MANAGEMENT SCIENCE
Vol. 23, No. 5, January, 1977
Printed in U.S.A.

Exceptional Paper

PARAMETRIC AND POSTOPTIMALITY ANALYSIS IN INTEGER LINEAR PROGRAMMING*†

A. M. GEOFFRION‡ AND R. NAUSS§

The purpose of this paper is to take stock of what is known and to suggest some conceptual foundations for future progress in the areas of postoptimality analysis and parametric optimization techniques for integer programming.

1. Introduction

Postoptimality analysis and parametric optimization techniques are fully developed aspects of linear programming. Their value in practical applications is by now well established. In the context of integer linear programming, however, these aspects have barely begun to be developed. The purpose of this paper is to take stock of what is known about this topic and to lay the foundation for future progress.

Our conceptual starting point is the notion that, in practical applications, typically one is faced not with a single numerical integer linear program to solve but rather with an entire *family* of numerical problems of interest. The members of the family may all have the same structure but differ as to the values of one or more coefficients, or they may even have different (but related) structures. The scope of this paper is limited to the first mentioned case.

By way of establishing some notation to be used subsequently, we write a general finite family of (possibly mixed) integer linear programs as $\{(P)^1, \ldots, (P)^k, \ldots, (P)^K\}$, where

$$\underset{x \geqq 0}{\text{Minimize}}\ c^k x \text{ subject to } A^k x \geqq b^k, \quad x_j \text{ integer}, j \in I. \qquad (P)^k$$

Here x is an n-vector, b^k, c^k, and A^k are given vectors and matrices of conformable dimensions, and I is an index set specifying which variables must be integer-valued. Normally the problem data b^k, c^k, and A^k will vary with the index k in a systematic rather than an arbitrary way. In some applications it is convenient to think of the data as varying continuously with a continuous-valued problem index. The most common situation is linear variation of c or b with respect to a single scalar parameter, which we shall write as:

$$\underset{x \geqq 0}{\text{Minimize}}\ (c + \theta f)x \text{ subject to } Ax \geqq b, \quad x_j \text{ integer}, j \in I, \text{ and} \qquad (P^\theta)$$

$$\underset{x \geqq 0}{\text{Minimize}}\ cx \text{ subject to } Ax \geqq b + \theta r, \quad x_j \text{ integer}, j \in I, \qquad (P_\theta)$$

* This paper has been refereed under guidelines for exceptional contributions.

† Accepted by Willard I. Zangwill; received February 25, 1976.

‡ University of California, Los Angeles.

§ University of Missouri–St. Louis.

where without loss of generality we can assume that the scalar parameter θ satisfies $0 \leqslant \theta \leqslant 1$.

How do such families of integer programs arise in practice? The answer is that they arise for all of the traditional reasons that spawn parametric linear programs and more general sequences of LPs, plus several new reasons without precedent in the domain of LP. The traditional reasons include the need for sensitivity analysis as a means of exploring the implications of questionable assumptions made in the model, tradeoff curve development as a way of dealing with multiple criteria, case studies to account for aspects that cannot be formulated as a single linear optimization problem, and so on.

In addition to the traditional reasons, it is important to recognize that families of integer programs arise for technical reasons quite peculiar to integer programming. For instance, IP models have no shadow prices or dual variables with an interpretation comparable to that in linear programming. In order to determine—even locally—the influence of varying a resource level on the optimal value, in general one must re-solve the problem with alternative resource levels. This requires parametric right-hand side analysis as in (P_θ) or some discrete version thereof. The absence of meaningful shadow prices in IP is a manifestation of a more general technical difficulty: *Neither the optimal value nor the optimal solution of an integer program need be continuous as a function of the coefficients defining the constraints.* Ordinarily this difficulty does not occur in linear programming, where small changes in the data lead to small changes in the results.[1] This property is one of the reasons why LP models usually behave "reasonably" from a managerial or engineering viewpoint when re-solved with alternative data values. Integer programming models, on the other hand, can behave in an erratic and unpredictable manner due to the presence of multiple discontinuities caused by (necessarily discrete) changes of value for the integer variables. It is therefore wise to conduct what might be called a *continuity analysis* study for most IP models in order to ascertain whether or not any discontinuities in the region of interest are large enough to diminish the usefulness of the numerical results. Continuity analysis for integer programming is an interesting subject for study in its own right and has recently been developed at length by M. Radke [15]. Suffice it to note here that the need for continuity analysis provides a strong incentive for the development of methods that can solve a family of related IPs with an effort less than directly proportional to that required to solve a single member of the family.

This paper is organized around four fundamental questions of relevance to postoptimality analysis and parametric approaches in the context of IP:

1. Having numerically solved a given integer linear program, to what extent can its data be changed without invalidating the optimality of this solution?

2. Having numerically solved one or more integer programs in a family of type (P^θ), or of type (P_θ), what can be said about the optimal value and/or solution of other members of the same family?

3. What can the user of a conventional branch-and-bound code do to achieve better-than-brute-force results when using it on a family of related problems?

4. How can conventional branch-and-bound algorithmic design be modified for greater effectiveness when applied to a family of related problems?

We study each question in turn. An effort is made for the sake of completeness to incorporate most of the pertinent results from the small extant literature on related topics. However, ϵ-optimality extensions of exact results are not included here. Such

[1] This statement must be qualified to account for certain pathological cases, and continuity of the optimal solution must be defined in terms of point-to-set mappings [15].

results are very important in practice but are omitted for the sake of brevity. It is usually a straightforward exercise to obtain these results from the exact results presented below.

