Mathematical Programming 26 (1983) 64–75
North-Holland Publishing Company

COMPLEXITY OF SOME PARAMETRIC INTEGER AND NETWORK PROGRAMMING PROBLEMS*

Patricia J. CARSTENSEN

Department of Mathematics, University of Michigan, Ann Arbor, MI, U.S.A.

Received 26 October 1981
Revised manuscript received 6 May 1982

Two examples of parametric cost programming problems—one in network programming and one in NP-hard 0–1 programming—are given; in each case, the number of breakpoints in the optimal cost curve is exponential in the square root of the number of variables in the problem.

Key words: Parametric Programming, 0–1 Programming, Network Programming, Complexity.

1. Introduction

In this paper we consider parametric cost versions of several special forms of linear and integer programming; that is, we examine the optimal cost curve $f(\lambda) = \min\{(c + c^*\lambda)x\colon\ Ax = b,\ x \geq 0\}$. In all of our examples the feasible set $\{x\colon Ax = b,\ x \geq 0\}$ will be bounded and non-empty so that $f(\lambda)$ will be finite for all λ. The curve $f(\lambda)$ is obviously piece-wise linear and concave; a natural measure of the complexity of a given class of such problems is the number of points of slope change, or *breakpoints*, that occur in $f(\lambda)$. Zadeh [10] and Murty [5] have shown that for arbitrary linear programs, the number of breakpoints can be exponential in the number of variables. We will give two examples where the number of breakpoints is of the order of $2^{\sqrt{n}}$, where n is the number of variables; we thus resolve the question of the worse case behavior in two further classes of problems.

The first special case we consider is network programming. Let $G = (\mathcal{N}, \mathcal{A})$ be a network, where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of arcs. One node s will be designated the source and a second one t will be the sink. For each arc $(i, j) \in \mathcal{A}$ let k_{ij} be the capacity of (i, j), let $(c_{ij} + c^*_{ij}\lambda)$ be the cost of using the arc at λ, and let x_{ij} be the flow on the arc. The lower bounds on the amount of flow that an arc can carry are all set at zero. Then the problem of finding a minimum cost flow of value v is:

$$\min \sum_{(i,j)\in\mathcal{A}} (c_{ij} + c^*_{ij}\lambda)x_{ij}$$

*This research is partially supported by the Air Force Office of Scientic Research. Air Force Number AFOSR-78-3646.

so that

$$\sum_{j\in A(i)} x_{ij} - \sum_{j\in B(i)} x_{ji} = \begin{cases} v & i = s, \\ -v & i = t, \\ 0 & \text{otherwise}, \end{cases}$$

$$0 \le x_{ij} \le k_{ij}.$$

Here $A(i) = \{j\colon (i, j) \in \mathcal{A}\}$ and $B(i) = \{j\colon (j, i) \in \mathcal{A}\}$.

Since polynomial methods of finding successive breakpoints are known for parametric minimum cost flow problems and since arbitrary optimal cost curves may be computed by methods which are polynomial in the amount of work to compute one breakpoint and the number of breakpoints [1], it was hoped that the number of breakpoints for minimal cost flow problems was also polynomial. This would have given a polynomial time algorithm for the (unparametrized) problem as follows:

(a) find any flow of the specified value v.

(b) find a set of costs c' for which this flow is optimal (say $c'_{ij} = 0$ when $f_{ij} \neq 0$.).

(c) solve the parametric problem with cost $c' + (c - c')\lambda$, where c is the given cost. At $\lambda = 1$, we have the optimal answer for the (unparametrized) original problem.

The result we give uses an example by Zadeh [10] of a parametric right-hand side, minimal cost flow problem on a network with $2n + 2$ nodes and $O(n^2)$ arcs; there are $O(2^n)$ breakpoints in $g(v)$, where $g(v)$ is the minimal cost possible when the value of the flow is v. The dual of a minimal cost flow problem is not necessarily a minimal cost flow problem so we cannot use this result directly for the parametric cost problem; we can, however, construct a parametric cost problem with one fewer breakpoint and with no more nodes in the underlying graph.

