and $\nu^{(f)} < \nu^{(i)}$ then replace $\nu^{(i)}$ by $\nu^{(f)}$. In any case, continue by fathoming. Note that when finished, either $\nu^{(1)}$ or $\nu^{(2)}$ will be equal to $\nu(P^V(T))$ and the other will be a valid bound on any completion of $P(T \cap x_{(B1)} \varepsilon T_i)$ where $x_{(B1)}{}^* \not\varepsilon T_i$.

Recall that any search procedure for optimal surrogate multipliers cannot be guaranteed monotonic in solution value. Thus just as an incumbent v is stored for $(D_S)$, an incumbent $x_{(B1)}$, $\nu^{(1)}$, and $\nu^{(2)}$ are also stored.

An outline for solving $(P^V)$, using all of the discustion above and based on the algorithm for solving (KNP), can now be presented in Figure 9. (KNP) is the equivalent of $(P^V)$.

## Interaction of the Surrogate Search Master Problems

The two surrogate dual algorithms which appear most promising as discussed in Chapter IV both keep a list of the x's generated by each surrogate relaxation and solve a master problem involving these x's to obtain a new surrogate multiplier v. These master problems, one for each candidate problem in a primal branch-and-bound procedure, may be seen to interact in such a way as to save a great deal of time in

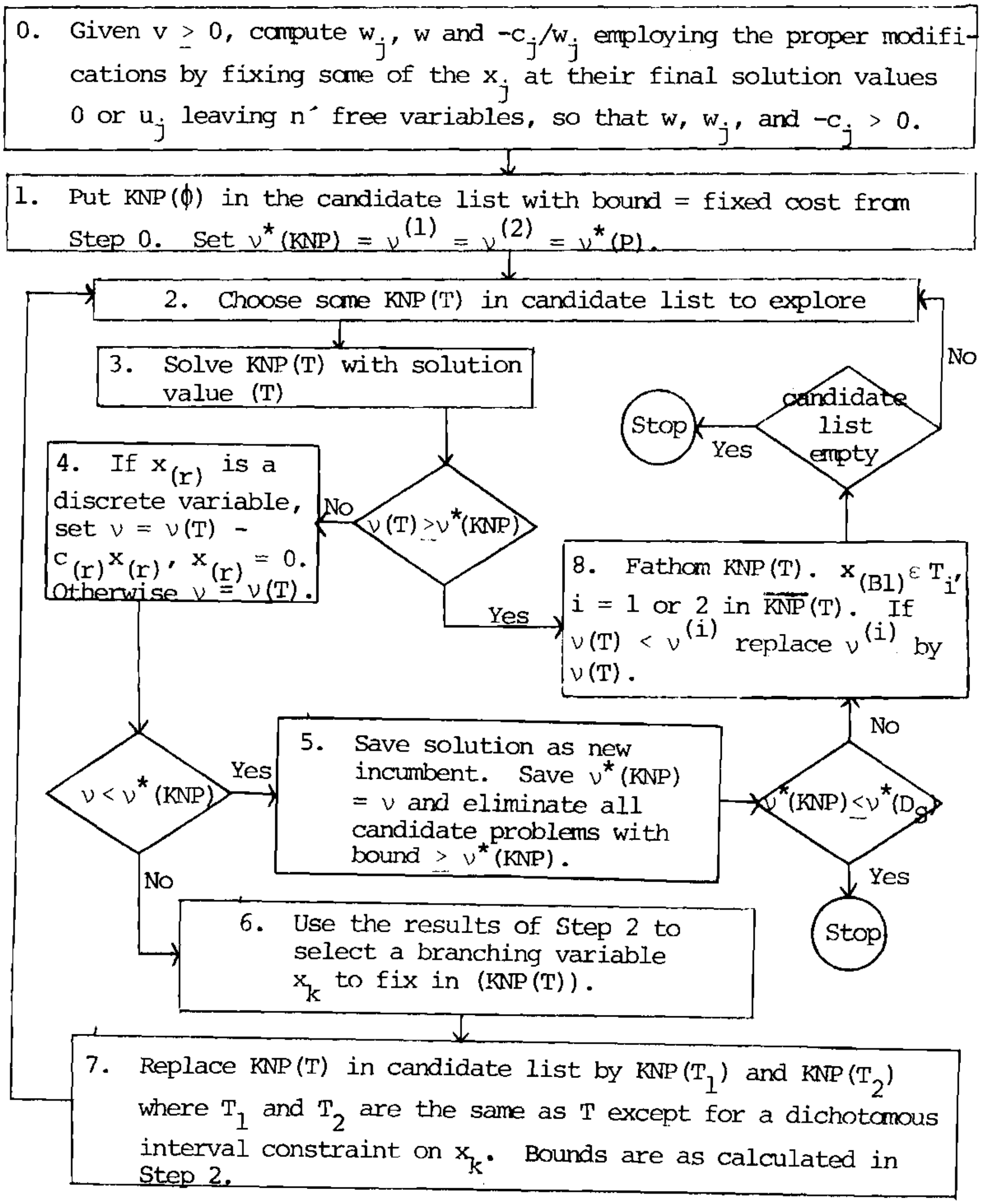

0. Given $v \geq 0$, compute $w_j$, w and $-c_j/w_j$ employing the proper modifications by fixing some of the $x_j$ at their final solution values 0 or $u_j$ leaving n´ free variables, so that w, $w_j$, and $-c_j > 0$.
1. Put KNP($\phi$) in the candidate list with bound = fixed cost from Step 0. Set $\nu^*(\text{KNP}) = \nu^{(1)} = \nu^{(2)} = \nu^*(P)$.
2. Choose some KNP(T) in candidate list to explore
3. Solve KNP(T) with solution value (T)
$\nu(T) \geq \nu^*(\text{KNP})$
No
Yes
4. If $x_{(r)}$ is a discrete variable, set $\nu = \nu(T) - c_{(r)}x_{(r)}$, $x_{(r)} = 0$. Otherwise $\nu = \nu(T)$.
$\nu < \nu^*(\text{KNP})$
Yes
No
5. Save solution as new incumbent. Save $\nu^*(\text{KNP}) = \nu$ and eliminate all candidate problems with bound $\geq \nu^*(\text{KNP})$.
6. Use the results of Step 2 to select a branching variable $x_k$ to fix in (KNP(T)).
7. Replace KNP(T) in candidate list by KNP($T_1$) and KNP($T_2$) where $T_1$ and $T_2$ are the same as T except for a dichotomous interval constraint on $x_k$. Bounds are as calculated in Step 2.
8. Fathom KNP(T). $x_{(B1)} \in T_i$, i = 1 or 2 in $\overline{\text{KNP}}$(T). If $\nu(T) < \nu^{(i)}$ replace $\nu^{(i)}$ by $\nu(T)$.
$\nu^*(\text{KNP}) \leq \nu^*(D_g)$
No
Yes
Stop
candidate list empty
Yes
No
Stop

Figure 9. Flow Chart of a Branch-and-Bound Procedure for $(P^V)$

solving $(D_S)$ at any proceeding node in a primal tree.