2. Data Changes that Preserve Optimality

Suppose that the problem

$$\underset{x \geqq 0}{\text{Minimize}}\ cx \text{ subject to } Ax \geqq b, \quad x_j \text{ integer}, j \in I \subseteq \{1, \ldots, n\}, \qquad (P)$$

has been solved numerically for an optimal solution x^*. What changes in the data coefficients will not destroy the optimality of x^* for the revised problem?

A. *Results Based on Elementary Observations*

Simple but useful results can be obtained with the help of the concept of restriction.

DEFINITION 2.1. A problem (Q) is said to be a *restriction* of problem (P) if the feasible region of (Q) is entirely contained within that of (P), and if the objective function of (Q) is at least as great as that of (P) everywhere on the feasible region of (Q).

PROPOSITION 2.2. *If an optimal solution x^* of (P) remains feasible in a restriction (Q) of (P), and if it has the same objective function value in (Q) as in (P), then it must be an optimal solution of (Q).*

The proof is elementary. The kinds of restrictions for which this result is particularly useful are those in which some components of A are decreased, some components of b are increased, or some components of c change in a benign way.

COROLLARY 2.2.1. *Let A' satisfy $A' \leqq A$ and $A'x^* \geqq b$. Then x^* remains optimal for (P) with A replaced by A'. (Note that for any j such that $x_j^* = 0$, the corresponding column of A can be decreased by an arbitrary amount.)*

COROLLARY 2.2.2. *Let b' satisfy $Ax^* \geqq b' \geqq b$. Then x^* remains optimal for (P) with b replaced by b'.*

COROLLARY 2.2.3. *Let c' satisfy $c_j' \geqslant c_j$ for all j such that $x_j^* = 0$, and $c_j' = c_j$ otherwise. Then x^* remains optimal for (P) with c replaced by c'.*

COROLLARY 2.2.4. *Suppose that (P) includes (among the $Ax \geqq b$ constraints) upper bounds u_j on the variables x_j for $j \in J \subseteq \{1, \ldots, n\}$. Let c' satisfy $c_j' \leqslant c_j$ for all $j \in J$ such that $x_j^* = u_j$, and $c_j' = c_j$ otherwise. Then x^* remains optimal for (P) with c replaced by c'.*

PROOF. Apply Proposition 2.2 to the following restriction of (P):

$$\underset{x \geqq 0}{\text{Minimize}}\ cx + \sum_{j \in J} (c_j - c_j')(u_j - x_j)$$

$$\text{subject to } Ax \geqq b \text{ and } x_j \text{ integer}, j \in I.$$

The kinds of data changes specified in the four corollaries may, of course, occur simultaneously without disturbing the optimality of x^*.

Proposition 2.2 is limited to data changes that cause the feasible region of (P) to shrink, or the objective function to worsen at values of x *other* than x^*, or both. Data changes that admit new feasible solutions obviously threaten the optimality of x^*; readily available conditions to preclude this possibility seem elusive except for the all-integer case, where it is easy to specify small data changes that cannot result in any

new feasible solutions at all (cf. [11]). Similarly, optimality is threatened by any change in c that can improve the evaluation of other feasible solutions relative to x^*. One situation in which the continued optimality of x^* can be assured in spite of such a change for c_j arises when a lower bound is available on the optimal value of (P) subject to the additional condition that x_j does *not* take on the value x_j^*. A typical result of this type is as follows.

PROPOSITION 2.3. *Suppose that a particular variable x_j is declared to be* 0-1 *and $x_j^* = 1$. Suppose further that the optimal value of $(P \mid x_j = 0)$ is known to be greater than or equal to the optimal value of (P) plus a certain nonnegative quantity Δ_j. Then x^* remains optimal in (P) with c_j replaced by any coefficient c_j' satisfying $c_j \leqslant c_j' \leqslant c_j + \Delta_j$.*

PROOF. Every feasible solution of (P) with $x_j = 1$ will worsen in value by $c_j' - c_j$; x^* will remain best among these, and can become second best to a solution with $x_j = 0$ only if $c_j' - c_j$ were to exceed the amount by which the optimal value of $(P \mid x_j = 0)$ exceeds the optimal value of (P). But $c_j' - c_j \leqslant \Delta_j$ and Δ_j is no greater than the latter amount by definition.

A value for Δ_j is available for some variables as a by-product of the solution of (P) by some integer programming algorithms—for instance, via the "penalties" often computed by branch-and-bound algorithms. (In fact, simple penalties are readily available at no extra computational cost for any integer programming algorithm that begins by solving (P) as an LP without the integrality conditions.)

Notice that Proposition 2.3 may allow an *increase* in c_j for j such that $x_j^* = 1$, whereas Corollary 2.2.4 allowed only a decrease. A companion result to Proposition 2.3, which we leave to the reader, addresses a *decrease* in c_j for j such that $x_j^* = 0$.

The simplicity of proof of the results obtained so far should not be interpreted as implying any lack of their practical utility. They are quite useful as far as they go. To go further, however, seems to require more elaborate computations that may not always yield positive results. This is the subject of the next subsection.

B. *Results Based on Sufficient Conditions for Optimality*

In the case of ordinary linear and convex programming, one can derive elegant optimality-preserving ranges on data coefficients by exploiting readily available optimality conditions. The necessity of these conditions implies that they can be specified numerically at an optimal solution x^*—typically in terms of an explicit optimal multiplier vector λ^*—and the sufficiency of these conditions enables the desired optimality-preserving ranges to be expressed numerically with the help of λ^*.

The situation in integer linear programming is more difficult because convenient necessary and sufficient optimality conditions are not generally available. Only sufficiency conditions are generally available, and their lack of necessity makes it difficult, if not impossible, to predict whether they will yield useful optimality-preserving ranges on the data coefficients in any particular application.

The most convenient sufficient conditions for optimality in integer programming are those based on the concept of relaxation.