The details of this example are given in Section 1.

The second case considered here is for 0–1 programming. In both Murty's and Zadeh's examples, the optimal solution for at least some λ's has some exponentially large components; also unparametrized linear programming problems, such as Klee and Minty [3], which may take an exponential number of pivots would have exponentially large components for some feasible solutions if integrality were required at all basic feasible solutions. It seems reasonable, therefore, to ask whether the requirement that all components of all feasible solutions be one or zero limits the number of slope changes. If this conjecture were true, furthermore, we could draw immediate corollaries about the complexity of the parametric cost shortest chain, assignment, and travelling salesman problems. However, we will give an example of a 0–1 programming problem with n^2 variables and $2^n - 1$ breakpoints. This example is interesting in itself since it can be relaxed to a parametric linear programming problem, which in turn can be transformed into an unparametrized problem that may require an exponential number of pivots to solve; this 'bad' behavior is driven only by the cost function so the problem may serve as a valuable pathological example in testing proposed algorithms. Furthermore, a more profound examination of what drives the 'bad' behavior in this case may give insight into the complexity of other parametric 0–1 problems.

The details of this example and several simple variations on it are given in section 2.

1. Network programming

In this section we show that any parametric-flow cost-minimization problem (that is, for a given network G, find the optimal flow for each possible value v of the flow) can be transformed into a minimum cost problem (i.e. with fixed value and parametric costs) on a network with the same nodes as G and one more arc. For the sake of brevity, we will call the first simply the parametric value problem on G and the second the parametric cost problem.

For any convex function $h:\mathbb{R}\to\mathbb{R}\cup\infty$ the conjugate of h is defined to be:

$$h^*(y^*) = -\min_{y}\{h(y) - y^*y\}.$$

It can be shown [6] that h^* is well defined and convex and that $h^{**} = h$. Geometrically $h^*(y^*) = -b$, where $z = y^*y + b$ is a supporting (one dimensional) hyperplane of $\{(y, z)\colon z \geq h(y)\}$. From this it is easy to obtain:

Lemma 1. *If $\{y\colon h(y) < \infty\}$ is bounded and $h(y)$ is piece-wise linear, then $h(y)$ has one more breakpoint than $h^*(y^*)$. (Here we consider the endpoints of the interval where $h(y)$ is finite to be breakpoints.) In fact the breakpoints of h^* correspond to the straight line segments (flats) of h and vice versa.*

Recall that a cut set $(S;T)$ of a graph G is a partition of the nodes of G into sets S and T so that $s\in S$ and $t\in T$. Given a cut set $(S;T)$, we define (S,T) to be the set of arcs from S to T; that is $(i,j)\in(S,T)$ if i is in S, j is in T and (i,j) is in $\mathcal{A}$. We emphasize that $(S;T)$ and (S,T) are entirely different objects.

Given the parametric value problem on a graph G which has finite capacities k_{ij} and costs c_{ij}, we construct a parametric cost problem as follows:

(1) Add one arc (s,t) with zero cost and infinite capacity to G. This arc need not be the unique arc from s to t in the new graph, but the arc (s,t) always refers to this added one in what follows. We will call the new graph G'.

(2) Choose any cut set $(S;T)$ in the original graph G and set the parametric costs in $G\subset G'$ by:

$$c_{ij}(\lambda) = \begin{cases} c_{ij} - \lambda & (i,j)\in(S,T),\\ c_{ij} + \lambda & (i,j)\in(T,S),\\ c_{ij} & \text{otherwise.}\end{cases}$$

(3) Require a flow from s to t of value v', where v' is the maximum flow possible in G.

Lemma 2. *Let $g(v)$ be the minimum cost that a flow of (parametric) value v can attain on G. Let $f(\lambda)$ be the minimum cost of a flow of value v' on G', as a function of the cost parameter λ. Then $f = -g^*$, and therefore f has one fewer breakpoint than g.*

Proof. Obviously the cost of any flow $x = (x_{ij})$ is a linear combination of functions linear is λ so is itself linear in λ.

