A CAPACITY IMPROVEMENT LOWER BOUND FOR FIXED CHARGE NETWORK DESIGN PROBLEMS

BRUCE W. LAMAR
University of California, Irvine, California

YOSEF SHEFFI
Massachusetts Institute of Technology, Cambridge, Massachusetts

WARREN B. POWELL
Princeton University, Princeton, New Jersey

(Received January 1985; revisions received August 1986, August 1987, June, November 1988, April 1989; accepted May 1989)

Network design problems concern flows over networks in which a fixed charge must be incurred before an arc becomes available for use. The uncapacitated, multicommodity network design problem is modeled with aggregate and disaggregate forcing constraints. (Forcing constraints ensure logical relationships between the fixed charge-related and the flow-related decision variables.) A new lower bound for this problem—referred to as the capacity improvement (CI) bound—is presented; and an efficient implementation scheme using shortest path and linearized knapsack programs is described. A key feature of the CI lower bound is that it is based on the LP relaxation of the aggregate version of the problem. A numerical example illustrates that the CI lower bound can converge to the optimal objective function value of the IP formulation.

This paper focuses on a new lower bound for the class of uncapacitated, multicommodity network design problems whose arc cost function consists of a fixed charge and a linear routing cost. A distinctive feature of this lower bound is that it is based on an aggregate problem formulation; that is, aggregate (rather than disaggregate) constraints are used to enforce logical relationships among the decision variables. Compared to their disaggregate counterparts, aggregate formulations are considered *weaker* in the sense that the LP relaxation of aggregate formulations is generally looser than that of disaggregate ones. We present theoretical and computational results, however, which show that in some circumstances this new lower bound can be as tight as the disaggregate LP relaxation.

The paper is organized into three sections. Section 1 formulates the fixed charge network design problem. Section 2 presents a new lower bound to the integer program and describes an efficient algorithm for computing it. Section 3 illustrates the use of this algorithm by a numerical example.

1. MODEL FORMULATION

The focus of this section is on problem formulation. It includes two parts: The first part defines the notation and formulates the network design problem as an aggregate integer program. The second part introduces the more common disaggregate formulation of the problem considered here and compares it to our aggregate formulation.

1.1. Integer Programming Formulation

The following notation is used to define the problem. Let

N = the set of nodes with generic element n,
A = the set of directed arcs with generic element a,
K = the set of commodities with generic element k.

For each arc $a \in A$, the *tail* and *head* nodes incident to arc a are denoted by $I(a)$ and $J(a)$, respectively. Also, for each node $n \in N$, let A_n denote the set of arcs whose tail node is n and let B_n denote the set of

Subject classifications: Networks/graphs: network design problems. Programming, integer, theory: lower bound for fixed charged problems.

arcs whose head node is n. That is

$$A_n = \{a\colon a \in A \text{ and } I(a) = n\}$$

$$B_n = \{a\colon a \in A \text{ and } J(a) = n\}.$$

For each commodity $k \in K$, let $O(k)$ and $D(k)$ denote, respectively, the *origin* and *destination* nodes for commodity k; and let d_k denote the quantity of commodity k supplied at node $O(k)$ and demanded at node $D(k)$. It is assumed that d_k is nonnegative.

The fixed charge for arc a is denoted by f_a and the routing cost (i.e., the cost per unit flow) for arc a and commodity k is denoted by $c_{a,k}$. It is assumed that these coefficients are nonnegative. In addition, a parameter used in the aggregate problem formulation is $u_a^{\max}$. This parameter denotes the maximum flow coefficient for arc a and can be taken as the sum $\sum_k d_k$. Note that this coefficient is not intended to capacitate the problem; rather, it is used in the problem formulation to enforce logical relationships among the decision variables.

The decision variables in the problem designate which arcs are selected in the network design and how much flow is carried on each of the selected arcs. For each arc a, let $y_a = 1$ if arc a is selected to be in the network and let $y_a = 0$ otherwise. Also, let $x_{a,k}$ denote the flow of commodity k carried on arc a and let x_a denote the sum of the commodity flows carried on arc a. The $\{x_{a,k}\}$ and $\{x_a\}$ decision variables are referred to, respectively, as the commodity specific and the aggregate arc flows.