DEFINITION 2.4. A problem (R) is said to be a *relaxation* of a problem (P) if the feasible region of (R) contains that of (P) and if the objective function of (R) is less than or equal to that of (P) on the feasible region of (P).

It is evident upon comparing Definitions 2.1 and 2.4 that relaxation and restriction are inverse relations: (R) is a relaxation of (P) if and only if (P) is a restriction of (R). Clearly the optimal value of any relaxation (restriction) of (P) is a lower (upper) bound on the optimal value of (P).

The following evident companion result to Proposition 2.2 states the natural sufficiency conditions associated with a relaxation.

PROPOSITION 2.5. *If x^0 is a feasible solution of (P) and cx^0 is identical with the optimal value of some relaxation (R) of (P), then x^0 must be optimal in (P).*

A general method for attempting to construct optimality-preserving ranges is this:

1. Based on knowledge of x^*, construct what is likely to be a "tight" relaxation (R^0) of (P); stop if the optimal value of (R^0) is strictly less than that of (P).
2. Determine a region over which the data coefficients can vary and yet leave x^* feasible with an objective function value for (the modified) (P) which equals that of the optimal value of (the modified) (R^0); x^* remains optimal over this data coefficient region.

It is understood that (R^0) is modified in the obvious way as the data coefficients of (P) are changed, so that the modified (R^0) corresponds to a bona fide relaxation of the modified (P).

The method as stated requires the optimal value of (R^0) to equal that of (P). This is a very stringent condition. In practice, one would be wise to seek an epsilon-optimality preserving region for some suitable positive epsilon. The above method has an obvious extension to accommodate this aim. In any case, the computational efficiency of the method depends upon how easily the modified relaxations can be solved.

This is the same method as is used in linear programming—although usually expressed in different but equivalent terms—to determine the well-known optimality ranges for objective function and right-hand side coefficients and for constraint coefficients corresponding to nonbasic variables. The choice of (R^0) in that case is the Lagrangean relaxation determined by the optimal multiplier vector of the original LP, an immediate by-product of any Simplex-type method of solution.

Lagrangean relaxation is also a natural choice for (R^0) in the context of integer linear programming [7]. To illustrate, suppose that the constraints $Ax \geqslant b$ of (P) are composed of two types,

$$A_1x \geqslant b_1 \quad \text{(general)}, \qquad A_2x \geqslant b_2 \quad \text{(special)},$$

where the second type of constraints are "special" in the sense that the Lagrangean relaxation

$$\begin{aligned} &\underset{x \geqq 0}{\text{Minimize}} \ cx + \lambda(b_1 - A_1x) \\ &\text{subject to } A_2x \geqq b_2, \quad x_j \text{ integer}, j \in I, \end{aligned} \tag{LR_λ}$$

can be solved relatively simply without the need for an iterative algorithm for any choice of $\lambda \geqq 0$ (see [7] for several illustrations). Suppose further that some particular $\lambda^0 \geqq 0$ is available as an inexpensive by-product of having solved (P)—perhaps as the optimal multiplier vector associated with the $A_1x \geqq b_1$ constraints in some LP problem related to (P). Then the previously described method could be applied with (LR_{λ^0}) in the role of (R^0). If the optimal value of (LR_{λ^0}) turns out to be the same as that of (P), step 2 of the method requires solving (LR_{λ^0}) for different values of c, b and A. This can be done quite efficiently (possibly analytically) thanks to the assumed simplicity of (LR_{λ^0}), provided that changes to A_2 and b_2 do not destroy this simplicity.

Other types of relaxations can be used to furnish (R^0). One possibility is the group theoretic relaxation initiated by R. Gomory. Shapiro [18] has recently worked out the details of a procedure for the all 0-1 case that can be viewed as the above method with a particular group theoretic relaxation [1] in the role of (R^0). The procedure has not been implemented computationally.

An entirely different source of sufficient conditions for integer programming optimality derives from the many equivalent representations into which any integer

linear program may be cast, as by applying nonsingular transformations of various kinds. Some of these representations may render totally obvious the optimality of a particular feasible solution. Think, for instance, of the equivalent representation given in the final tableau of the ordinary Simplex method. By inspection one can see that an optimal solution is obtained by setting the final nonbasic variables equal to zero; for this leads to feasible values of the basic variables, and a nonzero value for any nonbasic variable could only worsen the objective function value. All of the standard sensitivity/ranging results in linear programming can be deduced easily by simply perturbing this equivalent problem representation as necessary to correspond with perturbations of the original problem, and then determining *by inspection* whether the associated solution of the perturbed original problem is still feasible and optimal. The same approach can be attempted in integer linear programming, though not necessarily using the same class of nonsingular (Gaussian pivot) transformations. Bowman has developed this approach in the all-integer case for a class of transformations leading to so-called Hermitian basic solutions [3]. It has not been shown, however, that a representation of this form corresponding to an optimal solution exists for an arbitrary all-integer linear program. Computational implementation has not yet been undertaken.

Fleisher and Meyer [4] have very recently proposed some new sufficient optimality conditions for pure and mixed integer linear programming. They are currently investigating the possible usefulness of these conditions for sensitivity analysis.

3. Families with a Single Parameter in the Objective Function or Right-Hand Side: Drawing Conclusions after Solving but a Few Members

In §1, we defined the P^θ-family to be parametric in the objective function,

$$\underset{x \geqq 0}{\text{Minimize}}\,(c + \theta f)x \text{ subject to } Ax \geqq b, \quad x_j \text{ integer}, j \in I, \tag{P^θ}$$

and the P_θ-family to be parametric in the right-hand side,

$$\underset{x \geqq 0}{\text{Minimize}}\; cx \text{ subject to } Ax \geqq b + \theta r, \quad x_j \text{ integer}, j \in I. \tag{P_θ}$$

In both cases we may take $0 \leqslant \theta \leqslant 1$. What can be said about the optimal values or optimal solutions of other members of these families after having optimized for one or more values of θ? Ideally we would like to be able to infer optimal solutions for different values of θ, but partial information is also of interest.