Since $c_{st} = 0$ and the intercepts of the costs of all other arcs are the same as the cost in the parametric value problem, the intercept of the cost of the flow (x_{ij}) is equal to the cost in the parametric value problem of $x|_G$, the restriction of x to G.

The value of the flow $x|_G$ must equal the value of the flow from S to T; that is,

$$v' - x_{st} = \sum_{(i,j)\in(S,T)} x_{ij} - \sum_{(i,j)\in(T,S)} x_{ij}. \tag{1}$$

The slope of the cost of an arc is (-1) for $(i,j)\in(S,T)$ and is $(+1)$ for $(i,j)\in(T,S)$; otherwise it is zero. Thus the right-hand side of equation (1) is the negative of the slope of the cost of (x_{ij}). Therefore the slope of the cost of a flow on G' is $x_{st} - v'$. Thus

$$\begin{aligned} f(\lambda) &= \min_x\{\text{cost at } \lambda \text{ of the flow } x\} \\ &= \min_{0\le v\le v'}\{\min\{\text{cost at } \lambda \text{ of the flow } x \\ &\qquad \text{with the value of } x|_G \text{ equal to } v\}\} \\ &= \min_{0\le v\le v'}\{\min\{\text{cost in the parameteric value problem of } x|_G \\ &\qquad \text{where the value of } x|_G \text{ is } v\} - v\lambda\} \\ &= \min_v\{g(v) - v\lambda\} \\ &= -g^*(\lambda). \end{aligned}$$

The final statement in Lemma 2 follows from Lemma 1.

Since the above lemma holds for all graphs G and all cuts $(S;T)$ we immediately obtain:

Theorem 1. *For any n, there is a graph G_n with $2n+2$ nodes, n^2+n+2 arcs, and an assignment of linear cost functions to the arcs so that there are $2^n + 2^{n-2} - 1$ breakpoints in the optimal cost curve.*

Proof. Zadeh's [10] network has the stated number of nodes and n^2+n+1 arcs; there are 2^n+2^{n-2} breakpoints (counting the two where the problem becomes infeasible) in the curve $g(v)$ in the parametric value problem. In the construction above, we add one arc and lose one breakpoint so the theorem is proven.

2. 0–1 Problems

We will first examine, as a special case of 0–1 parametric problems, the parametric-cost minimum-cut-set problem. In this section we will consider networks with some arcs and some (undirected) lines $(i;j)$. The undirected lines may be considered to be pairs of arcs (i,j) and (j,i) with the same costs and with opposing directions. The minimum cut set problem is to find a cut $(S;T)$ which minimizes

$$c(S;T)=\sum_{\substack{i\in S\\ j\in T\\ (i,j)\in\mathcal{A}}} c_{ij}+\sum_{\substack{i\in S\\ j\in T\\ (i;j)\in\mathcal{A}}} c_{ij}.$$

The family of examples, one for every integer n, given in this paper will have negative costs on some of the arcs. The duals of minimum cut set problems are maximum flow problems when the costs are positive; thus the unparametrized versions of minimum cut set problems with $c_{ij}\geq 0$ can be solved in polynomial time. On the other hand, the minimum cut set problem with $c_{ij}<0$ is equivalent to the maximum cut set problem which is known to be NP-complete.

Definition. Let $G_n=(\mathcal{N}_n,\mathcal{A}_n)$, where $\mathcal{N}_n$ consists of $n+2$ nodes $s,t,1,2,3,\dots,n$; the nodes $1,2,\dots,n$ will be called the numbered nodes. $\mathcal{A}_n$ consists of lines between all pairs of numbered nodes, an arc from the source s to each numbered node, and one arc from each numbered node to the sink t. In other words, G_n is composed of a complete graph on n nodes, together with a source and a sink connected to each of these nodes. The costs on the arcs are given by:

$$c_{ij}(\lambda)=\begin{cases}0 & j=t,\\ 2^{j-1}\lambda & i=s,\\ -2^{i+j-3} & i,j\neq s,t.\end{cases}$$