As part of the problem formulation, let X denote the set of (aggregate) arc flow vectors $\mathbf{x} = (\ldots, x_a, \ldots)$ that conform to the multicommodity flow balance equations

$$\sum_{a\in A_n} x_{a,k} - \sum_{a\in B_n} x_{a,k} = \begin{Bmatrix} d_k & \text{if } n = O(k) \\ -d_k & \text{if } n = D(k) \\ 0 & \text{otherwise} \end{Bmatrix} \quad \text{for any } n, k \tag{1a}$$

$$x_a = \sum_k x_{a,k} \qquad \text{for any } a \tag{1b}$$

$$x_{a,k} \geq 0 \qquad \text{for any } a, k. \tag{1c}$$

(Unless denoted otherwise, "$\sum$" and "for any" include all elements of the relevant set. For instance, in (1b), "$\sum_k$" means "$\sum_{k\in K}$" and "for any a" means "for any $a \in A$".)

The uncapacitated, multicommodity fixed charge network design problem (denoted **FCND**) can be formulated as the following integer program.

Program FCND

$$\min_{\mathbf{x}\in X} \sum_a f_a \cdot y_a + \sum_k \sum_a c_{a,k} \cdot x_{a,k} \tag{2a}$$

subject to

$$x_a \leq u_a^{\max} \cdot y_a \quad \text{for any } a \tag{2b}$$

$$y_a \in \{0, 1\} \quad \text{for any } a. \tag{2c}$$

Let $\mathbf{y}^* = (\ldots, y_a^*, \ldots)$ denote the optimal arc selection vector, let $\mathbf{x}^* = (\ldots, x_a^*, \ldots)$ denote the optimal (aggregate) arc flow vector, and let z^***[FCND]** denote the optimal objective function value of program **FCND**. (Throughout this paper, the optimal objective function value for any problem **P** is denoted as $z^*[P]$.)

Objective function (2a) minimizes the total system costs, including both fixed and variable costs, for all arcs and commodities in the network. Carrying the minimization over set X guarantees that the arc flows for each commodity are restricted to feasible paths in the network. The aggregate *forcing* constraints (2b) ensure that an arc carries flow only if it is selected to be in the network. Constraints (2c) ensure the integrality of the arc selection decision variables, $\{y_a\}$.

The LP relaxation of program **FCND** is formed by replacing the integrality conditions (2c) with the nonnegativity constraints

$$y_a \geq 0 \quad \text{for any } a. \tag{3}$$

Observe that this relaxation is equivalent to the following shortest path program (with flow assignment).

Program SP

$$\min_{\mathbf{x}\in X} \sum_k \sum_a \left(\frac{f_a}{u_a^{\max}} + c_{a,k}\right) \cdot x_{a,k}. \tag{4}$$

Let $\bar{\mathbf{x}}^* = (\ldots, \bar{x}_a^*, \ldots)$ denote the optimal (aggregate) arc flow vector and let z^***[SP]** denote the optimal objective function value of program **SP**.

The equivalence between program **SP** and the original LP relaxation (using (1), (2a), (2b) and (3)) can be shown by using Balinski's (1961) observation that constraints (2b) will always be satisfied with equality in the LP relaxation of program **FCND**. Thus, since $u_a^{\max} > 0$ for any a, constraints (2b) can be solved explicitly for $\{y_a\}$. Substituting $\{x_a/u_a^{\max}\}$ for $\{y_a\}$ yields the formulation given in (4). (Note that constraints (3) can be omitted in **SP** because the nonnegativity of $\{y_a\}$ is ensured by the fact that the arc flows in set X are required to be nonnegative.)