It is clear that the results of the previous section are applicable here. Corollaries 2.2.3, 2.2.4, and Proposition 2.3 are pertinent to the P^θ-family, Corollary 2.2.2 is pertinent to the P_θ-family when $r \geqq 0$ or $r \leqq 0$, and the method based on Proposition 2.5 is pertinent to both families and rendered much simpler because data coefficients can vary only in a one-dimensional fashion.

It will be convenient to discuss the two families of problems separately.

A. *The P_θ-Family*

An important distinction is whether or not the P_θ-family is monotone.

DEFINITION 3.1. The P_θ-family is said to be *monotone* if $r \geqq 0$ or $r \leqq 0$.

It suffices to discuss the case $r \geqq 0$, as the case $r \leqq 0$ can be reduced to the former case via an equivalent parameterization $\gamma = 1 - \theta$.

The obvious and essential characteristic of a monotone P_θ-family is that the members of such a family are nested restrictions of one another or, what is the same thing, are nested relaxations of one another (depending on whether one views θ as increasing from 0 to 1 or decreasing from 1 to 0).

PROPOSITION 3.2. *When* $r \geqq 0$, *the optimal value of* (P_θ) *is monotone nondecreasing on* [0, 1] *and any solution feasible at* θ' *remains feasible for all* $\theta \leqslant \theta'$.

COROLLARY 2.2.5. *When* $r \geqq 0$, *any optimal solution of* $(P_{\theta'})$ *remains optimal at all* $\theta \geqslant \theta'$ *for which it remains feasible.*

For any pure integer problem having a bounded feasible region at $\theta = 0$, Corollary 2.2.5 (which, as the numbering indicates, is based on Proposition 2.2) implies that only a finite number of reoptimizations need be performed if θ commences at 0 and advances to 1.

There is very little that can be said in general about a nonmonotone P_θ-family because then changes in θ shrink some parts of the feasible region while expanding other parts. Nothing like the two results above is generally true.

Whether or not a P_θ-family is monotone, however, lower bounds on the optimal value over all θ are usually available as a relatively inexpensive by-product of having solved any single member of the family. The reason is that the solution to an integer linear program usually involves the generation of reasonably good relaxations of it, the most important types of which can be solved at other values of θ to yield corresponding lower bounds. This is particularly convenient to do with the conventional LP relaxation, for which efficient parametric linear programming techniques can be used, and also with Lagrangean relaxation when θr does not upset the structure of the "special" constraints (see §2.B).

Such lower bounds can be used in conjunction with upper bounds obtained from known feasible solutions to obtain hopefully useful epsilon-optimality conclusions about members of the P_θ-family not yet explicitly addressed computationally.

B. *The* P^θ*-Family*

The P^θ-family is a good deal more convenient to work with than the P_θ-family. There are two reasons for this. One is that changes in θ do not affect the feasible region, and the other is that the following very strong characterization is available for the optimal value as a function of θ.

PROPOSITION 3.3. *The optimal value of* (P^θ) *for* $0 \leqslant \theta \leqslant 1$ *is piecewise-linear, continuous, and concave on its finite domain.*

This well-known result (e.g., [12]) is a direct consequence of the fact that the feasible region of P^θ can be replaced in principle by its convex hull (the so-called integer polyhedron) without altering the optimal value, thereby reducing the P^θ-family in principle to the case of ordinary parametric linear programming in the objective function—a case for which the given characterization has long been established.

In most applications the feasible region will be bounded, in which case there will be a finite number of segments to the piecewise-linear optimal value function on [0, 1]. Thus the P^θ-family is inherently finite rather than infinite.

The great utility of Proposition 3.3, in conjunction with the fact that θ does not affect the feasible region, is illustrated in Figures 1A and 1B. The first figure depicts the objective function value, as a function of θ, of the optimal solutions to (P^0) and (P^1) and also of an intermediate feasible solution $\hat{x}$ to one of these problems. The shaded area represents the region of uncertainty concerning the optimal value of (P^θ). The second figure illustrates how the region of uncertainty is diminished after solving $(P^{0.5})$. In these figures and in general, the upper boundary of the region of uncertainty —call it UB(θ)—is determined by the lower envelope (pointwise minimum) of the linear functions $cx^k + \theta f x^k$ determined by the known feasible solutions $x^1, \ldots, x^k, \ldots$ to (P^θ), while the lower boundary—call it LB(θ)—is determined by

linear interpolation between the plotted points corresponding to the optimal values of the solved members of the P^θ-family. The validity of UB($\cdot$) is evident from the fact that θ does not affect feasibility, while the validity of LB($\cdot$) is a direct consequence of Proposition 3.3.

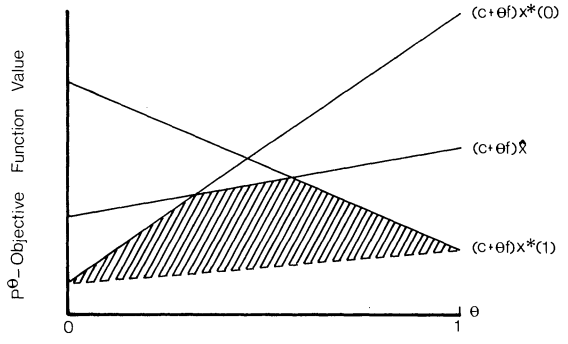

FIGURE 1A

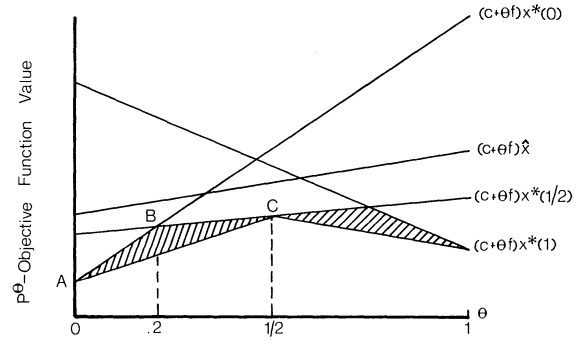

FIGURE 1B

In the illustration of Figure 1 and in most other cases one can contrive, the optimal value of (P^θ) as a function of θ is pinned down quite quickly as successive members of the family are solved, especially if a little care is taken to choose each new member to be solved in such a way that its θ tends to coincide with the maximum difference $\mathrm{UB}(\theta) - \mathrm{LB}(\theta)$. In fact, it is easy to see that the difference is reduced to 0 over some interval whenever the optimal value of (P^θ) for some new choice of θ happens to coincide with the current UB($\cdot$) or LB($\cdot$) function. Thus in Figure 1B an optimal value of $(P^{0.2})$ equal to UB(0.2) or LB(0.2) would completely eliminate the region of uncertainty over the interval $[0, \frac{1}{2}]$ (the optimal value function for (P^θ) would necessarily coincide with $\overline{ABC}$ in the first case and $\overline{AC}$ in the second).

It was noted in connection with Proposition 2.3 that lower bounds are often available (e.g., from penalties) on the amount by which the optimal value of a problem would increase if one of the integer variables were forced to take a different value from its value in the computed optimal solution. Such bounds can be used in conjunction with the UB($\cdot$) bounds to yield simple sufficiency tests for whether a variable can be fixed at a certain value for a certain θ-interval without loss of optimality. For instance, suppose UB($\cdot$) is as shown in Figure 2 after having solved (P^0) and (P^1). Suppose further that the 0-1 variable x_{j_0} takes on the value 1 in both $x^*(0)$ and $x^*(1)$. Let Δ^0 [resp. Δ^1] be a penalty (lower estimate) for the increase that would be caused in the optimal value of (P^0) [resp. (P^1)] by forcing x_{j_0} to have value 0 rather than 1. It follows from Proposition 3.3 that the dotted line is a valid extension of the two bounds to the entire interval $0 \leqslant \theta \leqslant 1$, and hence that x_{j_0} can be fixed at 1

without loss of optimality for $0 \leqslant \theta \leqslant \theta_1$ and $\theta_2 \leqslant \theta \leqslant 1$, where θ_1 and θ_2 are the points at which the dotted line intersects UB(·) in Figure 2.

It is easy to devise a number of variants of the result pictured in Figure 2. Any attempt to formalize the general case would only obscure the inherent simplicity of the underlying rationale.

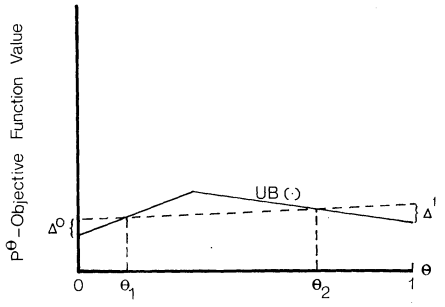

FIGURE 2

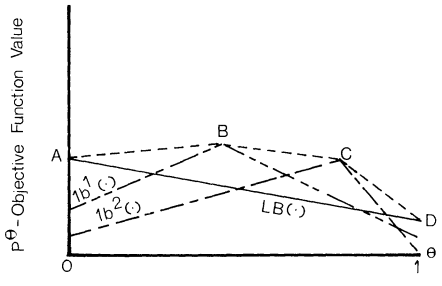

FIGURE 3

We have previously noted the common availability of various lower bounds on the optimal value of (P^θ) as a by-product of relaxations used to solve one or another member of the family. If such bounds are available they can be used to strengthen the interpolative LB(·) bounds mentioned above.

COROLLARY 3.3.1. *If* LB(·) *is the previously defined lower bound function on the optimal value of* (P^θ) *and* lb(·) *is another valid lower bound function (perhaps obtained from some relaxation), then the upper concave envelope of* LB(·) *and* lb(·) *is also a valid lower bound function.*

Figure 3 illustrates this corollary of Proposition 3.3 for a situation where (P^0) and (P^1) have been solved, yielding points A and D, and where two relaxations yield two other lower bound functions on the optimal value of (P^θ). The upper concave envelope $\overline{ABCD}$ is also a valid lower bound function.

One additional result can be stated for (P^θ), namely that the individual objective function components $cx^*(\theta)$ and $fx^*(\theta)$ vary monotonely on [0, 1] for any optimal solution $x^*(\theta)$. The result is well known from other contexts, is easy to prove, and is useful by itself as well as in conjunction with Proposition 3.3 as a source of interpolative and extrapolative bounds on $cx^*(\theta)$ and $fx^*(\theta)$ when $x^*(\cdot)$ has been determined over just a subset of the values of interest.

PROPOSITION 3.4. *If* $0 \leqslant \theta_1 \leqslant \theta_2$, *then* $cx^*(\theta_1) \leqslant cx^*(\theta_2)$ *and* $fx^*(\theta_1) \geqslant fx^*(\theta_2)$.

4. Using Conventional Branch-and-Bound Wisely to Solve a Family of Related Problems

Previous sections have dealt with aspects that are largely independent of the computational method used to solve individual members of the family of integer linear programs being addressed. The presumption was that only conventional methods were available. In this section we are concerned with how to exploit the inherent flexibility of conventional integer programming methods so as to enhance their computational efficiency for solving families of related problems. Serious modifications to conventional methods are deferred until the next section; here we shall make use only of the user options available with most integer programming systems.

Most of the observations made in previous sections provide directly applicable guidance for the effective use of a given integer programming code. When confronting a family of problems, one should plan to solve the members in a sequence that

enhances the opportunities for the results of §§2 and 3 to yield useful information. One should be especially alert for partial orderings of the family based on the relation of restriction (or relaxation).

Nearly all branch-and-bound algorithms have provision for user-supplied bounds, both upper and lower, on the optimal value of a problem. Such prior information usually improves the performance of the algorithm until better bounds are found in the course of the computations. Prior knowledge of a good feasible solution is useful for the upper bound it provides on the optimal value, and also as the optimal solution itself in case the algorithm should demonstrate that no better solution exists. Knowledge of part of an optimal solution can be used to reduce the size of the problem by fixing the appropriate variables permanently at their optimal values. These kinds of prior information—part or all of an optimal or good feasible solution, and lower and upper bounds on the optimal value—are frequently available at nominal computational cost for other members of the problem family upon solution of one member. So it is important to select the order of solution carefully.

The family of integer programs rendered most tractable by the observations of §§2 and 3 is probably the P^{θ}-family. The discussion attending Figure 1 even suggests an efficient procedure for selecting the next choice θ^{k+1} at which to optimize depending on the outcomes for $\theta^1, \ldots, \theta^k$. See [11, Sec. V.D] and [15, Appendix 9A] for details concerning such procedures.

The second most tractable is the P_{θ}-family when it is monotone. A natural (and finite) sequence of θ values for optimization in the pure integer case is to begin at $\theta^1 = 0$ and to select θ^{k+1} as the value just below which the optimal solution for θ^k becomes infeasible (refer to Corollary 2.2.5). The opposite direction of traversal is attractive when only a finite subset of θ's in the unit interval is of interest, since then the optimal solution for each problem must be a (hopefully good) feasible solution for the next.

More generally, and beyond the usually obvious ways of using the results of §§2 and 3, the challenge faced by a user with an entire family of problems to solve is this: How can one take advantage of the information generated by the solution to one member in order to reduce the amount of computational work necessary to solve another member?

Some possible answers to this question are as follows.

1. Take the optimal solution to the first problem and try to modify it so as to be a good feasible solution to the next problem. This can be approached as a manual revision based on insight into the nature of the problem family (e.g., reduce the levels of the least marginally profitable activities in order to accommodate a reduced resource level). It can also be approached more formally, as by using linear programming to reoptimize the continuous variables with the integer variables held at their optimal values from the first problem.

2. Assign at least some branching priorities based on an anaylsis of the solution history of the first problem. Generally it is desirable to branch early on the integer variables which have the greatest impact on the solution, and most commercial codes have provision for user-supplied branching priorities.

3. Initialize the pseudocosts [2], [6] at values obtained while solving the first problem.

4. In a large problem, solve the initial LP relaxation as a revision to a basis stored from the first problem.

5. Redesigning Branch-and-Bound to Solve a Family of Related Problems

We now consider how conventional branch-and-bound could be modified to deal more effectively with entire families of integer linear programs.

At least three approaches are possible:

(i) Proceed in the same spirit as §4, but perform additional calculations in the course of solving each problem expressly for the purpose of providing added information for subsequent use when solving other members of the family.

(ii) Generalize in a natural manner the basic concepts of branch-and-bound from the context of a single isolated integer problem to the context of a family of related problems.

(iii) Devise a procedure for recovering satisfaction of the termination conditions by which branch-and-bound demonstrates the optimality of the final incumbent.

Each approach is discussed in turn. See [11] for additional discussion and computational experience pertinent to all three approaches.

A. *Additional Calculations to Improve the Conventional Approach of* §4

§4 described how conventional branch-and-bound codes could be adapted without significant internal modification for use with an entire family of problems. That discussion is suggestive of the kinds of internal modifications that might be useful. Probably the most important ones would be (a) the generation of additional solutions which are likely to be good feasible solutions to other problems in the family, and (b) the generation of improved lower bounds applicable to other problems in the family—as by reoptimizing the initial LP relaxation for the entire family at the time the first member is undertaken. The options are, for the most part, self-evident. For this reason, and also because a detailed discussion would have much in common with the development of the next subsection, we shall describe here only a proposal of Piper and Zoltners [14] for the pure integer case that has been tested computationally.

Piper and Zoltners address primarily the case where every member of the family has the same feasible region:

$$\underset{x \geqq 0}{\text{Minimize}}\ c^h x \text{ subject to } Ax \geqq b, \quad x_j \text{ integer, all } j, \qquad (P)^h$$

for $h = 1, \ldots, H$. They propose a very simple modification of the usual branch-and-bound fathoming criteria so that after solving $(P)^1$, say, instead of having as the final incumbent a single optimal solution of $(P)^1$ one has a set of feasible solutions $\{x^1, \ldots, x^k, \ldots, x^K\}$ satisfying the property:

$$\tilde{Z}^1 \triangleq \text{Max}\{c^1x^1, c^1x^2, \ldots, c^1x^K\} \leqslant c^1x,$$
$$\text{for all } x \text{ feasible in } (P)^1 \text{ other than } x^1, x^2, \ldots, x^K. \qquad (1)$$

Roughly speaking, one has the K best feasible solutions of $(P)^1$. The revised fathoming criteria allow the user to prespecify an upper bound both on K and on the amount of permissible suboptimality (i.e., on $\text{Max}\{c^1x^1, \ldots, c^1x^K\} - \text{Min}\{c^1x^1, \ldots, c^1x^K\}$).