The cut set problem on this network has a physical interpretation [8, 9]. Each numbered node represents a job which can be performed on either of two machines, represented by s and t. For any pair of jobs—that is, any pair of numbered nodes i and j—$c_{ij}(\lambda)$ is the fixed communications cost (or interference savings) of doing the jobs on different machines. $c_{si}(\lambda)$ is the variable cost of doing the ith job on the first machine: $c_{it}(\lambda)$ is the cost of doing it on the second machine. Here λ may represent the level of the activity in the shop; we assume

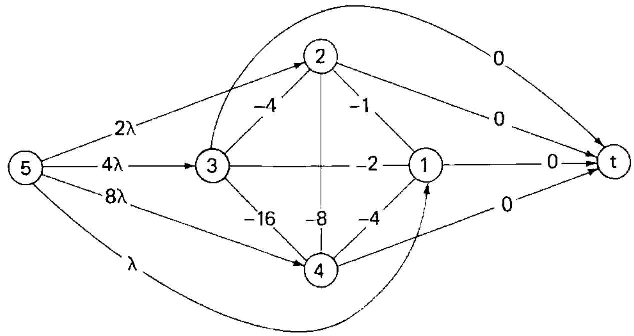

Fig. 1. The network G_4.

that the cost of using the machine is linear in the level of activity. If $(S; T)$ is an optimal cut at some value of λ, the numbered nodes in T should be done on the first machine (i.r., on s) while those in S should be done on the second machine (i.e., on t) to minimize the total cost of accomplishing the jobs.

For convenience, we set $S' = S \setminus \{s\}$ and $T' = T \setminus \{t\}$.

Lemma 3. *For any* $m = 0, 1, 2, \ldots, 2^n - 1$, *there is a cut* (S_m, T_m) *with cost*

$$c(S_m; T_m) = m\lambda - \tfrac{1}{2}m(2^n - m - 1).$$

Proof. First we give a simple formula for the cost of a cut.

$$\begin{aligned} c(S; T)(\lambda) &= \sum_{\substack{i \in S \\ j \in T \\ (i,j) \in \mathcal{A}}} c_{ij}(\lambda) \\ &= \sum_{i \in T'} c_{si} + \sum_{i \in S'} c_{it} + \sum_{\substack{i \in S' \\ j \in T'}} c_{ij} \\ &= \Big(\sum_{j \in T'} 2^{j-1}\Big)\lambda + 0 + \sum_{\substack{i \in S' \\ j \in T'}} (-2^{i+j-3}) \\ &= \Big(\sum_{j \in T'} 2^{j-1}\Big)\lambda - \frac{1}{2}\sum_{i \in S'} 2^{i-1}\Big(\sum_{j \in T'} 2^{j-1}\Big) \\ &= \Big(\sum_{j \in T'} 2^{j-1}\Big)\lambda - \frac{1}{2}\Big(\sum_{i \in S'} 2^{i-1}\Big)\Big(\sum_{j \in T'} 2^{j-1}\Big). \end{aligned} \tag{2}$$

Now let $m = \sum_{j=1}^{n} a(j)2^{j-1}$, where $a(j)$ is 0 or 1. Set $S'_m = \{j\colon a(j) = 0\}$ and $T'_m = \{j\colon a(j) = 1\}$. Then $m = \sum_{j \in T'_m} (2^{j-1})$ is the slope of $c(S_m; T_m)$. Furthermore

$$2^n - 1 = \sum_{i=1}^{n} 2^{i-1} = \sum_{i \in S'_m} 2^{i-1} + \sum_{j \in T'_m} 2^{j-1} = \sum_{i \in S'_m} 2^{i-1} + m$$

so

$$\sum_{i \in S'_m} 2^{i-1} = 2^n - m - 1.$$

Therefore, by equation (2),

$$c(S_m;T_m)(\lambda) = \sum_{j\in T_m}(2^{j-1})\lambda - \tfrac{1}{2}\Big(\sum_{j\in S_m}2^{j-1}\Big)\Big(\sum_{j\in T_m}2^{j-1}\Big)$$
$$= m\lambda - \tfrac{1}{2}(2^n - m - 1)m.$$