Consider the primal branch-and-bound tree shown in Figure 10 for a 0-1 integer linear programming problem. Assume that a master problem or at least a list of the x's generated in solving $(D_S)$ at node 0 has been kept and it is now time to branch on $x_1$. Scan the master problem at node 0 and place all $x^i$, $i = 1,2,\ldots,k$ which satisfy $x_1^i = 0$ in a new master problem for solving $(D_S(T))$ at node 1 of the primal tree. All solutions $x \varepsilon S$, $x_1 = 1$ such that $cx < \nu(D_S(0))$ have been made infeasible by the optimal surrogate multiplier at node 0. If one is to improve on $\nu(D_S(0))$ as a bound after

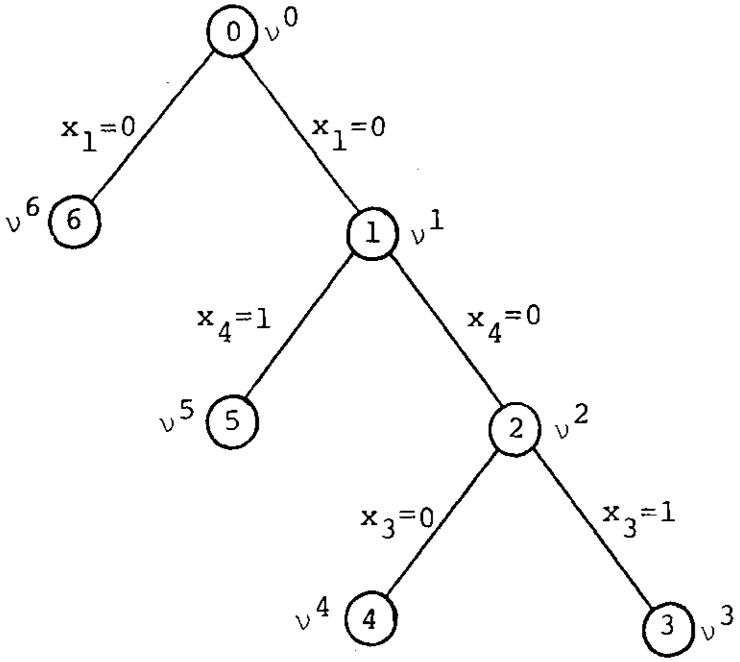


Figure 10. Example of a Primal Branch-and-Bound Tree

branching on $x_1$, then all of these x's must be included in the new master problem at node 1. This is valid since the candidate problem at node 1 is a more constrained version of (P), and all the x's put in the master problem satisfy this extra constraint. None of these x's are feasible to (P) or else the branching would not have occurred. This procedure may be continued as follows. In solving $(D_S(T))$ at node 1, possibly more x's are generated. When branching to node 2, all such x's in the master problem at node 1 with $x_4 = 0$ may be put in the master problem to begin solving the surrogate dual at node 2. Any candidate problem may be chosen to be explored next in a branch-and-bound procedure and a number of strategies have been suggested. The "last-in first-out" or LIFO procedure always chooses the most recently added member of the candidate list to explore. Referring to Figure 10, the nodes have been numbered in the order in which a LIFO procedure would explore them. Hence the order of branching is from node 0 to node 1 to node 2 and to node 3 at which time node 3 is fathomed, either because the incumbent solution to (P) was exceeded, a feasible solution was obtained, or it was determined that $x_1 = 0$, $x_4 = 0$ and $x_3 = 1$ precluded any feasible solution to (P). Thus 'back-tracking' goes to node 4 which is also fathomed, leading back to node 5. In a LIFO procedure note that there are never more than two nodes at any given level of the tree, a level being defined by the number of fixed variables or extra constraints on (P). For instance in Figure 10, the fathoming of nodes 2 and 5 must occur before node 6 is chosen as the node from which to branch. In large integer programming problems, where many x's from previous surrogate master problems are to be stored, storage can be a main concern and it is minimized by using the LIFO branching procedure.

The master problem interactions can be shown to be very efficient in terms of a LIFO branching procedure for (P). Again consider Figure 10 and the following use of a 'current table' and a 'save table.' At node 0, the master problem consists of the following x's, say for n = dimension of x = 5.

| | | | | | |
|---|---|---|---|---|---|
| $x^1$ | 0 | 0 | 1 | 1 | 1 |
| $x^2$ | 1 | 0 | 1 | 0 | 1 |
| $x^3$ | 0 | 1 | 1 | 0 | 0 |
| $x^4$ | 1 | 1 | 0 | 1 | 0 |

Branching takes place to node 1. Those x's which have $x_1 = 0$ ($x^1$ and $x^3$) are placed in the 'current table' for the 'current' or next-to-be-explored candidate problem. The other x's ($x^2$ and $x^4$) are placed in the 'save table' and it is noted that at level 1 of the tree, the next open slot in the save table is in row 3. Node 1 is now explored and some new x's are generated and put in the current table which becomes

| | | | | | |
|---|---|---|---|---|---|
| $x^1$ | 0 | 0 | 1 | 1 | 1 |
| $x^3$ | 0 | 1 | 1 | 0 | 0 |
| $x^5$ | 0 | 0 | 1 | 0 | 1 |
| $x^6$ | 0 | 1 | 0 | 1 | 1 |

Now it is time to branch to node 2, so those x's which have $x_4 = 0$ remain in the current table, i.e., $x^3$ and $x^5$. $x^1$ and $x^6$ are placed in the save table and it is noted that the next open slot in the save table at level 2 of the tree is 5.

The current table is now

| | | | | | |
|---|---|---|---|---|---|
| $x^3$ | 0 | 1 | 1 | 0 | 0 |
| $x^5$ | 0 | 0 | 1 | 0 | 1 |

and the save table is

| | | | | | | |
|---|---|---|---|---|---|---|
| $x^2$ | 1 | 0 | 1 | 0 | 1 | L = 1 |
| $x^4$ | 1 | 1 | 0 | 1 | 0 | |
| $x^1$ | 0 | 0 | 1 | 1 | 1 | L = 2 |
| $x^6$ | 0 | 1 | 0 | 1 | 1 | |

Assume that, in contrast to Figure 10, fathoming occurs at node 2, possibly after generating some more x's. Now the current table can be cleared since it is no longer necessary to explore any candidate problem with $x_1 = 0$ and $x_4 = 0$. In fact these x's will never be generated or needed again, since either $x_1$ or $x_4$ or both will always be fixed at 1 in any future candidate problems. Now the LIFO branching procedure goes to node 5 with $x_1 = 0$ and $x_4 = 1$. But some of these x's are stored in the save table from the last slot in the save table (5 - 1 = 4) back to the next available slot stored after the previous level, level 1, which is slot number 3. These are put in the current table which is now

$$\begin{array}{cccccc} x^1 & 0 & 0 & 1 & 1 & 1 \\ x^6 & 0 & 1 & 0 & 1 & 1 \end{array}$$

and the save table is now

$$\begin{array}{cccccc} x^2 & 1 & 0 & 1 & 0 & 1 \\ x^4 & 1 & 1 & 0 & 1 & 0 \end{array}$$