The hope is that one of the solutions $\{x^1, \ldots, x^K\}$, and in particular the one which minimizes c^hx over this set, will be optimal in $(P)^h$ for $h \neq 1$. A sufficient condition for this to be so is

$$\text{Min}\{c^hx^1, \ldots, c^hx^K\} \leqslant \text{optimal value of } (P)^h \text{ with } c^1x \geqslant \tilde{Z}^1 \text{ appended.} \qquad (2)$$

The sufficiency of (2) follows directly from (1) and the fact that for every feasible solution x of $(P)^h$, either $c^1x < \tilde{Z}^1$ or $c^1x \geqslant \tilde{Z}^1$. Now the left-hand side of (2) is easy to compute, but the right-hand side is not. So various simple underestimates (via relaxation) of the right-hand side of (2) are introduced to simplify this sufficient condition (at the expense, of course, of weakening it). For instance,

$$\text{Min}\{c^hx^1, \ldots, c^hx^K\} \leqslant \underset{x \geqq 0}{\text{Min}}\, c^hx \quad \text{subject to } Ax \geqq b, \quad c^1x \geqslant \tilde{Z}^1, \qquad (3)$$

is sufficient for the optimum of $(P)^h$ to be in the set $\{x^1, \ldots, x^K\}$ because (3) implies (2).

In computational tests on several standard pure integer test problems, judicious application of these ideas led to only a modest amount of extra work to find $\{x^1, \ldots, x^K\}$ and yet this set reliably contained the optimal solution for a surprisingly wide range of deviations from c^1. Sufficient condition (3) and others were successful in proving (2) a high percentage of the time, particularly when the deviations from c^1 were small.

The major shortcoming of the Piper-Zoltners approach is that it is impractical for mixed integer programming, and probably also for pure integer programs with a large number of nearly optimal solutions.

B. *Direct Generalization of Branch-and-Bound*

Ordinary branch-and-bound addresses a single integer programming problem. Yet many of the basic concepts generalize naturally if one replaces the single problem by a related *family* of problems. Each relaxation could be replaced by a family of relaxations, each branch could be applied to all members of the family, the incumbent solution would be a family of incumbents, and so on (cf. [11, pp. 15–16]). The exact nature of the generalization is not unique; several variants appear worthy of investigation.

Marsten and Morin [10] have recently worked out the details of one plausible Dakin-class variant for the pure integer case in which the right-hand side varies parametrically (i.e., the P_θ-family). They assign a distinguished role to (P_0) in that the only LPs ever solved have $\theta = 0$, the optimal multiplier vectors of which are used in the natural way to obtain dual bounds for all $0 \leqslant \theta \leqslant 1$. Feasible solutions found at one value of θ are feasibility-checked for other values of θ and used accordingly to update the incumbents. Branching is always performed simultaneously and identically for all $0 \leqslant \theta \leqslant 1$, and a candidate problem is not considered fathomed until it is truly fathomed for all $0 \leqslant \theta \leqslant 1$. Preliminary computational experience for three monotone capital budgeting problems looks promising.

C. *Recovery of the Termination Conditions of Branch-and-Bound*

Postoptimality analysis and parametric techniques for ordinary linear programming can be viewed naturally in terms of recovering the standard termination conditions associated with the Simplex method. These conditions consist of nonnegativity requirements (or nonpositivity requirements, depending on sign conventions) on the border of the final "tableau." The tableau corresponds to an equivalent representation of the original LP problem, and satisfaction of the nonnegativity requirements renders completely obvious the optimality of the associated basic solution.

The termination conditions associated with a branch-and-bound method are of a different sort. There is no final tableau. Instead, there is an exhaustive partition of the solution space of the original integer programming problem along with proof that no cell of this partition can contain a feasible solution superior to the final incumbent. This partition is specified by the fathomed nodes of the complete branch-and-bound tree. The tests by which each of these nodes is fathomed are just sufficient conditions —usually based on some convenient relaxation—that no feasible solution corresponding to the node could possibly be superior to the final incumbent.

Just as in linear programming where one can revise the original problem and check whether the correspondingly revised final tableau still satisfies the termination conditions, so in integer programming can one revise the original problem and check whether the correspondingly revised partition of the solution space can contain a feasible solution superior to the best known feasible solution of the revised problem. Naturally this requires recording information concerning successful fathoming tests during the course of solving the original problem. And just as one can continue

applying the Simplex method (either primal or dual) in order to "clean up" any violations of the terminal conditions of the revised LP problem, so in integer programming can one continue to apply the branch-and-bound method to fathom (refining the partition if necessary) any nodes of the revised partition that are no longer fathomed by the natural revisions of the originally successful fathoming tests.

The pioneer of this approach to postoptimal analysis and parametric techniques in integer programming is G. Roodman [16], [17]. His first paper on the subject developed details in the context of E. Balas' well-known additive algorithm. Limited computational experience with small problems suggests that this approach may be several times more efficient for solving a handful of related integer programs than the brute force method of solving each member of the family from scratch. This paper inspired Piper and Zoltners [13] to rework and refine Roodman's treatment, with particular attention to questions of efficient computer implementation. No computational results were presented. Roodman's second paper [17] recast the development in the far more interesting context of LP-based branch-and-bound. This is an important contribution to which we cannot do justice here. The details are intricate, though straightforward from the viewpoint proposed here if one is sufficiently facile with postoptimal analysis in LP. Encouraging but very preliminary computational experience was reported.

6. Conclusion

The theory and computational techniques currently available for parametric and postoptimality analysis in integer linear programming are clearly in an early stage of their development. They are built almost entirely on (a) elementary bounds related in one way or another to the complementary concepts of relaxation and restriction, (b) the concavity of the optimal value of the (P^θ)-family as a function of θ, (c) the common sense exploitation of the user options available with most branch-and-bound codes, (d) a natural generalization of the branch-and-bound approach from a single problem to an entire family of problems, and (e) the notion of checking and reestablishing by further computations, if necessary, a set of sufficient conditions for optimality (these sufficiency conditions can be expressed in terms of a relaxation or an equivalent problem representation or a partition of the solution space satisfying certain properties).