The analysis conducted later in this paper involves an ensemble of shortest path programs in which the maximum flow parameter vector $\mathbf{u}^{\max} = (\ldots, u_a^{\max}, \ldots)$ is altered systematically. Thus, let **SP(u)** denote the shortest path program of the form of (4) in which the vector $\mathbf{u} = (\ldots, u_a, \ldots)$ is substituted for $\mathbf{u}^{\max}$. (In this paper, it will be convenient to use both **SP** and **SP**($\mathbf{u}^{\max}$) to refer to the LP relaxation of program **FCND**.)

The significance of basing a lower bound on program **SP** is discussed next.

1.2. Aggregate Versus Disaggregate Formulations

As mentioned earlier, the forcing constraints (2b) in program **FCND** are aggregated (i.e., summed) over commodities. Many researchers, however, have chosen to formulate the fixed charge network design problem using disaggregate forcing constraints (Magnanti and Wong 1984). In the disaggregate formulation, denoted **DFCND**, constraints (2b) are replaced with

$$x_{a,k} \leqslant d_k \cdot y_a \quad \text{for any } a, k. \tag{5}$$

Observe that programs **FCND** and **DFCND** are equivalent: they have the same feasible region, optimal solution(s), and objective function value. The reason most analysts have preferred formulation **DFCND** is that the feasible region of its LP relaxation, denoted **DLP**, is contained within the feasible region of program **SP** (the LP relaxation of program **FCND**). Consequently, the following relationships hold (Rardin 1982)

$$\begin{aligned} z^*[\mathbf{SP}] &\leqslant z^*[\mathbf{DLP}] \\ &\leqslant z^*[\mathbf{DFCND}] = z^*[\mathbf{FCND}]. \end{aligned} \tag{6}$$

In short, the disaggregate formulation yields a tighter LP relaxation than that of the aggregate model. In fact, in many cases the solution to **DLP** is also optimal (or nearly optimal) in **DFCND** (and **FCND**).

In light of the discussion above, it is somewhat surprising that our lower bound is based on a seemingly weaker aggregate formulation. Our reasons are twofold. First, because it is a shortest path program, **SP(u)** is easy to solve; and second, as shown in the next section, the *capacity* parameter vector, **u**, can be adjusted to produce bounds that are significantly tighter than the LP relaxation of the aggregate program formulation.

2. CAPACITY IMPROVEMENT PROCEDURE

The capacity improvement (CI) procedure presented in this section is a method of obtaining a lower bound to z^*[**FCND**]. Naturally, since **SP** is the LP relaxation of **FCND**, z^*[**SP**] is a lower bound to z^*[**FCND**]. However, as mentioned in connection with (6), because **FCND** is an aggregate formulation, its LP relaxation will, in general, yield a loose lower bound. A tighter lower bound can be obtained by using a capacity parameter vector, **u**, that is smaller than $\mathbf{u}^{\max}$. Clearly, then, $z^*[\mathbf{SP(u)}] \geqslant z^*[\mathbf{SP}]$. The *trick* to this approach is to determine conditions on **u** such that $z^*[\mathbf{SP(u)}] \leqslant z^*[\mathbf{FCND}]$. This is the idea behind the CI procedure described in this section.

The presentation here is divided into three parts: The first part proves the validity of the lower bound developed by the CI procedure; the second part shows that this lower bound is easy to compute because it can be determined by solving a set of linearized knapsack problems; and the third part demonstrates how the CI procedure can be used in an iterative fashion to obtain a successively tighter lower bound to z^*[**FCND**].