Possibly more x's are generated at node 5 and placed in the current table. A fathoming then occurs at node 5 and a 'backtracking' takes place to node 6. The 'other side' of level 2 has been explored so the backtracking must be to level 1. The current table is again cleared and the elements in the save table from the last slot to the first slot for level 1 savings (slot 1) are placed in the current table. The procedure continues, with only two lists being necessary to easily store, update, and use all of the x's generated by solving surrogate relaxations throughout the primal branch-and-bound procedure. Note that no x's will be regenerated using this procedure, and again that once the current candidate problem is fathomed those x's may be taken out of storage. The following is a formal outline for branching and fathoming while employing the current and save tables for a general integer linear programming problem. Let

L = current level in primal branch-and-bound tree

$T^1_{(L)}$, $T^2_{(L)}$ = two new candidate problems created at level L, $T^1_{(L)}$ is candidate problem chosen to explore next

SAVBND (L) = bound saved for candidate problem at level L which is not being explored next

NXSV = next available slot of the save table

NXCR = next available slot of the current table

NSV(L) = next available slot of the save table at level L in the primal tree

$\nu^*(P)$ = incumbent solution value to (P)

Branching:

If SAVBND (L) $< \nu^*(P)$ place all x's from the current table satisfying $x \varepsilon T^2_{(L)}$ into the save table, updating NXSV. In any case, let NSV (L) = NXSV and remove all x's satisfying $x \varepsilon T^2_{(L)}$ from the current table, closing up the current table and updating NXCR. Determining if $x \varepsilon T^2_{(L)}$ is done simply by checking the single component of x upon which the branching occurred.

Fathoming:

Clear the current table by setting NXCR = 1. (If

$T^2_{(L)}$ has already been explored, SAVBND(L) = $+\infty$.) If SAVBND(L) $\geq \nu^*(P)$ replace NXSV by NSV(L - 1) and L by L - 1 until a candidate problem is found to explore. Place rows NSV(L - 1) to NSV(L) - 1 from the save table into the current table. Update NXCR.

After branching or fathoming more x's are generated while solving $(D_S(T))$ and placed in the current table until it is time to branch or fathom again.

# CHAPTER VI

# COMPUTATIONAL ANALYSIS

The theoretical developments of Chapter III suggested that the value of the surrogate dual would most likely provide a better bound on an integer programming problem than the value of the lagrangian dual. In Chapter IV a number of surrogate multiplier search procedures were presented, and Chapter V developed the role and application of surrogate duality in a branch-and-bound procedure. Here, these theoretical investigations will be supported by empirical evidence on a set of randomly generated 0-1 integer programming test problems. After a discussion of the test problems employed, the surrogate multiplier search procedures are analyzed and one is selected for use in a primal branch-and-bound procedure. The question of smaller primal-dual gaps and the efficiency of the 'save table,' 'current table' concepts of Chapter V are then analyzed. All of the algorithms presented were implemented in FORTRAN on the Georgia Institute of Technology's CDC Cyber 74.

## Random Problem Generation

The following class of problems was used to demonstrate the algorithms developed and to address the theoretical result of gap improvement.

$$\begin{aligned} \text{Min} \quad & cx \\ \text{s.t.} \quad & Ax \leq b \\ & x \in S \end{aligned}$$

where $S = \{x: \ 0 \leq x \leq 1, \text{ integer}\}; \ A, b, -c \geq 0$

Note that the above problem has the lagrangian integrality property, but as noted in Chapter III, many problems of interest do.

The test problems were randomly generated with the following characteristics. The cost vector c was integer and uniformly distributed over the range -10 to -100. A density level (per cent of nonzero entries) was set for the constraint matrix A and nonzero cells were integer and uniformly distributed between 1 and 10. The resource vector b was chosen integer and uniformly distributed between 1/4 and 3/4 times the expected row sum of the constraint matrix A. Thus one might expect about 1/2 of the variables to be at a value of one in an optimal solution to the primal problem. Of course, $x = 0$ is a feasible integer solution. The problem sizes used were 5 x 10, 10 x 20, and 15 x 30, i.e., $n = 2 \cdot m$. Within this general framework problems were characterized by an integer seed given to a pseudo-random number generator. A FORTRAN code of the random problem generator is presented in the Appendix.

## Benders' Type Procedure

An outline of the Benders' type procedure derived independently by Banerjee (4) was presented in Chapter IV, and two possible refinements were discussed. Normalizing the vector of infeasibility $(Ax^j - b)$, as discussed in Chapter IV, can lead to faster convergence and fewer regenerations when employing constraint dropping in the master problems. All of the computational experience reported here for (BTP) employs this normalization technique. The main interest in this section is to demonstrate the effect of the second refinement discussed in Chapter IV; that is, the ability to stop the solution procedure for $(P^V)$ short of optimality. This may be done whenever an x which is feasible in $(P^u)$ is found which has solution value less than or equal to the present surrogate dual incumbent solution value. In Chapter V, it was shown that the branch-and-bound procedure suggested for solving the knapsack subproblems, $(P^u)$, could easily obtain such solutions.

Before discussing the effect of using this early stopping of subproblems or knapsacks in (BTP), it is first necessary to clarify how (BTP) was solved for the test problems generated. The basic statement of the algorithm is given in Chapter IV, with specific details discussed below. In Step 1, the subproblems are solved as shown in Figure 9 (in Chapter V) with $\nu^*(P) = +\infty$. The branching variable is always chosen to be the fractional variable, $x_\ell$, resulting from solving $(\overline{KNP}(T))$. This branching rule for solving 0-1 knapsack problems was suggested by Greenberg and Hegerich (27) who demonstrated its computational superiority over various other branching rules. In Step 2, the linear programming problem $(LP)_k$ is solved efficiently by using the dual simplex procedure starting from the previous tableau from $(LP)_{k-1}$. Computational experience indicates that relatively little time is spent on Step 2 of the algorithm in comparison to Step 1 where knapsack problems are solved. The effect of using the early stopping of subproblems or knapsacks in solving the surrogate dual is shown in Table 1. The problem density used was .20 with five replications for each problem size. The improvement in solution times using the early stopping rule is obvious with average solution times reduced by 6.5, 54.1, and 58.2 percent for the 5 x 10, 10 x 20, and 15 x 30 size problems respectively. Note that fewer knapsacks or subproblems were solved and a large percentage of these terminated early in the refined procedure. Thus early termination both reduced the number of knapsacks and reduced the time to "solve" each knapsack.

The constraint dropping technique was employed in all the test problems in Table 1 with the result that for the early termination procedure 0%, 4% and 2.1% regeneration occurred for the 5 x 10, 10 x 20 and 15 x 30 size problems respectively. Without early termination these percentages were 0%, 7.3% and 5.8% respectively. Thus only a small percentage of regenerations occur and the savings in linear pro-

Table 1. Effect of Early Termination of $(P^V)$ in (BTP)

| Problem Size* (Density .20) | Average Solution Time (Sec.) | Time Range (Sec.) | Average No. Knapsacks | Percent Early Knapsacks |
|---|---|---|---|---|
| 5 x 10 | 0.062 | [.022, .148] | 6.4 | 0 |
| | 0.058 | [.018, .128] | 6.4 | 34.8 |
| 10 x 20 | 2.104 | [.986, 3.362] | 42.2 | 0 |
| | 0.966 | [.404, 1.596] | 31.2 | 75.0 |
| 15 x 30 | 27.213 | [10.158, 81.160] | 117.0 | 0 |
| | 11.387 | [4.256, 33.632] | 87.2 | 80.3 |

*5 Replications per cell

gram size is clearly evident from the average number of knapsacks reported in Table 1.

Note that the average solution time is much closer to the lower end of the time range for the five replicates for each problem size. Empirical evidence showed a large variance in solution time with generally one of the five problems taking a comparatively great deal of time. This phenomenon might be expected with such relatively unstructured randomly generated test problems. However, the ranges for the different problem sizes do not overlap and the trend is clear.

To understand these somewhat counter-intuitive results, consider Figure 11 which presents graphically the convergence of a typical 10 x 20 problem under the two procedures. Note that the early termination scheme stabilized the knapsack solution values after fewer subproblems. As shown in Chapter IV, at most m constraints of the form $v(Ax^j - b) \geq z$ will be binding in an optimal solution to (BTP). Naturally, all the binding $x^j$ will ultimately have to have solutions $cx^j$ near $\nu(D_S)$. Figure 11 illustrates that by terminating $(P^u)$ whenever an x feasible to $(P^u)$ is found with cost less than or equal to the $(D_S)$ incumbent solution value, these higher cost x's are generated earlier.

## Linear Relaxation Master Problem Procedure

### Choosing Epsilon

In Chapter IV it was noted that a heuristic stopping

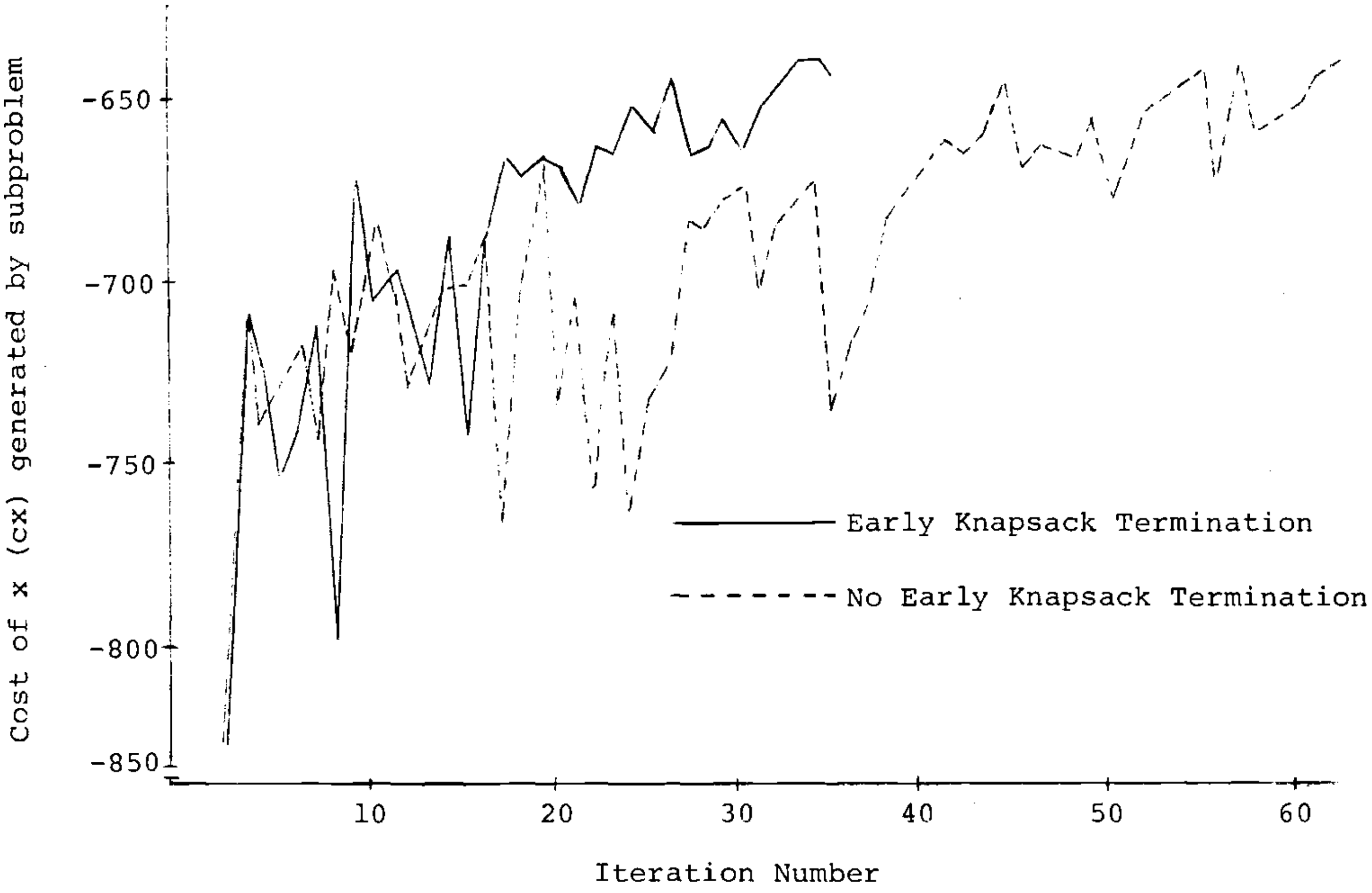


Figure 11. Comparative Convergence of (BTP) with Early Subproblem Termination

rule is required for (LRMP). Also it is necessary to choose a value for the parameter epsilon. Note that if $v^{\ell+1} = v^{\ell} + t(Ax^{\ell} - b)$, where $t = \frac{-v^{\ell}(Ax^{i} - b) + \varepsilon}{(Ax^{i} - b)(Ax^{i} - b)}$, then $x^{\ell}$ is cut off by $\varepsilon$ in $(P^{v^{\ell+1}})$. That is

$$v^{\ell+1}(Ax^{\ell} - b) = \left[v^{\ell} + t(Ax^{\ell} - b)\right](Ax^{\ell} - b) =$$

$$v^{\ell}(Ax^{\ell} - b) - v^{\ell}\frac{(Ax^{\ell} - b)(Ax^{\ell} - b)(Ax^{\ell} - b)}{(Ax^{\ell} - b)(Ax^{\ell} - b)} +$$

$$\varepsilon\frac{(Ax^{\ell} - b)(Ax^{\ell} - b)}{(Ax^{\ell} - b)(Ax^{\ell} - b} = \varepsilon .$$

It was shown that for finite S, convergence is assured for any $\varepsilon > 0$. However, it is evident that too small a choice of $\varepsilon$ could lead to slow convergence, and too large a choice of $\varepsilon$ could lead to a large number of iterations in solving the master problem via linear relaxation. Much 'overstepping' might occur. The choice of epsilon is also clearly tied to the heuristic stopping rules for (LRMP). Recall from Chapter IV that the linear relaxation procedure terminates only for a consistent system of inequalities so that an upper bound must be set on the number of iterations for a given master problem. If the master problem is inconsistent, then the incumbent solution to the surrogate dual is optimal.