Theorem 2. *Let $f_n(\lambda)$ be the optimum cost attained in the minimum cut set problem on G_n at λ. Then $f_n(\lambda)$ has $2^n - 1$ breakpoints.*

Proof.

$$f_n(\lambda) = \min\{c(S_m;T_m): m = 0, 1, \dots, 2^n - 1\}.$$

For any $m = 0, 1, \dots, 2^n - 1$ we explicitly compute the λ for which $(S_m;T_m)$ is optimal; that is, for $m \neq m'$ we want to find all λ so that

$$c(S_m;T_m)(\lambda) < c(S_{m'};T_{m'})(\lambda).$$

By Lemma 3, we see this occurs when

$$m\lambda - \tfrac{1}{2}m(2^n - m - 1) < m'\lambda - \tfrac{1}{2}m'(2^n - m' - 1),$$
$$(m - m')\lambda < \tfrac{1}{2}((m - m')(2^n - 1) + (m' + m)(m' - m)).$$

If $m' < m$, then

$$\lambda < \tfrac{1}{2}(2^n - 1 - (m' + m)).$$

This inequality is strongest when $m' = m - 1$ so we have shown that $c(S_m;T_m)(\lambda) < c(S_{m'};T_{m'})(\lambda)$ for all $m' < m$ and $\lambda < \tfrac{1}{2}(2^{n-1} - m)$.

Similarly we can show that $c(S_m;T_m)(\lambda) < c(S_{m'};T_{m'})(\lambda)$ for $m' > m$ and all $\lambda > 2^{n-1} - m - 1$.

Thus $(S_m;T_m)$ is the optimal solution for $2^{n-1} - m - 1 < \lambda < 2^{n-1} - m$. Each of the 2^n cuts corresponds to a line segment of the curve and there are $2^n - 1$ breakpoints.

Notice that both the largest slope and the largest intercept of the $c_{ij}(\lambda)$ are exponential in the number of nodes. This will in general be necessary.

Property. *Suppose in the parametric* 0–1 *problem*

$$\min \sum (c_i^*\lambda + c_i)x_i,$$

$$Ax \geq b,$$
$$x_i = 0 \text{ or } 1, \quad i = 1, 2, \dots, n,$$

we know that c_i and c_i^* are integers for all i and that $|c_i^*| < C^*$ and $|c_i| < C$. Then the number of breakpoints can be no greater than $\min\{(2C^* + 1)n, (4C + 2)n\}$.

Proof. There are at most $(2C^*+1)n$ different slopes and at most $(2C+1)n$ different intercepts in the costs of feasible vectors x. The successive slopes of the segments on the optimal cost curve do not increase, so each slope may occur at most once. Since only the lines with the greatest slope and the least slope of all those having a given intercept can be optimal over some interval in λ, the number of breakpoints is at most twice the number of intercepts.

We now want to write the cut set problem as an explicitly 0–1 programming problem.

Lemma 4. *The cut set problem on G_n is equivalent to the following integer programming problem:*

$$\mathrm{P}_n: \quad \min \sum_{j=2}^{n}\sum_{i=1}^{j-1} c_{ij}(\lambda)u_{ij} + \sum_{i=1}^{n} c_{si}(\lambda)\pi_i + \sum_{i=1}^{n} c_{it}(\lambda)(1-\pi_i),$$

$$\pi_i - \pi_j \le u_{ij} \le \pi_i + \pi_j, \tag{3}$$

$$\pi_j - \pi_i \le u_{ij} \le 2-(\pi_i+\pi_j), \tag{4}$$

$$u_{ij}, \pi_i \in \{0,1\}, \quad i = 1,2,\ldots,n,$$
$$j = 1,2,\ldots,n, \quad i<j.$$

Here c_{ij} is the cost of the line $(i;j)$, c_{si} is the cost of (s,i) and c_{it} is the cost of (i,t).

We remark that this is not the standard formulation of the cut set problem, which only uses the lower inequalities in (3) and (4). We require the upper bounds because of the negative costs.