2.1. Lower Bound

The following paragraphs develop a method for obtaining a lower bound to z^*[**FCND**] that is at least as tight as z^*[**SP**]. The approach uses a value t, referred to as the *target value*. Suppose, for purposes of discussion, that t is known to be an upper bound to z^*[**FCND**]. This information can be used to determine an upper bound (or *capacity*) on the flow of a generic arc, say arc b, in the optimal solution to **FCND**. Let $\bar{w}_b$ be a trial value of such a capacity and consider the program $\mathbf{A}_b(\bar{w}_b)$ (referred to as the *auxiliary program for arc b*) obtained by adding the constraint

$$x_b \geqslant \bar{w}_b \tag{7}$$

to program **SP**, the LP relaxation of **FCND**. If $z^*[A_b(\bar{w}_b)]$ is greater than or equal to t (and t, by assumption, is greater than z^*[**FCND**]), then clearly the flow on arc b cannot exceed $\bar{w}_b$ in any optimal solution to **FCND**. Thus, $\bar{w}_b$ is a valid capacity for the flow on arc b. The *best* capacity parameter (that is, the tightest valid upper bound on flow) is identified by finding the minimum of $u_b^{\max}$ and the smallest value of $\bar{w}_b$ such that $z^*[\mathbf{A}_b(\bar{w}_b)]$ is greater than or equal to t. That is, let

$$w_b(t) = \min\{\bar{w}_b\colon z^*[\mathbf{A}_b(\bar{w}_b)] \geqslant t\} \tag{8}$$

and define $u_b(t)$, referred to as an *improved capacity parameter for arc b*, as

$$u_b(t) = \min\{w_b(t), u_b^{\max}\}. \tag{9}$$

(Program $\mathbf{LK}_b(t)$, presented shortly, finds such a capacity parameter.)

The procedure described in the preceding paragraph can be performed for each arc $b \in A$ (or any subset of the arcs contained in A). That is, for each arc $b \in A$, a separate auxiliary program $\mathbf{A}_b(\bar{w}_b)$ is created and (8) and (9) are used to determine the improved capacity parameter for that arc. This, then, determines an improved capacity parameter *vector*, $\mathbf{u}(t) = (\ldots, u_b(t), \ldots)$.

Next, consider program $\mathbf{SP}(\mathbf{u}(t))$. (Remember that $\mathbf{SP}(\mathbf{u}(t))$ is the shortest path program defined in (4) with $\mathbf{u}^{\max}$ replaced by $\mathbf{u}(t)$.) Because $t > z^*[\mathbf{FCND}]$, this means that $\mathbf{u}(t)$ is an upper bound to $\mathbf{x}^*$ (where $\mathbf{x}^*$ is the optimal aggregate arc flow vector in **FCND**) and so, the optimal solution to **FCND** is contained in the feasible region of $\mathbf{SP}(\mathbf{u}(t))$. Thus, $z^*[\mathbf{SP}(\mathbf{u}(t))]$ is a valid lower bound to $z^*[\mathbf{FCND}]$.

The discussion above assumed that t was known to be an upper bound to $z^*[\mathbf{FCND}]$. Now suppose that t is an arbitrarily selected value. Then either t must itself be a lower bound to $z^*[\mathbf{FCND}]$ or (if $t > z^*[\mathbf{FCND}]$) then $z^*[\mathbf{SP}(\mathbf{u}(t))]$ must be a valid lower bound to $z^*[\mathbf{FCND}]$. This information can be combined to define the *capacity improvement* (CI) *lower bound*, $\ell(t)$. Specifically

$$\ell(t) = \min\{t, z^*[\mathbf{SP}(\mathbf{u}(t))]\}. \tag{10}$$

Note, in addition, that since $u_b(t)$ is less than or equal to $u_b^{\max}$ for all arcs (see (9)), then $z^*[\mathbf{SP}(\mathbf{u}(t))]$ must be greater than or equal to $z^*[\mathbf{SP}]$, the optimal objective function value of the original LP relaxation. Thus, if t is greater than or equal to $z^*[\mathbf{SP}]$, then so is $\ell(t)$.

In summary, the above discussion has proved the following proposition (Lamar 1985).

Proposition. *If* $t \geqslant z^*[\mathbf{SP}]$, *then* $z^*[\mathbf{SP}] \leqslant \ell(t) \leqslant z^*[\mathbf{FCND}]$.

An efficient method for determining $\ell(t)$ is described next.