Table 2 presents the results of using three different values of epsilon on three different size problems with three

Table 2. (LRMP) for Different Choices of Epsilon

| (1) Problem Size* (Density .20) | (2) Epsilon | (3) Average Solution Time (Sec.) | (4) Average Number of Knapsacks | (5) Second Max Linear Relaxation Iterations Range, Avg. | (6) Max Non-improvement Knapsack Iterations Range, Avg. | Percent of Optimality at Termination |
|---|---|---|---|---|---|---|
| 5 x 10 | 10 | .138 | 6.67 | [2, 6]<br>3.33 | [1, 2]<br>1.3 | 100 |
| | 20 | .121 | 5.67 | [2, 27]<br>10.33 | [2, 2]<br>2.0 | 100 |
| | 30 | .124 | 6.00 | [2, 3]<br>2.66 | [1, 3]<br>2.0 | 100 |
| 10 x 20 | 10 | .869 | 13.33 | [2, 7]<br>4.33 | [1, 3]<br>2.0 | 100 |
| | 20 | .584 | 11.00 | [2, 5]<br>3.7 | [2, 4]<br>2.7 | 100 |
| | 30 | .714 | 12.33 | [2, 26]<br>12.67 | [2, 3]<br>2.33 | 100 |
| 15 x 30 | 10 | 14.042 | 40.00 | [41,49]<br>44.33 | [3, 7]<br>5.67 | 99.93 |
| | 20 | 13.684 | 38.00 | [7, 90]<br>36.67 | [3, 8]<br>6.00 | 99.93 |
| | 30 | 11.756 | 30.67 | [12,16]<br>13.7 | [3, 7]<br>4.4 | 99.91 |

*3 problems per cell

replications per cell. The same three problems were used for all levels of $\varepsilon$ in any (m x n) cell, but the problems are a different set than those used in Table 1. An upper bound of 100 was placed on the maximum linear relaxation iterations before accepting the current incumbent as optimal. Column five reports the second maximum number of linear relaxation iterations employed during the solution procedure. Thus that column is a measure of the maximum number of linear relaxations actually required in feasible master problems (the last one is always infeasible). The percent of optimality for the heuristic procedure (LRMP) is based on the incumbent solution value if stopping had occurred when this second maximum number of linear relaxation iterations had been exhausted. Column six reports the maximum number of iterations in $(P^v)$ which occurred before a new incumbent was found for $(D_S)$.

First, observe that very few linear relaxation iterations were required when master problems are feasible. Values in column five are in most cases of the same magnitude as m.

Next, note that while there seems to be little difference in choosing epsilon for m x n = 5 x 10, it is clear that the best choice of epsilon is near n for the two larger problem sizes. Choosing epsilon equal to n gives fewer iterations in $(P^u)$, has faster solution time, and has the smallest number of linear relaxation iterations.

Finally, note that the maximum number of iterations in $(P^V)$ without a new incumbent is small for all choices of epsilon, so that the steps taken in v appear to be close to ascent directions. Typical empirical evidence indicates that a new incumbent was found during each of the initial iterations in $(P^V)$, with nonimproving iterations occurring close to termination. Too small a choice of epsilon still appeared to find these near ascent directions but generated more x's; i.e., more iterations in $(P^V)$. These x's were 'stepped over' when a larger epsilon was used.

From these observations heuristic rules for the testing of (LRMP) were derived. Epsilon is set at n in all future analyses, and the limit on linear relaxation iterations is fixed at m + 5. Failure to produce new incumbents is not used to terminate $(D_S)$.

### Early Termination of $(P^V)$ in (LRMP)

Table 3 presents the same type of results for (LRMP) as presented in Table 1 for (BTP). Test problems used are the same as those of Table 1 but different than those used in Table 2 to derive heuristic rules. Again it can be seen that early termination of the knapsacks or subproblems leads to a more efficient solution procedure. Here, however, the improvement is not nearly as drastic.

## Comparison of (BTP) and (LRMP)

The same problems were used in Tables 1 and 3 so that a direct comparison of the best procedures for (BTP) and

Table 3. Effect of Early Termination of ($P^V$) in (LRMP)

| Problem Size* (Density .20) | Average Solution Time (Sec.) | Time Range (Sec.) | Average No. Knapsacks | Percent Early Knapsacks | Percent Optimal at Termination |
|---|---|---|---|---|---|
| 5 x 10 | .033 | [.018, .058] | 6.0 | 0 | 100 |
| | .043 | [.024, .074] | 4.6 | 17.4 | 100 |
| 10 x 20 | .697 | [.260, 1.592] | 12.6 | 0 | 100 |
| | .658 | [.232, 1.468] | 12.6 | 34.9 | 99.6 |
| 15 x 30 | 6.54 | [3.276, 14.118] | 28.8 | 0 | 100 |
| | 4.434 | [2.194, 10.982] | 26.0 | 43.1 | 99.8 |

*5 Replications per cell

(LRMP) is possible. Table 4 summarizes these results and clearly shows the superiority of the (LRMP) procedure. Both total solution time and the number of knapsack subproblems are substantially reduced when (LRMP) is employed.

The only possible drawback of (LRMP) is that it provides only a near optimal solution to $(D_S)$ due to the need for a heuristic stopping rule. However, Table 3 shows that (LRMP) is certainly a very near optimal algorithm. Of the fifteen total replications, only three problems were not 100% optimal, leading to an average percent of optimality of 100.0, 99.6, and 99.8 for the 5 x 10, 10 x 20, and 15 x 30 size problems respectively.

Another point of interest is the comparison of solution times for (LRMP) and the linear programming relaxation of (P). Using a revised simplex procedure in the CDC math science library (ZX-LP3), produced average solution times of .160, .827, and 2.182 seconds for the 5 x 10, 10 x 20, and 15 x 30 size problems respectively. The average solution times for $(D_S)$ using (LRMP) compare favorably with these times. If the surrogate dual can provide better bounds on $\nu(P)$ compared to $\nu(\bar{P})$ or $\nu(D_L)$, then even much slower times than reported here might be satisfactory for beneficial application of the surrogate dual in a branch-and-bound procedure.

## The Primal Branch-and-Bound Procedure

Drawing on the results of the previous analyses, a

Table 4. Comparison of the (BTP) and (LRMP) Procedures

| Problem Size* (Density .20) | Procedure | Average Time (Sec.) | Time Range (Sec.) | Average Knapsacks | Percent Early Knapsacks | $\frac{\text{BTP Time}}{\text{LRMP Time}}$ |
|---|---|---|---|---|---|---|
| 5 x 10 | BTP | .058 | [.018, .128] | 6.4 | 34.8 | 1.35 |
| | LRMP | .043 | [.024, .074] | 4.6 | 17.4 | |
| 10 x 20 | BTP | .966 | [.404, 1.596] | 31.2 | 75.0 | 1.49 |
| | LRMP | .658 | [.232, 1.468] | 12.6 | 34.9 | |
| 15 x 30 | BTP | 11.387 | [4.256, 33.632] | 87.2 | 80.3 | 2.57 |
| | LRMP | 4.434 | [2.194, 10.982] | 26.0 | 43.1 | |

*5 Replications per cell

version of (LRMP) was further tested in the full branch-and-bound context. The general branch-and-bound approach is outlined in Chapter I, and specific details with respect to surrogate duality are presented in Chapters IV and V. Before discussing computational results a summary of the algorithm for the 0-1 integer programming test problems is given below in a step-by-step fashion.