Also some variables which appear in standard formulations are unnecessary here; for example u_{si} is not used as $u_{si}=\pi_i$.

Proof of Lemma 4. We give a cost-preserving one-to-one correspondence between the set of feasible solutions to P_n and the set of cuts of G_n. Let $S(\pi,u)=\{i: \pi_i=0\}\cup\{s\}$; $T(\pi,u)=\{i: \pi_i=1\}\cup\{t\}$. Then $(S(\pi,u);T(\pi,u))$ is a cut of G_n. The function $(\pi,u)\mapsto(S(\pi,u);T(\pi,u))$ gives a one-to-one correspondence since the values of the π_i can be chosen arbitrarily and since we can show by a trivial calculation that u is determined by

$$u_{ij}=\begin{cases}0 & \pi_i=\pi_j,\\ 1 & \pi_i\neq\pi_j.\end{cases}$$

The cost of the arc(s,i) contributes to $c(S(\pi,u);\ T(\pi,u))$ if and only if $i\in T(\pi,u)$—that is, $\pi_i=1$ and the cost $c_{si}(\lambda)$ contributes to the objective function in P_n. Similarly we can show the correspondence of the cost $c_{it}(\lambda)$ in the two problems.

Finally suppose the line $(i;j)$ has one end in $S(\pi,u)$ and one in $T(\pi,u)$; here i

and j are numbered nodes with $i<j$. Then $\pi_i \neq \pi_j$ so $u_{ij}=1$. Thus the cost of (π, u) is equal to the $(S(\pi, u); T(\pi, u))$.

From Lemma 4, we can easily derive:

Theorem 3. *For any n, there exists a parametric cost 0–1 programming problem with $\frac{1}{2}n(n-1)+n=\frac{1}{2}n(n+1)$ variables and 2^n-1 breakpoints in the optimal cost function.*

We can transform P_n into a number of other 0–1 problems, all having the same (up to addition of a constant) optimal cost curve; thus we can extend the result of Theorem 3 to several other problems.

Corollary 1. *For every n, there exists a parametric cost 0–1 problem with $\frac{1}{2}(9n^2-7n)$ variables, equality constraints, and 2^n-1 breakpoints in the optimal cost curve.*

Proof. By the addition of slack variables, P_n can be transformed into a problem with equality constraints. Furthermore, each slack variable y is an integer between 0 and 2, inclusive. Therefore we may substitute two 0–1 variables y_1 and y_2 for y. Since there are four inequalities for each of the $\frac{1}{2}n(n-1)$ variables u_{ij}, we add $4(n^2-n)$ variables. The cost curve of course remains the same as we have only changed the description of the feasible set.

Corollary 2. *For every n, there exists a parametric cost knapsack problem with $\frac{1}{2}(9n^2-7n)$ variables, an equality constraint, and 2^n-1 breakpoints in the optimal cost curve.*

Proof. There are standard techniques—called aggregation procedures—for combining equality constraints in bounded integer programming problems [4]. In fact, Rosenberg [7] shows that if $|A_{i\cdot}x-b_i|<M$ for all $x\in I^n$ and each equation $A_{i\cdot}x=b_i$, then

$$\{x\in I^n\colon Ax=b,\ x \text{ integer}\}=\left\{x\in I^n:\sum_{i=1}^{m} M^{i-1}A_{i\cdot})x=\sum_{i=1}^{m} M^{i-1}b_i\right\}.$$

Here $A_{i\cdot}$ is the ith row of A. In our example, we may take $M=4$ so no coefficient in the resulting knapsack problem is greater than 4^m, $m=\frac{1}{2}(9n^2-7n)$.

One may wonder if the knapsack problem with inequality constraints, positive weights, and positive costs in the interval between the first and the last breakpoints can also have an exponential number of breakpoints. (The weights in a knapsack problem are the coefficients of the x_i in the single constraint.) After all, the inequality problem has a guaranteed ϵ-accurate polynomial heuristic while the equality problem has such a heuristic only if $\mathrm{P}=\mathrm{NP}$. Unfortunately, we can show that the inequality-constrained problem is no better than the equality constrained one.