There is little opportunity for further conceptual development of (a) and (b). What is required now is to bring these concepts into wider use.

Area (c) is a highly cultivated ad hoc art among experienced practitioners of integer programming. Available folk wisdom needs to be pulled together and tested via systematic computational studies.

The big gains remaining are in areas (d) and (e). The major conceptual options for (d) are clear at this time. Proper computational testing will, however, require the most careful attention to the data structures used to support computational implementation. Comparative performance on large or difficult problems of alternative algorithmic options is known to be exquisitely sensitive to the choice of data structures used to implement them. The design of supporting data structures is also critical in area (e). Additional conceptual development in this area is needed to find new and useful sufficient conditions for optimality that exploit knowledge of an optimal solution (better relaxations or equivalent problem representations or perhaps something entirely different).

What new work remains to be done beyond the five areas discussed above? One major new direction would be to develop tailored techniques for various special but important classes of problems. We have made virtually no assumptions in this paper

concerning special features of problem structure; it stands to reason that present methods could be improved and new methods developed if the problem setting were to be specialized. In this connection, one should be especially alert for strong characterizations of the optimal solution or its value as a function of parametric data changes (recall how useful the concavity property of Proposition 3.3 proved to be for the case of the P^θ-family).

Another worthy line of development would be to attempt natural generalizations of other approaches to integer programming besides branch-and-bound. This paper has confined attention to branch-and-bound because it is the only generally successful approach known at the present time. Yet it is possible that a cutting-plane, group theoretic, or other approach may prove superior for parametric and postoptimality analysis once some member of the problem family has been solved. Frank [5] has shown, for instance, that Gomory cuts lend themselves quite readily to such use in principle. Holm and Klein [8] have extended Frank's results and have applied them to Gomory's cutting plane algorithms for both the pure and mixed integer cases. Computational studies are in progress.[2]

[2] The authors are pleased to acknowledge the incisive criticisms of M. Radke. This work was partially supported by the National Science Foundation and by the Office of Naval Research. Reproduction in whole or in part is permitted for any purpose of the United States Government.

References

1. BELL, D. E. AND SHAPIRO, J. F., "A Finitely Convergent Duality Theory for Zero-One Integer Programming," RM-75-33, International Institute for Applied Systems Analysis (July 1975).
2. BÉNICHOU, M., GAUTHIER, J. M., GIRODET, P., HENTGÈS, G., RIBIÈRE, G. AND VINCENT, O., "Experiments in Mixed Integer Linear Programming," *Mathematical Programming*, Vol. 1, No. 1 (1971).
3. BOWMAN, V. J., JR., "Sensitivity Analysis in Linear Integer Programming," *AIIE Technical Papers* (1972).
4. FLEISHER, J. M. AND MEYER, R. R., "New Sufficient Optimality Conditions for Integer Programming and Their Application," Computer Sciences Technical Report, University of Wisconsin, Madison (January 1976).
5. FRANK, C. R., JR., "Parametric Programming in Integers," *Operations Research Verfahren*, Vol. III (1967).
6. GAUTHIER, J. M. AND RIBIÈRE, G., "Experiments in Mixed Integer Linear Programming Using Pseudo-Costs," forthcoming in *Mathematical Programming*.
7. GEOFFRION, A. M., "Lagrangean Relaxation for Integer Programming," *Mathematical Programming Study 2* (1974).
8. HOLM, S. AND KLEIN, D., "Parametric Analysis for Integer Programming Problems," Working Paper 76-4, School of Business and Organizational Sciences, Florida International University (February 1976).
9. JEROSLOW, R. G., "The Principles of Cutting-Plane Theory," Parts I and II, Graduate School of Industrial Administration, Carnegie-Mellon University (February 1974 and August 1975).
10. MARSTEN, R. E. AND MORIN, T. L., "Parametric Integer Programming: The Right-Hand-Side Case," WP 808-75, Sloan School of Management, MIT (September 1975).
11. NAUSS, R. M., "Parametric Integer Programming," Ph.D. Dissertation, available as Working Paper No. 226, Western Management Science Institute, UCLA (January 1975).
12. NOLTEMEIER, H., "Sensitivitatsanalyse bei diskreten linearen Optimierungsproblemen," in M. Beckmann and H. P. Kunzi (eds.), *Lecture Notes in Operations Research and Mathematical Systsms*, No. 30, Springer-Verlag, New York, 1970.
13. PIPER, C. J. AND ZOLTNERS, A. A., "Implicit Enumeration Based Algorithms for Postoptimizing Zero-One Programs," Management Sciences Research Report No. 313, Graduate School of Industrial Administration, Carnegie-Mellon University (March 1973).
14. ——— AND ———, "Some Easy Postoptimality Analysis for Zero-One Programming," *Management Science*, Vol. 22, No. 7 (1976).
15. RADKE, M. A., "Sensitivity Analysis in Discrete Optimization," Ph.D. Dissertation, available as Working Paper No. 240, Western Management Science Institute, UCLA (September 1975).
16. ROODMAN, G. M., "Postoptimality Analysis in Zero-One Programming by Implicit Enumeration," *Naval Research Logistics Quarterly*, Vol. 19 (1972).
17. ———, "Postoptimality Analysis in Integer Programming by Implicit Enumeration: The Mixed Integer Case," The Amos Tuck School of Business Administration, Dartmouth College (October 1973).
18. SHAPIRO, J. F., "Sensitivity Analysis in Integer Programming," Paper presented at the Workshop on Integer Programming, Bonn, Germany, September 1975.