Statement of the Algorithm

In terms of the notation of Chapters IV and V, and the step numbers of Figure 1, the primal branch-and-bound algorithm used in experimentation may be stated as:

Step 0a: Generate a random 0-1 integer programming problem according to input specifications of problem size and density as outlined in this chapter and the Appendix. Then go to Step 0b.

Step 0b: Initialize the pointers of the next open slot in the current and save tables; i.e., set NXCR = NXSV = 1. Set epsilon = n and the surrogate multiplier vector v = 1. Place $P(\phi)$ in the candidate list with bound $-\infty$, set $\nu^*(P) = +\infty$. Go to Step 1.

Step 1: Choose the most recently created unexplored candidate problem, P(T), to explore next. This corresponds to the LIFO procedure discussed in Chapter V. Set $\nu^*(D_S(T))$ = saved bound for completions of P(T). If NXSC = 1, go to Step 2. Otherwise go to Step 4.

Step 2a: Solve $(P^V(T)$ as outlined in Figure 9, always branching on $x_{(\ell)} = 0$ first, employing a LIFO procedure. Go to Step 2b.

Step 2b: If $\nu(P^V(T)) \geq \nu^*(P)$, go to Step 8 and fathom. If the solution to $(P^V(T))$ is feasible for (P), go to Step 3. Otherwise, go to Step 2c.

Step 2c: Add the solution to $(P^V(T))$ to the current table, incrementing NXCR by one. If $\nu(P^V(T)) > \nu^*(D_S(T))$, replace $\nu^*(D_S(T))$ by $\nu(P^V(T))$, and update the incumbent branching variable and bounds $x_{(B1)}$, $\nu^{(1)}$, and $\nu^{(2)}$. Then go to Step 4.

Step 3: Save solution as new incumbent and eliminate any members of the candidate list with bound $\geq \nu^*(P)$. Go to Step 8.

Step 4: Use linear relaxation on the x's in the current table as outlined in Chapter IV. If the number of iterations is greater than ITMAX = m + 5, go to Step 6. Otherwise update v and go to Step 2.

Step 6: Branch on $x_{(B1)}$ as defined in Step 2c. If $\nu^{(1)} \leq \nu^{(2)}$, branch on $x_{(B1)} = 0$. Otherwise check to see if $x_{(B1)} = 1$ precludes a feasible solution to (P). If so, branch on $x_{(B1)} = 0$, otherwise choose $x_{(B1)} = 1$. Go to Step 7.

Step 7: Adjust the current table and save table as outlined in Chapter V, creating two new candidate problems with bounds $\nu^{(1)}$ and $\nu^{(2)}$.

Step 8: Fathom P(T), adjusting the save and current tables as outlined in Chapter V. If no candidate problems remain, stop; $\nu^*(P)$ is optimal for (P). Otherwise, the fathoming procedure outlined in Chapter V will place any appropriate x's for the candidate problem to be explored next in the current table. Go to Step 1.

Computational Results

Table 5 presents the results of employing the above algorithm on three problem sizes with a low and a high density and five replications per cell. The principal cause for interest in surrogate duals is improvement in bounds. The percent of the LP to IP gap closed by the surrogate dual, i.e.,

$$(\nu(D_S) - \nu(\bar{P}))/(\nu(P) - \nu(\bar{P}))$$

appears substantial. These improved bounds resulted in total solution times at least comparable to other published results. The large range for a given cell is perhaps to be expected with such unstructured randomly generated problems.

Some measure of the efficiency of the interaction between the primal and the subproblem branch-and-bound procedures is provided by the remaining columns of Table 5. As expected the principal part of all time spent on candidate problems is consumed in knapsack subproblems. Values in column 8 range from 71% to 82%. However, the number of knap-

Table 5. Primal Branch-and-Bound Empirical Results*

| (1) Problem Size Density | (2) % of LP to IP Gap Closed by $D_S$ Average [Range] | (3) Total Time of Branch-and-Bound Procedure (Sec.) | (4) Time for First $D_S$, i.e., $D_S(S)$ (Sec.) | (5) Total Nodes Explored | (6) Knapsacks Solved Per Node | (7) $D_S$ Time/ Node (Sec.) (% of 1st Node) | (8) Percent in Knapsack Time/Node |
|---|---|---|---|---|---|---|---|
| 5 x 10<br>.10 | 74.5<br>[8.3, 100] | .034 | .024 | 2.0 | 2.67 | .013<br>(54.2) | 72.3 |
| 10 x 20<br>.10 | 15.2<br>[3.7, 45.5] | 3.20 | .525 | 18.0 | 5.31 | .122<br>(3.8) | 71.3 |
| 15 x 30<br>.10 | 5.9<br>[1.7, 14.0] | 35.93 | 3.10 | 109.4 | 5.69 | .337<br>(10.9) | 73.0 |
| 5 x 10<br>.25 | 51.7<br>[14.4, 100] | .140 | .057 | 4.8 | 3.37 | .028<br>(20.0) | 82.1 |
| 10 x 20<br>.25 | 17.4<br>[10.5, 28.1] | 6.18 | 1.26 | 44.0 | 3.88 | .139<br>(11.0) | 79.1 |
| 15 x 30<br>.25 | 6.6<br>[3.3, 8.7] | 107.3 | 4.92 | 308.2 | 4.18 | .325<br>(6.6) | 79.7 |

*Five Replications per cell

sack subproblems solved at any particular node is quite small (column 6). The small numbers are a consequence of the save table - current table scheme developed in Chapter V. Another indication of the efficiency of the save table approach is the relation between the mean time to solve the first surrogate dual (column 4) and the mean time to solve all surrogate duals (column 7). For larger problem sizes the average surrogate dual---which begins with many $x^k$ saved from previous knapsacks---solves in 5-10% of the time for the first dual.

# CHAPTER VII

# CONCLUSIONS AND RECOMMENDATIONS

The primary objective of the research reported in this dissertation was to investigate theoretical and algorithmic issues in the use of surrogate duals or their extensions to solve integer programming problems. A number of theoretical results were developed in Chapters II and III, algorithmic developments are presented in Chapters IV and V, and computational experience is reported in Chapter VI. The principal results can be summarized as follows:

1. Generalization of Geoffrion's Integrality Property to the Surrogate Dual and its Extensions. The bounding weakness of many natural lagrangian dual formulations of integer programs can be demonstrated through Geoffrion's (20) integrality property which gives a sufficient condition under which the lagrangian dual cannot produce a better bound than the linear programming relaxation of the same problem. Generalization of the integrality property in Chapter III leads to the result that such bounding weakness would generally not be expected for the surrogate, composite, or multiple surrogate duals. Empirical confirmation of these results is presented in Chapter VI using randomly generated 0-1 integer programming test problems.