2.2. Knapsack Interpretation

The following paragraphs show how the improved capacity parameter vector, $\mathbf{u}(t)$, can be obtained by solving a set of linearized knapsack programs. Since such programs can be solved by a greedy-type algorithm, this means that $\mathbf{u}(t)$ can be determined very efficiently.

This procedure can be described by considering a general arc $b \in A$. Remember that the intermediate capacity parameter, $w_b(t)$, is obtained by adding constraint (7) to program **SP**. To describe the effect of this constraint, let $\Delta_{b,k}$ denote the marginal cost difference between the quantities:

i. the optimal LP routing of a unit of commodity k from $O(k)$ to $D(k)$ using arc b; and
ii. the current LP routing of a unit of commodity k from $O(k)$ to $D(k)$ (which is optimal in program **SP** without the constraint that arc b be used).

This marginal cost can be decomposed into the components:

- the linearized unit cost on arc b, (11a)
- the shortest path from $O(k)$ to $I(b)$, (11b)
- the shortest path from $J(b)$ to $D(k)$, (11c)
- the shortest path from $O(k)$ to $D(k)$. (11d)

The marginal cost $\Delta_{b,k}$ is then computed as (11a) plus (11b) plus (11c) minus (11d).

Observe that $\Delta_{b,k}$ can be determined directly from the solution of program **SP**. To see this, let $\{v_{n,k}\}$ denote the optimal dual variables associated with constraints (1a) in program **SP** and note that, for any commodity k, the shortest path from any node m to any node n is given by $v_{n,k} - v_{m,k}$. Thus, combining terms in (11) yields

$$\Delta_{b,k} = \frac{f_b}{u_b^{\max}} + c_{b,k} + v_{I(b),k} - v_{J(b),k}. \tag{12}$$

Notice, however, that the right-hand side of (12) is simply the reduced cost for arc b and commodity k and thus is available directly from the optimal solution to program **SP**.

Consider the following linear program—with decision variables $\{r_{b,k}\}$ and parameter t—which is also focused on a particular arc b.

Program $\mathbf{LK}_b(t)$

$$\text{Maximize} \quad \sum_k d_k \cdot r_{b,k} \tag{13a}$$

subject to

$$\sum_k (d_k \cdot \Delta_{b,k}) \cdot r_{b,k} \leqslant t - z^*[\mathbf{SP}] \tag{13b}$$

$$0 \leqslant r_{b,k} \leqslant 1 \quad \text{for any } k. \tag{13c}$$

Observe that this program is a linearized 0–1 knapsack program and thus can be solved simply by ranking the $|K|$ marginal costs $\{\Delta_{b,k}\}$ in increasing order (Dantzig 1957). By using efficient sorting techniques (see, for example, Ahrens and Finke 1975), this program can be solved very fast.

The purpose of program $\mathbf{LK}_b(t)$ is to determine the value of $w_b(t)$. Let $\{r_{b,k}^*\}$ denote the optimal value of the decision variables in program $\mathbf{LK}_b(t)$. The commodities, k, with the smallest marginal cost, $\Delta_{b,k}$, will be associated with the decision variables with positive optimal value in program $\mathbf{LK}_b(t)$. These same commodities will be the ones routed through arc b in the optimal solution to the auxiliary program, $\mathbf{A}_b(\bar{w}_b)$. Thus, the aggregate flow carried on arc b can be expressed as $\sum_k d_k \cdot r^*_{b,k}$. This means that, for any given value to the parameter t, the intermediate capacity parameter, $w_b(t)$, is simply

$$w_b(t) = z^*[\mathbf{LK}_b(t)]. \tag{14}$$

Once $w_b(t)$ has been determined using (14), the improved capacity parameter, $u_b(t)$, can be obtained by the minimization given in (9). Similarly, the improved capacity parameter vector, $\mathbf{u}(t) = (\ldots, u_b(t), \ldots)$, can be computed by solving a separate linearized knapsack program for each arc $b \in A$. The lower bound, $\ell(t)$, then can be determined using (10).