For simplicity's sake, we will write the set of knapsack problems described in Corollary 2 as:

$$\begin{aligned} \max \quad & v(\lambda)x, \\ \text{st} \quad & wx = W_0, \\ & x_i = 0 \text{ or } 1, \quad i = 1, 2, \ldots, n. \end{aligned}$$

Notice the objective function is now maximized; this follows the usual practice in the literature on the knapsack problem.

Without loss of generality we may assume that $w \geq 0$ since if $w_i < 0$, we may replace x_i with its complementary variable $(1 - x_i) = \bar{x}_i$ and change W_0 appropriately.

Claim. In fact, for every i, $w_i \neq 0$.

Proof. If $w_i = 0$, the feasibility of a vertex in I^n is independent of its ith coordinant; that is, if x is a solution with a zero in its ith coordinate and x' is the solution obtained from x by changing the zero in the ith coordinate to a one, then x is feasible if and only if x' is. Re-examining the problem given in Corollary 2, we see that no variable such as x_i exists. This completes the proof of the claim.

We define two constants:

$$C = \max_{\substack{w \cdot x < W_0 \\ x_i = 0 \text{ or } 1}} \{w \cdot x\}$$

$$L = \max_{\lambda \in [-2^n, 2^n]} \left\{ \max_{w \cdot x \leq W_0} \{v(\lambda) \cdot x\} - \min_{w \cdot x = W_0} \{v(\lambda) \cdot x\} \right\}.$$

Let $M = L/(W_0 - C)$ and $v'(\lambda) = v(\lambda) + Mw$.

Claim. For the cost $v'(\lambda)$ and for $\lambda \in (-2^n, 2^n)$, the optimal solution is always in the hyperspace $wx = W_0$ for the inequality-constrained problem as well as the equality constrained one.

Proof. Choose $x^{(1)}$ and $x^{(2)}$ feasible to $\{x\colon wx \leq W_0,\ x_i = 0 \text{ or } 1 \text{ for } i = 1, \ldots, n\}$ and so that $wx^{(1)} = W_0$ and $wx^{(2)} < W_0$. Then

$$\begin{aligned} v'(\lambda)(x^{(2)} - x^{(1)}) &= (v(\lambda) + Mw)(x^{(2)} - x^{(1)}) \\ &= v(\lambda)(x^{(2)} - x^{(1)}) + Mw(x^{(2)} - x^{(1)}) \\ &\leq L + M(C - W_0) \\ &\leq L - \frac{L(C - W_0)}{(C - W_0)} = 0. \end{aligned}$$

This proves the claim.

Notice that Mwx is a constant on the hyperspace $wx = W_0$ so the optimal cost curve with cost $v'(\lambda)$ is just the optimal cost curve for $v(\lambda)$ translated upward by some amount MW_0. In fact, since no component of w is zero, we may choose the multiplier M of w large enough so that, in addition, all the components of $v'(\lambda)$ are positive for λ in the interval $(-2^n, 2^n)$ and so that there are still 2^{n-1} breakpoints in the optimal cost curve. Thus we have:

Corollary 3. *For every n, there exists a parametric cost knapsack problem with an inequality constraint,* $\frac{1}{2}(9n^2 - 7n)$ *variables, positive weights,* $2^n - 1$ *breakpoints in the optimal cost curve in the interval* $(-2^n, 2^n)$, *and with positive cost components* $v_i'(\lambda)$ *for* $\lambda \in (-2^n, 2^n)$.

We remark that a guaranteed ϵ-accurate heuristic—for example, that of Ibarra and Kim [2]—for the unparametrized problem can be used at specific λ to find an approximately optimal solution there. Putting together several of these solutions would give an approximation to the optimal cost curve $f(\lambda)$. This solution need not be concave since the slopes of the costs of the approximations need not decrease with λ. Furthermore, we cannot guarantee ϵ-accuracy in polynomial time since ϵ-accuracy implies that the cost curve of the approximate solutions lies between $f(\lambda)$ and $(1-\epsilon)f(\lambda)$ and we can choose ϵ small enough that this requirement forces approximate solution to change at least once for every slope change in $f(\lambda)$. See Fig. 2. Therefore the heuristic would have to be applied at least 2^n times for the problem in Corollary 3.