2. Investigation of the Occurrence of Gaps Between the Various Duals. Past interest has focused on conditions under which there are no primal-to-dual gaps, but such gaps are expected in most integer programming problems. The more important question considered here, is the existence of gaps between the various duals and hence opportunities for improved bounding power. A necessary and sufficient gap detection test was developed for the lagrangian-to-surrogate relationship which is sufficiently general to show that a lagrangian-to-surrogate duality gap is very likely. Sufficient conditions were also developed for the surrogate-to-composite and composite-to-multiple surrogate gap cases. Empirical evidence presented in Chapter VI demonstrated significant gaps and thus bound improvements between the lagrangian and surrogate duals for randomly generated 0-1 test problems.

3. Characterization of the Composite and Multiple Surrogate Dual Functions. The composite and multiple surrogate dual functions were shown to lack quasi-concavity which is perhaps a minimum requirement for efficient multiplier search procedures. This result leads both to weaker gap characterizations for the composite and multiple surrogate cases and the general conclusion that such extensions are not computationally promising.

4. Development of Search Procedures for Optimal Surrogate Dual Multipliers. In contrast to the composite and

multiple surrogate cases, several efficient procedures were developed for calculating optimal surrogate dual multipliers. A linear programming-based search procedure, suggested previously, was refined and tested, and two new procedures were developed. For all these procedures it was shown that the surrogate subproblems, $(P^v)$, need not always be solved optimally in order for the search algorithms to converge. In fact empirical results in Chapter VI show convergence is faster if the knapsack-like subproblems are not fully optimized. The two search procedures deemed most efficient were also compared in the solution of random 0-1 test problems. The most promising procedure appears to be a linear relaxation-based method developed in this research.

5. Application of Surrogate Duality in a Primal Branch-and-Bound Procedure. It was shown that the inner play between the surrogate subproblem and the primal branch-and-bound trees can be exploited to produce a number of computational efficiencies. Most important was a restarting procedure which precludes the necessity of having to solve numerous surrogate subproblems at each node of a primal branch-and-bound tree. Empirical evidence presented in Chapter VI suggests this procedure greatly reduces total computation time.

Taken together, these results seem to imply considerable promise for surrogate dual approaches in integer programming. Optimal or near optimal surrogate multipliers can be obtained efficiently, often by only partially solving a small number of knapsack-like subproblems. Moreover, the bounds produced by surrogate duals are substantially better than those obtained from linear programming relaxations.

Successful integer programming applications of the lagrangian dual have resulted from the exploitation of special structure in specific classes of integer programming problems. The encouraging theoretical, algorithmic and empirical results presented in this dissertation clearly point to the need for future research in applying the surrogate dual to special classes of integer programming problems. It is anticipated that by employing surrogate duality and taking advantage of special problem structures, new and more efficient solution procedures will be obtained for a number of integer programming problems. A good deal of computational fine tuning of the surrogate methods, perhaps best undertaken with respect to special classes of problems, could also be useful in some or all of the following areas:

·Extension of the (LRMP) procedure to the S not finite case and further investigation into constraint dropping in both the (LRMP) and (BTP) procedures.

·Initial choice of a surrogate multiplier, including restart at new nodes in the primal branch-and-bound tree.

·Selection of a level for epsilon in the (LRMP) procedure, possibly one which changes either while solving a particular surrogate dual or when reaching levels in the primal branch-and-bound tree which have the effect of reducing problem dimensionality.

·Further development of good branching procedures which draw on the inner play between the surrogate subproblem and primal branch-and-bound trees.

·Application of the current table, save table concepts to more general primal branching procedures.

# APPENDIX

## RANDOM PROBLEM GENERATOR

The following is a FORTRAN code of the random problem generator used for the computational experience reported in Chapter VI. The random number generator is a multiplicative congruential random number generator for a forty-eight bit binary machine. The higher order digits are lost in any number generated which is greater than $2^{48}$. Dividing by $2^{48}$ produces a number between zero and one.

```
      SUBROUTINE GENRIP(N,M,DENS,ISEED,A,B,C)
      DIMENSION A(30,60),B(30),C(60)
      T=(5.5*N*DENS)/4
C        COEFF OF A UNIFORM(1,10) WITH DENSITY=DENS
C        B UNIFORM((1/4,3/4)*EXP VAL SUM OF ROW COEFF)
      DO 10 I=1,M
      VAL=(2.0*RANDG(ISEED))+1.0
      VAL=VAL*T
      IVAL=VAL
C        IVAL IS GREATEST INTEGER OF VAL
      B(I)=IVAL
      DO 20 J=1,N
      IF (RANDG(ISEED) .LT. DENS) GO TO 30
      A(I,J)=0
      GO TO 20
30    VAL=(10.*RANDG(ISEED))+1.0
      IVAL=VAL
      A(I,J)=IVAL
20    CONTINUE
10    CONTINUE
C        C UNIFORM(-10,-100)
      DO 40 J=1,N
      VAL=(90.*RANDG(ISEED))+11.0
      IVAL=VAL
40    CONTINUE
      RETURN
      END
      FUNCTION RANDG(ISEED)
C        ISEED IS INTEGER SEED
C        RANDG RETURNS REAL ON (0,1)
      ISEED=ISEED*16777219
      RANDG=ISEED/(281474976710656.0)
      RETURN
      END
```

Randomly generated nine digit integer seeds were used for the problems in Tables 1, 3, and 4 which made comparisons on the same set of test problems.

| 5 x 10 | 10 x 20 | 15 x 30 |
|---|---|---|
| 452675969 | 888759193 | 055101937 |
| 983576025 | 814546993 | 089587913 |
| 685763697 | 690518817 | 266443233 |
| 375619913 | 640835617 | 845273913 |
| 170027105 | 067443577 | 942547153 |

The following seeds were used for the test problems in Table 2.

| 5 x 10 | 10 x 20 | 15 x 30 |
|---|---|---|
| 948152249 | 373563153 | 276996233 |
| 146399057 | 101051113 | 973857697 |
| 357170985 | 590103897 | 025901817 |

The following seeds were used for the test problems in Table 5.

### Density = .10

| 5 x 10 | 10 x 20 | 15 x 30 |
|---|---|---|
| 226462577 | 134914337 | 957799377 |
| 508304969 | 974578809 | 080352169 |
| 251554913 | 607770129 | 584728001 |
| 241886905 | 745935337 | 711650841 |
| 466631761 | 809141761 | 033583793 |

### Density = .25

| 5 x 10 | 10 x 20 | 15 x 30 |
|---|---|---|
| 826617897 | 715663961 | 421477769 |
| 717948993 | 026715377 | 185410209 |
| 326144153 | 894129609 | 796644345 |
| 006571825 | 684845281 | 922901393 |
| 886123017 | 761377849 | 071514473 |

# BIBLIOGRAPHY

1. Agmon, S., "The Relaxation Method for Linear Inequalities," Canadian Journal of Mathematics, 6, 382-392, 1954.

2. Balas, E., "Discrete Programming by the Filter Method," Operations Research, 19-5, 915-957, 1967.

3. Balas, E., "Minimax and Duality for Linear and Nonlinear Mixed Integer Programming," Working Paper No. 9-69-7, Carnegie-Mellon University, 1969.

4. Banerjee, K., "Generalized Lagrange Multipliers in Dynamic Programming," Research Report ORC 71-12, Operations Research Center, University of California, Berkeley, 1971.

5. Bazaraa, M. S. and J. Goode, "A Survey of Various Tactics for Generating Lagrangian Multipliers in the Context of Lagrangian Duality," unpublished research report in the School of Industrial and Systems Engineering, Georgia Institute of Technology, 1974.

6. Bazaraa, M. S., personal communication, 1976.

7. Benders, J. F., "Partitioning Procedures for Solving Mixed-Variables Programming Problems," Numerische Mathematik, 4, 238-252, 1962.

8. Brooks, R. and A. M. Geoffrion, "Finding Everett's Lagrange Multipliers by Linear Programming," Operations Research, 14, 1149-1153, 1966.

9. Cabot, V. A., "An Enumeration Algorithm for Knapsack Problems," Operations Research, 18, 306-311, 1970.

10. Dantzig, G. B. and P. Wolfe, "The Decomposition Algorithm for Linear Programming," Econometrica, 29, 767-778, 1961.

11. Everett, H., "Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources," Operations Research, 11, 399-417.

12. Fayard, D. and G. Plateau, "Resolution of the 0-1 Knapsack Problem: Comparison of Methods," Mathematical Programming, 8, 272-307, 1975.

13. Fisher, M. L., "Optimal Solution of Scheduling Problems Using Generalized Lagrange Multipliers: Part I," Operations Research, 21, 1114-1127, 1973.

14. Fisher, M. L., W. D. Northrup and J. F. Shapiro, "Using Duality to Solve Discrete Optimization Problems: Theory and Computational Experience," Mathematical Programming Study, 3, 59-94, 1975.

15. Ford, L. R. and D. R. Fulkerson, Flows in Networks, Princeton University Press, 1962.

16. Garfinkel, R. S. and G. L. Nemhauser, Integer Programming, John Wiley and Sons, 1972.

17. Geoffrion, A. M., "An Improved Implicit Enumeration Approach for Integer Programming," Operations Research, 17-3, 437-454, 1969.

18. Geoffrion, A. M., "The Capacitated Plant Location Problem with Additional Constraints," Paper presented to the Joint National Meeting of AIIE, ORSA and TIMS, Atlantic City, November, 1972.

19. Geoffrion, A. M., "A Guided Tour of Recent Advances in Integer Linear Programming," SIGMAP Newsletter, 17, 22-32, 1974.

20. Geoffrion, A. M., "Lagrangian Relaxation and Its Uses in Integer Programming," Mathematical Programming Study, 2, 82-114, 1974.

21. Geoffrion, A. M. and R. E. Marsten, "Integer Programming Algorithms: A Survey," Management Science, 18, 465-491, 1972.

22. Gilmore, P. C. and R. E. Gomory, "Multistage Cutting Stock Problems of Two and More Dimensions," Operations Research, 13, 94-120, 1965.

23. Gilmore, P. C. and R. E. Gomory, "The Theory and Computation of Knapsack Functions," Operations Research, 14, 1045-1074, 1966.

24. Glover, F., "A Multiphase-Dual Algorithm for the Zero-One Integer Programming Problem," Operations Research, 13-6, 879-919, 1965.

25. Glover, F., "Surrogate Constraints," Operations Research, 16, 741-749, 1968.

26. Glover, F., "Surrogate Constraint Duality in Mathematical Programming," Operations Research, 23-3, 434-451, 1975.

27. Greenberg, H. and R. L. Hegerick, "A Branch Search Algorithm for the Knapsack Problem" Management Science, 16, 327-332, 1970.

28. Greenberg, H. J. and W. P. Pierskalla, "Surrogate Mathematical Programming," Operations Research, 18, 924-939, 1970.

29. Greenberg, H. and W. P. Pierskalla, "Quasi-Conjugate Functions and Surrogate Duality," Cahiers Cent. d´Etudes Recherche Operationelle, 15, 437-448, 1973.

30. Hansen, K. H. and J. Krarup, "Improvements of the Held-Karp Algorithm for the Symmetric Travelling-Salesman Problem," Mathematical Programming, 7, 87-96, 1974.

31. Held, M. and R. M. Karp, "The Travelling Salesman Problem and Minimum Spanning Trees," Operations Research, 18, 1138-1162, 1970.

32. Held, M. and R. M. Karp, "The Travelling Salesman Problem and Minimum Spanning Trees: Part II," Mathematical Programming, 1, 6-25, 1971.

33. Kolesar, P. J., "A Branch and Bound Algorithm for the Knapsack Problem," Management Science, 13, 723-735, 1967.

34. Lasdon, L. S., Optimization Theory for Large Systems, The MacMillan Company, 1970.

35. Luenberger, D. G., "Quasi-Convex Programming," SIAM Journal of Applied Mathematics, 16, 1090-1095, 1968.

36. Mangasarian, O. L., Nonlinear Programming, McGraw-Hill, 1969.

37. Motzkin, T. S. and I. J. Schoenberg, "The Relaxation Method for Linear Inequalities," Canadian Journal of Mathematics, 6, 393-404, 1954.

38. Nemhauser, G. L. and Z. Ullman, "A Note on the Generalized Lagrange Multiplier Solution to an Integer Programming Problem," Operations Research, 16, 450-452, 1968.

39. Rardin, R. L., "Group Theoretic and Related Approaches to Fixed Charge Problems," Ph.D. dissertation, School of Industrial and Systems Engineering, Georgia Institute of Technology, December, 1973.

# VITA

Mark Henry Karwan was born November 16, 1951 in Cleveland, Ohio and grew up in the Cleveland area. He graduated from University School in 1970 and immediately entered the Johns Hopkins University, having been awarded a University scholarship. As an undergraduate, his major area of study was Mathematical Sciences. While at Johns Hopkins University, he was nominated to the engineering honor society, Tau Beta Pi, and was awarded the Bachelor of Engineering Science degree with departmental and general honors.

While completing a Master's Project on the routing of delivery vehicles, Mr. Karwan was a consultant to Management Advisory Services, Inc. of Columbia, Maryland, contributing to the development of a comprehensive out-patient facility program for the state of Maryland. Upon receiving the Master of Science in Engineering degree, he accepted a President's Fellowship to attend the Georgia Institute of Technology. There he pursued a doctoral program in Operations Research, and for a year was a graduate assistant responsible for the teaching of undergraduate operations research courses. He was awarded the Ph.D. degree in 1976, and immediately accepted a permanent faculty position as an Assistant Professor in Industrial Engineering at the State University of New York at Buffalo.

Mr. Karwan was married in August, 1973, to Miss Sabina Matarazzo of East Hanover, New Jersey.