An important feature of the procedure outlined in the preceding paragraphs is that $\ell(t)$ is relatively easy to compute. As described above, the calculation of the coefficients $\{\Delta_{b,k}\}$ is straightforward and the solution of program $\mathbf{LK}_b(t)$ can be determined by a greedy-type algorithm. This means that it is easy to determine the vector $\mathbf{u}(t)$ used in program $\mathbf{SP}(\mathbf{u}(t))$. And, since $\mathbf{SP}(\mathbf{u}(t))$ is itself easy to solve (it is a shortest path program), the lower bound, $\ell(t)$, can be determined very efficiently.

The techniques just introduced can also be used iteratively to generate a successively tighter lower bound. This procedure is discussed next.

2.3. Iterative Procedure

The CI procedure can be used within an iterative framework to obtain a successively tighter lower bound to $z^*[\mathbf{FCND}]$. The concept here is to use the improved capacity parameter vector from the previous iteration to determine the vector for the current iteration. (Thus, the material presented earlier in this section can be viewed as the initial iteration.) See Lamar for a proof that this procedure produces a sequence of nondecreasing lower bounds to $z^*[\mathbf{FCND}]$.

To describe the iterative process, let i be the iteration counter for $i = 1, 2, \ldots$. Iterations $i - 1$ and i are referred to, respectively, as the *previous* and *current* iterations. For a given target value, t, let $\mathbf{u}^i(t)$ denote the current improved capacity parameter vector and let $\ell^i(t)$ denote the current CI lower bound to $z^*[\mathbf{FCND}]$. (By definition, let $\mathbf{u}^0(t) \equiv \mathbf{u}^{\max}$ and let $\ell^0(t) \equiv z^*[\mathbf{SP}]$.)

Each iteration of the procedure is comprised of three steps. First, for each arc $b \in A$, the current intermediate capacity parameter, $w^i_b(t)$, is set equal to the optimal objective function value of a linearized knapsack program of the form of (13) except that $z^*[\mathbf{SP}(\mathbf{u}^{i-1}(t))]$ rather than $z^*[\mathbf{SP}]$ is used in constraint (13b) and the marginal cost coefficients, $\{\Delta_{b,k}\}$, are computed from the reduced costs of program $\mathbf{SP}(\mathbf{u}^{i-1}(t))$ rather than program $\mathbf{SP}$. Second, the current improved capacity parameter for arc b, $u^i_b(t)$, is computed as the minimum of $w^i_b(t)$ and $u^{i-1}_b(t)$; and the current improved capacity parameter vector, $\mathbf{u}^i(t) = (\ldots, u^i_b(t), \ldots)$, is obtained by computing $u^i_b(t)$ for each arc $b \in A$. Finally, the current shortest path program, $\mathbf{SP}(\mathbf{u}^i(t))$, is solved and the current lower bound, $\ell^i(t)$, is computed as the minimum of t and $z^*[\mathbf{SP}(\mathbf{u}^i(t))]$.

The algorithm continues until $\ell^i(t) - \ell^{i-1}(t)$, the relative improvement of the lower bound between iterations, becomes sufficiently small. The next section illustrates the operation of the CI procedure with a simple example.

3. NUMERICAL EXAMPLE

The purpose of the example given in this section is to demonstrate that the CI lower bound can converge to the optimal objective function value of the aggregate and disaggregate integer programs (i.e., $\ell^i(t) = z^*[\mathbf{FCND}] = z^*[\mathbf{DFCND}]$). The multicommodity network used here is shown in Figure 1. For convenience, let the number of commodities, $|K|$, be denoted as $\bar{k}$. For $k = 1, 2, \ldots, \bar{k}$, commodity k originates at node 0 and terminates at node k. Each demand d_k is assumed to be unity. Arc (0, 1) is designated as arc b. It has a fixed charge of $f_b = 1$ and a routing cost of $c_{b,k} = 1$ for each commodity k. All other arcs have zero fixed charge and zero routing costs. The maximum possible flow on arc b is $\sum_k d_k = \bar{k}$ and so $u^{\max}_b = \bar{k}$.