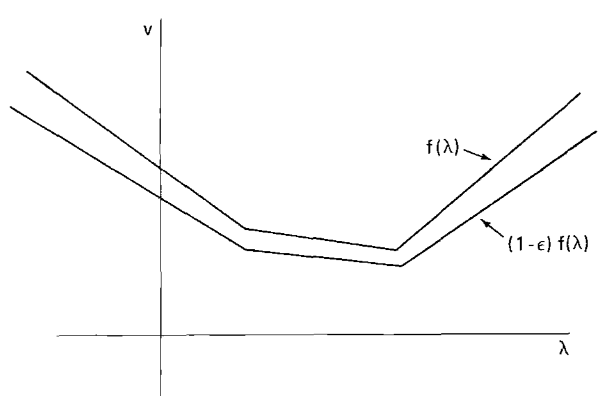

Fig. 2. Here $f(\lambda) = \max\{v(\lambda)\cdot x\text{: feasible } x\}$. An ϵ-accurate heuristic would give $g(\lambda)$, where $(1-\epsilon)f(\lambda) \le g(\lambda) \le f(\lambda)$ and $g(\lambda) = v(\lambda)\cdot x$ for some feasible x. If ϵ is small enough, $g(\lambda)$ will contain at least as many straight line segments as $f(\lambda)$.

Conclusions

Some interesting questions remain.

(1) Find a heuristic that approximates the optimal cost curve in the knapsack problem with an inequality constraint 'fairly well' and 'for most λ's,' where the phrases in quotation marks are deliberately left vague since the definition of the problem is really much of the question.

(2) The problems in Section 2 have $2^n - 1$ breakpoints, and $O(n^2)$ variables so the number of breakpoints is $O(2^{\sqrt{m}})$, where m is the number of variables. Are there families of 0–1 problems with m variables and at least a^m breakpoints, for some $a > 1$?

(3) Is there a polynomial bound on the number of breakpoints in the optimal cost curve of a parametric cost shortest chain problem, or of a parametric cost assignment problem?

(4) The property following Theorem 2 gives some simple conditions that guarantee the number of breakpoints is polynomial in the number of variables. What other (more interesting) conditions can be given?

Acknowledgment

The author would like to thank two anonymous referees and Professor Katta Murty for helpful suggestions for this paper.

References

[1] D. Gusfield, "Sensitivity analysis for combinatorial optimization", Memorandum number UCB/ERL M80/22, Electronics Research Laboratory (Berkeley, CA, 1980).

[2] O.H. Ibarra and C.E. Kim, "Fast approximation algorithms for the knapsack and sum of subset problem", *Journal of the Association for Computing Machinery* 22 (1975) 463–468.

[3] V. Klee and G.J. Minty, "How good is the simplex algorithm?" in O. Shisha, Ed., *Inequalities III* (Academic Press, New York, 1969) pp. 159–175.

[4] G. Mathews, "On the partition of numbers", *Proceedings of the London Mathematical Society* 28 (1897) 486–490.

[5] K. Murty, "Computational complexity of parametric linear programming", *Mathematical Programming* 19 (1980) 213–219.

[6] R. Rockafellar, *Convex analysis* (Princeton University Press, Princeton, NJ, 1970).

[7] I.G. Rosenberg, "Aggregation of equations in integer programming", *Discrete Mathematics* 10 (1974) 325–341.

[8] H. Stone, "Multiprocessor scheduling with aid ne.work flow algorithms", *IEEE Transactions on Software Engineering* SE-3 (1977) 85–93.

[9] H. Stone, "Critical load factors", *IEEE Transactions on Software Engineering* SE-4 (1978) 254–258.

[10] N. Zadeh, "A bad network problem for the simplex method and other minimum cost flow algorithms", *Mathematical Programming* 5 (1973) 255–266.