The optimal solution to the fixed charge network design problem for the network in Figure 1 can be obtained by inspection. It consists of sending a unit of flow over each arc $(0, k)$ for $k = 1, 2, \ldots, \bar{k}$ and zero flow over all other arcs. The optimal objective function value is the cost of sending flow over arc b. Thus, $z^*[\mathbf{FCND}] = z^*[\mathbf{DFCND}] = z^*[\mathbf{DLP}] = 2$, and $z^*[\mathbf{SP}] = 1 + (1/\bar{k})$. Note that there is no optimality gap for $\mathbf{DLP}$, but that the gap of $\mathbf{SP}$ is $100 \cdot (1 - (1/\bar{k}))/2\%$. Observe also that, as $\bar{k}$ increases, so does the optimality gap of $\mathbf{SP}$ which, as Cornuejols, Fisher and Nemhauser (1977) point out, is a

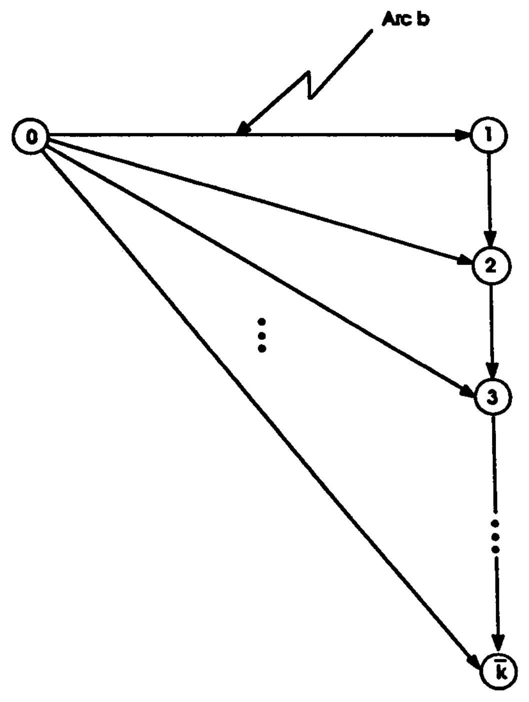

Figure 1. Network for numerical example.

weakness of the aggregate LP relaxation for this class of problems.

For any $\bar{k}$, the optimality gap associated with the aggregate LP relaxation can be reduced by using the CI procedure. The target value, t, for this example is chosen in the range of $1 + (1/\bar{k}) < t < 1 + \bar{k}$. (Target values outside this range are not meaningful because they produce a CI lower bound that is no better than the optimal objective function value of **SP**.) As mentioned before, only one arc has a fixed charge: arc b. Thus, only one capacity parameter needs to be improved (i.e., reduced): $u_b^i(t)$. For this simple network, $u_b^i(t)$ can be expressed as

$$u_b^i(t) = \frac{t \cdot u_b^{i-1}(t)}{1 + u_b^{i-1}(t)} \tag{15}$$

with, by definition, $u_b^0(t) = u_b^{\max} = \bar{k}$. Equation 15, a first-order difference equation, can be solved explicitly. This yields

$$u_b^i(t) = \frac{t - 1}{1 - (1/(t)^i) \cdot [1 - (t-1)/\bar{k}]} \tag{16}$$

where $(t)^i$ denotes the constant t taken to the ith power.

For this example, $z^*[\mathbf{SP}(\mathbf{u}^i(t))]$, the optimal objective function value of program $\mathbf{SP}(\mathbf{u}^i(t))$, equals $(1/u_b^i(t)) + 1$ and thus can be expressed explicitly (after rearranging terms) as

$$z^*[\mathbf{SP}(\mathbf{u}^i(t))] = \frac{t}{t-1} - \frac{1}{(t)^i} \cdot \left(\frac{1}{t-1} - \frac{1}{\bar{k}}\right). \tag{17}$$

Thus, $\ell^i(t)$, the CI lower bound in the ith iteration, can be expressed as

$$\ell^i(t) = \min\left\{t, \frac{t}{t-1} - \frac{1}{(t)^i} \cdot \left(\frac{1}{t-1} - \frac{1}{\bar{k}}\right)\right\}. \tag{18}$$

Figure 2 depicts $\ell^i(t)$ given in (18) as a function of the constant target value, t, for several iterations of the CI algorithm. This figure illustrates the two commodity case (i.e., $\bar{k} = 2$). For $\bar{k} > 2$, a similar set of curves is produced, but the $z^*[\mathbf{SP}]$ line is shifted downward. Observe that for any choice of t in the range $1 + (1/\bar{k}) < t < 1 + \bar{k}$, the CI lower bound is strictly greater than the aggregate LP relaxation.

Also, (16), (17) and (18) can easily be evaluated as $i \to \infty$. Specifically, note that for any $\bar{k}$, because $t > 1$, $1/(t)^i \to 0$ as $i \to \infty$. Thus, in the limit, $u_b^\infty(t) = t - 1$, $z^*[\mathbf{SP}(\mathbf{u}^\infty(t))] = t/(t-1)$, and

$$\ell^\infty(t) = \min\{t, t/(t-1)\}. \tag{19}$$

Equation 19 is depicted in Figure 2 as the line with $i = \infty$. This line represents, for any given target value, the maximum value that the CI lower bound can attain. Observe that, in particular, for the *critical* target value of $t = 2$, the CI lower bound converges to the optimal objective function value of the IP program, thus eliminating the optimality gap associated with the aggregate LP relaxation. This is true regardless of the problem size, $\bar{k}$.

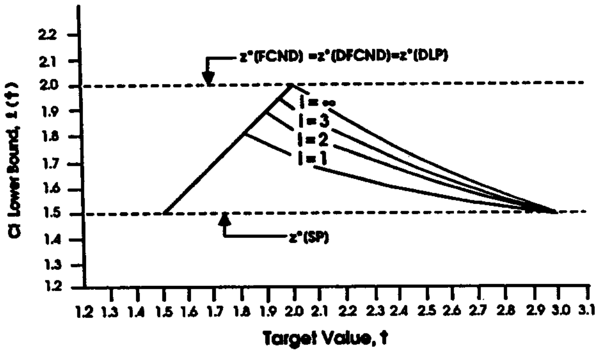

Figure 2. Capacity improvement lower bound for numerical example.

ACKNOWLEDGMENT

This work was supported in part by a research contract from I. U. International to the Massachusetts Institute of Technology and Princeton University. Additional funds were provided by the Institute of Transportation Studies and the Graduate School of Management, University of California, Irvine. The authors acknowledge the many thoughtful suggestions given by the referees of this article. Their comments have resulted in an improved paper.

REFERENCES

AHRENS, J. H., AND G. FINKE. 1975. Merging and Sorting Applied to the 0–1 Knapsack Problem. *Opns. Res.* **23**, 19–32.

BALINSKI, M. L. 1961. Fixed-Cost Transportation Problems. *Naval Res. Logist. Quart.* **8**, 41–54.

CORNUEJOLS, G., M. FISHER AND G. L. NEMHAUSER. 1977. Location of Bank Accounts to Optimize Float: An Analytical Study of Exact and Approximate Algorithms. *Mgmt. Sci.* **23**, 789–810.

DANTZIG, G. B. 1957. Discrete-Variable Extremum Problems. *Opns. Res.* **5**, 266–276.

LAMAR, B. W. 1985. Network Design Algorithms With Applications to Freight Transportation. Unpublished Ph.D. Dissertation, Department of Civil Engineering, Massachusetts Institute of Technology, Cambridge, Mass.

MAGNANTI, T. L., AND R. T. WONG. 1984. Network Design and Transportation Planning: Models and Algorithms. *Trans. Sci.* **18**, 1–55.

RARDIN, R. L. 1982. Tight Relaxations of Fixed Charge Network Flow Problems. Report No. J-82-3, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta.