International Institute for Applied Systems Analysis
IIASA www.iiasa.ac.at

An Efficient Algorithm for Bicriteria Minimum-Cost Circulation Problem

Katoh, N.

IIASA Working Paper

WP-87-098

July 1987

WORKING PAPER

AN EFFICIENT ALGORITHM FOR BICRITERIA
MINIMUM-COST CIRCULATION PROBLEM

Naoki Katoh

July 1987
WP-87-98

International Institute
for Applied Systems Analysis
A-2361 Laxenburg/Austria

IIASA

An Efficient Algorithm for Bicriteria Minimum-cost Circulation Problem

Naoki Katoh

July 1987
WP-87-98

INTERNATIONAL INSTITUTE FOR APPLIED SYSTEMS ANALYSIS
A-2361 Laxenburg, Austria

Foreword

This paper is concerned with a bicriteria minimum-cost circulation problem which arises in interactive multicriteria decision making. The author presents a strongly polynomial algorithm for this problem, which runs in $O(\min\{n^6\log^3 n, n^4(n\log n + m)\log^5 n\})$ time, where n and m are the numbers of vertices and edges in a graph respectively. It is achieved by making use of the parametric characterization of optimal solutions and a strongly polynomial algorithm for the single objective minimum-cost circulation problem.

Alexander B. Kurzhanski
Chairman
System and Decision Sciences Program

An Efficient Algorithm for Bicriteria Minimum-cost Circulation Problem

Naoki Katoh

1. Introduction

In recent years, many types of interactive optimization methods have been developed and used in practical situations in order to support multicriteria decision makings (see the book by Sawaragi, Nakayama and Tanino [19] and Wierzbicki and Lewandowski [27] for the survey of this topic). Given an *admissible decision set* (or a *feasible decision set*) $X \subseteq R^n$, and p objective functions, $f_1, f_2, \cdots, f_p$ (all are assumed to be minimization for convenience), the following problem formulations have been used in various situations of interactive multicriteria decision makings:

$$\underset{x \in X}{\text{minimize}} \ \max_{1 \le i \le p} \{\alpha_i f_i(x) + \beta_i\} \quad , \tag{1}$$

where α_i and β_i are positive and real constants respectively, which are computed based on the information supplied by the decision maker and/or the decision support system.

α_i and β_i are typically determined in the following manner by the reference point method, which is one of the well known methods used in interactive multicriteria decision support systems (see [27] for the survey of reference point methods). This method requires the decision maker to specify the aspiration level q_i and the reservation level r_i for each objective f_i. The values of q_i and r_i are respectively interpreted as the desirable outcome for i-th objective that the decision maker would like to attain, and the maximum allowable outcome for i-th objective. Then the degree of the achievement of a given $x \in X$ for an i-th objective is measured by

$$\mu_i(q_i, r_i, f_i(x)) = (r_i - f_i(x))/(r_i - q_i) \quad . \tag{2}$$

The aggregated degree of the achievement for x is then measured by

$$s = \min_{1 \le i \le p} \mu_i(q_i, r_i, f_i(x)) \quad . \tag{3}$$

The method solves the following problem:

$$\underset{x \in X}{\text{maximize}} \quad s \quad , \tag{4}$$

and provides its optimal solution x^* to the decision maker. If x^* is not satisfactory for the decision maker, he or she may respecify the aspiration and/or reservation levels and the above process is repeated until a satisfactory solution is obtained. At each round of this iteration, we need to solve the problem (4). Letting $\alpha_i = 1/(r_i - q_i)$ and $\beta_i = -r_i/(r_i - q_i)$, we have

$$\alpha_i f_i(x) + \beta_i = -\mu_i(q_i, r_i, f_i(x)) \quad .$$

Therefore, the problem (4) is equivalent to problem (1).

Some other modifications and generalizations of this achievement function have been proposed by several authors, i.e., Wierzbicki [22, 23, 24, 25, 26], Nakayama [16], Steuer and Choo [20], (see also [27] for general discussion about achievement functions). Many of those achievement functions have the form similar to the one in (3).

In view of this, it is of great significance to study the computational complexity required for solving the problem (1).

We concentrate on the case where $p = 2$ and each single objective problem $P_i, i = 1,2$ defined below

$$P_i : \underset{x \in X}{\text{minimize}} \; f_i(x) \tag{5}$$

is a minimum cost circulation problem (SMCP). Both problems are assumed to have optimal solutions. We shall study problem (1) with such restrictions, which we call a ***bicriteria minimum-cost circulation problem*** (BMCP). Given a directed graph $G = (V, E)$, where V and E denote the sets of vertices and edges respectively, a single objective minimum-cost circulation problem (SMCP) can be written as follows.

$$SMCP : \text{minimize} \sum_{e \in E} c(e)x(e) \tag{6}$$

subject to

$$\{\Sigma x(e) | e = (u,v) \in E\} = \{\Sigma x(e') | e' = (v,w) \in E\} \text{ for } v \in V \tag{7}$$

$$a(e) \leq x(e) \leq b(e) \text{ for } e \in E \quad . \tag{8}$$

Here $a(e), b(e)$ and $c(e)$ are given integer numbers. $a(e) = -\infty$ and $b(e) = +\infty$ are allowed. Let the objective functions f_1 and f_2 for Problem BMCP be

$$f_1(x) = \sum_{e \in E} c_1(e)x(e) \text{ and } f_2(x) = \sum_{e \in E} c_2(e)x(e) \quad , \tag{9}$$

and define

$$g_i(x) = \alpha_i f_i(x) + \beta_i \quad , \quad i = 1, 2 \quad , \tag{10}$$

where $c_1(e)$ and $c_2(e)$ are integers and $\alpha_1, \alpha_2 > 0$. Problem BMCP is then describesub as follows.

$$\textit{BMCP}: \text{minimize } \max\{g_1(x), g_2(x)\} \tag{11}$$

subject to the constraints of (7) and (8).

Recently Tardos [21] discovered a strongly polynomial algorithm for solving Problem SMCP, the existence of which was an open problem since Edmonds and Karp [5] proposed a polynomial time algorithm for it. An algorithm that solves a problem whose input consists of n real numbers is *strongly polynomial* if

(a) it performs only elementary arithmetic operations (additions, subtractions, comparisons, multiplications and divisions),

(b) the number of operations required to solve the problem is polynomially bounded in n, and

(c) when applied to rational data, the size of the numbers (i.e., the number of bits required to represent the numbers) that the algorithm generates is polynomially bounded in n and the size of the input numbers.

Based on Tardos' result, Fujishige [7], Orlin [17], Galil and Tardos [9] proposed more efficient strongly polynomial algorithms. Among them, the one given by Galil and Tardos [9] is the fastest, which runs in $O(n^2(m + n \log n)\log n)$ time, where $n = |V|$ and $m = |E|$.

The major goal of this paper is to propose a strongly polynomial algorithm for solving Problem BMCP, which runs in $O(min\{n^6 \log^3 n, n^4(n \log n + m)\log^5 n\})$ time. Notice that that Problem BMCP can be equivalently transformed to the fol-

lowing form.

BMCP': minimize z

subject to (7), (8) and

$$g_i(x) \leq z \quad , \quad i = 1,2 \tag{12}$$

Such reformulation has been used in the more general setting in order to solve problem (1) (see Chapter 7 of the book [19]). This approach may not be recommended in case the set X has a good structure, since the new constraints (12) added to the original feasible decision set X may destroy the good structure of X. In our problem, we cannot guarantee any more the total unimodularity of the constraint matrix associated with the constraints (7), (8) and (12) for the above problem BMCP', while the constraint matrix associated with the constraints (7) and (8) is known to be totally unimodular (see the books by Lawler [13] and Papadimitriou and Steiglitz [18]), which enables us to develop efficient algorithms for Problem SMCP.

The algorithm proposed here, on the other hand, does not use the above formulation, but takes full advantage of the good structure of the constraints (7) and (8). It employs as a subroutine the strongly polynomial algorithm for solving Problem SMCP by Galil and Tardos [9], and finds an optimal solution of Problem BMCP in $O(\min\{n^6 \log^3 n, n^4(m + n \log n)\log^5 n\})$ time. The techniques we use are related to Megiddo [14, 15]. The problems treated in [14, 15] are, however, different from ours. Our result implies that the basic ideas developed by [14, 15] can be utilized to solve a class of problems which have the objective function such as the one in (1) with $p = 2$.

Our problem is also related but not equivalent to the minimum cost circulation problem with one additional linear constraint, which was studied by Brucker [2]. The algorithm proposed by [2] is, however, not strongly polynomial. The techniques developed here can be directly used to improve the running time of Brucker's algorithm to have a strongly polynomial algorithm whose running time is the same as the one for BMCP. We also show that the techniques developed here can be extended to the case where the objective function is not the one as in (11) but is such as $(\Sigma \alpha_i |f_i(x) - \beta_i|^p)^{\frac{1}{p}}$, where p is a positive integer.

This paper is organized as follows: Section 2 gives some basic results. Section 3 presents an outline of the algorithm for solving Problem BMCP. Section 4 gives the detailed description of the algorithm which runs in $O(n^4(n \log n + m)^2 \log^2 n)$ time. Section 5 improves the running time of the algorithm explained in Section 4 to $O(\min\{n^6 \log^3 n, n^4(m + n \log n) \log^5 n\})$, based on the idea given by Megiddo [15], which employs the idea of parallel combinatorial algorithms to speed up the running time for many types of combinatorial optimization problems not including our type of problem, though in fact we do not need any parallel processor but simulate the parallel algorithm in a serial manner. Section 6 discusses some extensions of our approach to other types of problems such as the minimum cost circulation problem with one additional linear constraint.

2. Basic Concepts and Properties

Let $X \subseteq R^E$ denote the set of $|E|$-dimensional vectors x satisfying (7) and (8), i.e., X is the feasible decision set, and let $f(x) = (f_1(x), f_2(x)) : R^E \to R^2$ denote the function that maps $x \in X$ to the *objective plane* R^2. Define

$$Y = \{(f_1(x), f_2(x)) | x \in X\} \quad , \tag{13}$$

which is called the *feasible set* (or *admissible outcome* set). Notice that set Y is a convex polygon since X is a convex polyhedron and both $f_1(x)$ and $f_2(x)$ are linear. A vector $y = (y_1, y_2) \in Y$ is called *efficient* if there does not exist $y' = (y_1', y_2') \in Y$ such that $y_i' \leq y_i$ holds for $i = 1,2$ and at least one inequality holds strictly. A set of all efficient vectors is called the *efficient set*, which we denote Y_0. A vector $y = (y_1, y_2) \in Y$ is called *weakly efficient* if there does not exist $y' = (y_1', y_2') \in Y$ such that $y_i' < y_i$ holds for each $i = 1,2$. An $x \in X$ such that $f(x)$ is efficient is called an *efficient solution*. The sets Y and Y_0 are illustrated in Figure 1 as the shaded area and the thick piecewise linear curve, respectively.

The following auxiliary problem with nonnegative parameter λ plays a central role in our algorithm.

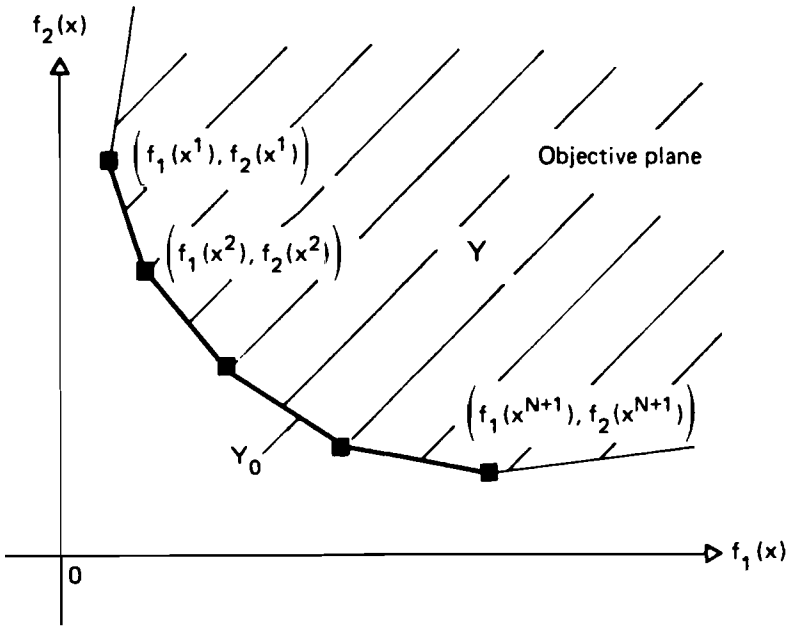

Figure 1. Illustration of the sets Y and Y_0.

$$P(\lambda): v(\lambda) \equiv \text{minimize } f_1(x) + \lambda f_2(x) \qquad (14)$$

subject to (7) and (8).

It is well known (see [10] for example) that the function $v(\lambda)$ is piecewise linear and concave in λ, as illustrated in Figure 2, with a finite number of joint points $\lambda_{(1)}, \lambda_{(2)}, \cdots, \lambda_{(N)}$ with $\lambda_{(1)} < \lambda_{(2)} < \ldots < \lambda_{(N)}$. Here N denotes the number of total joint points, and let $\lambda_{(0)} = 0$ and $\lambda_{(N+1)} = \infty$ for convenience. Define for each $\lambda \in [0, \infty)$

$$X^*(\lambda) \equiv \{x \in X | x \text{ is optimal to } P(\lambda)\} \quad . \qquad (15)$$

The following lemma is well known in the theory of linear parametric programs (see Gal [8] for the survey of this topic). In what follows, for two real numbers a, b with $a \leq b$, (a, b) and $[a, b]$ stand for the open interval $\{t | a < t < b\}$ and the closed interval $\{t | a \leq t \leq b\}$ respectively.

Lemma 1.

(i) For any $\lambda \in (\lambda_{(k-1)}, \lambda_{(k)})$, $k = 1, \ldots, N+1$, we have

$$X^*(\lambda) \subset X^*(\lambda_{(k-1)}) \text{ and } x^*(\lambda) \subset X^*(\lambda_{(k)}) \quad . \qquad (16)$$

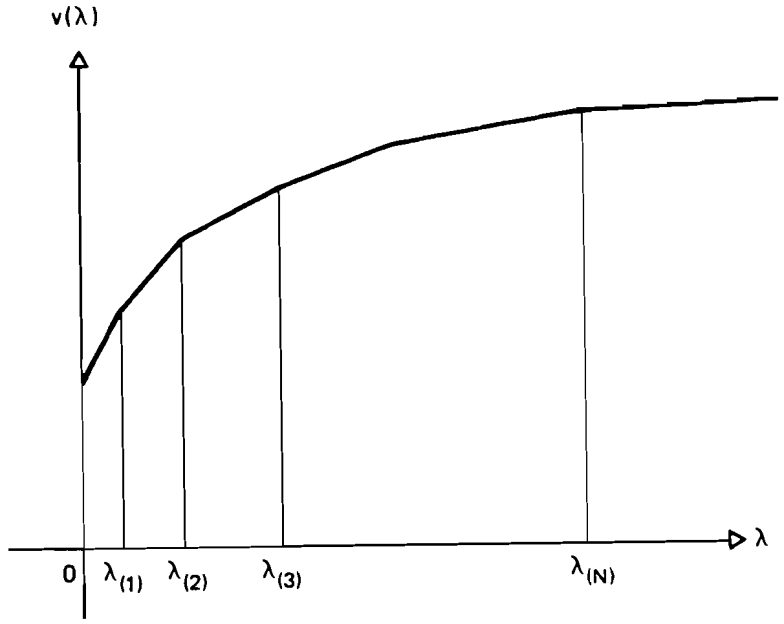

Figure 2. Illustration of $v(\lambda)$.

(ii) For any two distinct $\lambda, \lambda' \in (\lambda_{(k-1)}, \lambda_{(k)})$, $k = 1,\ldots,N+1$, we have

$$X^*(\lambda) = X^*(\lambda') \quad . \tag{17}$$

(iii) For any $\lambda \in (\lambda_{(k-1)}, \lambda_{(k)})$ and any $\lambda' \in (\lambda_{(k)}, \lambda_{(k+1)})$, $k = 1,\ldots,N$,

$$X^*(\lambda_{(k)}) = \{\mu x + (1-\mu)x' \mid 0 \le \mu \le 1, x \in X^*(\lambda) \text{ and } x' \in X^*(\lambda')\} \quad .$$

Let for $k = 1,\ldots,N+1$

$$X_k^* = \{x \in X \mid x \in X^*(\lambda) \text{ for all } \lambda \in [\lambda_{(k-1)}, \lambda_{(k)}]\} \quad . \tag{18}$$

By Lemma 1, $X_k^* = X^*(\lambda)$ holds for all $\lambda \in (\lambda_{(k-1)}, \lambda_{(k)})$. The following lemma is known in the theory of parametric linear programming.

Lemma 2.

(i) For any two $x, x' \in X_k^*$ with $1 \le k \le N+1$,

$$f_1(x) = f_1(x') \text{ and } f_2(x) = f_2(x')$$

hold.

(ii) For any $x \in X^*_{k-1}$ and any $x' \in X^*_k$ with $2 \le k \le N+1$,

$$f_1(x) < f_1(x') \text{ and } f_2(x) > f_2(x')$$

hold.

Lemma 3.

(i) For any $\lambda \ge 0$ and any $x \in X^*(\lambda)$, $f(x)$ is weakly efficient.

(ii) For any $\lambda > 0$ and any $x \in X^*(\lambda)$, $f(x)$ is efficient.

(iii) For any $x \in X^*_k$, $k = 1,2,\ldots,N+1$, $f(x)$ is a vertex of set Y.

Proof. The proof of (i) is given by Dinkelbach [4] and Bowman [1]. (ii) is proved as follows. If there exists $x' \in X$ such that $f_i(x') \le f_i(x)$, $i = 1,2$, hold and one of inequalities is strict. In any case, by $\lambda > 0$, it implies

$$f_1(x') + \lambda f_2(x') < f_1(x) + \lambda f_2(x) \quad ,$$

contradicting that x is optimal to $P(\lambda)$. (iii) is proved as follows. Since $f(x)$ is efficient, $f(x)$ is on the boundary of set Y. If $f(x)$ is not a vertex, it can be represented by a convex combination of two vertices. That is, there exist $x', x'' \in X$ and μ with $0 < \mu < 1$ such that

$$x = \mu x' + (1-\mu)x'' \quad . \tag{19}$$

Since x is optimal to $P(\lambda)$ with $\lambda \in (\lambda_{(k-1)}, \lambda_{(k)})$, it follows that

$$f_1(x') + \lambda f_2(x') = f_1(x'') + \lambda f_2(x'') = f_1(x) + \lambda f_2(x) \quad ,$$

since otherwise $f_1(x') + \lambda f_2(x') < f_1(x) + \lambda f_2(x)$ or $f_1(x'') + \lambda f_2(x'') < f_1(x) + \lambda f_2(x)$ holds by (19), contradicting the optimality of x to $P(\lambda)$. Therefore, both x' and x'' are optimal to $P(\lambda)$ and by Lemma 2 (i) $f_1(x) = f_1(x') = f_1(x'')$ and $f_2(x) = f_2(x') = f_2(x'')$ follow. This contradicts that $f(x)$, $f(x')$ and $f(x'')$ are distinct points in the objective plane. □

By Lemma 3 (iii) $f(x) = (f_1(x), f_2(x))$ maps all $x \in X^*_k$ to a unique efficient vertex in Y, and it is easy to see that such mapping from X^*_k, $k = 1,\ldots,N+1$ to the set of efficient vertices is one to one. Therefore we use the notation x^k to represent any $x \in X^*_k$ in what follows. As k increases from 1 to $N+1$, the corresponding efficient vertex moves from top-left to bottom-right in the objective plane (see Figure 1). The edge connecting two consecutive efficient vertices corresponding to X^*_k and X^*_{k+1} respectively corresponds to all optimal solutions of

$P(\lambda_{(k)})$. The following lemma gives a basis for our algorithm.

Lemma 4.

(i) If $g_1(x^1) > g_2(x^1)$, then x^1 is optimal to Problem BMCP.

(ii) If $g_1(x^{N+1}) < g_2(x^{N+1})$, then x^{N+1} is optimal to Problem BMCP.

(iii) If neither (i) nor (ii) holds, there exists k^* with $1 \leq k^* \leq N$ such that

$$g_1(x^{k^*}) \leq g_2(x^{k^*}) \text{ and } g_1(x^{k^*+1}) \geq g_2(x^{k^*+1}) \tag{20}$$

hold. Letting μ be the solution of the following linear equation

$$\mu g_1(x^{k^*}) + (1-\mu) g_1(x^{k^*}) = \mu g_2(x^{k^*+1}) + (1-\mu) g_2(x^{k^*+1}) \quad , \tag{21}$$

then

$$x^* = \mu x^{k^*} + (1-\mu) x^{k^*+1} \tag{22}$$

is an optimal solution of BMCP.

Proof. (i) If x^1 is not optimal to BMCP, there exists $\hat{x} \in X$ such that $f_1(\hat{x}) < f_1(x^1)$ and $f_2(\hat{x}) < f_1(x^1)$ hold. $f_1(\hat{x}) < f_1(x^1)$ implies that x^1 is not optimal to $P(0)$, but x^1 is optimal to $P(0)$ by Lemma 1 (iii). This is a contradiction. (ii) is proved in a manner similar to (i). (iii) First note that by Lemma 2 (ii) and by definition of $g_t(x)$, there exists k^* such that x^{k^*} and x^{k^*+1} satisfy (20). In addition, by Lemma 2 (ii), the linear equation of (21) in μ has a unique solution satisfying $0 \leq \mu \leq 1$. x^* defined by (22) is then optimal to $X^*(\lambda_{(k^*)})$ by Lemma 1 (iii), and $f(x^*)$ is efficient by Lemma 3 (ii). It follows from (21) and (22) that

$$g_1(x^*) = g_2(x^*)$$

holds. Since $f(x^*)$ is efficient, there is no $x \in X$ such that $f_1(x) < f_1(x^*)$ and $f_2(x) < f_2(x^*)$ hold. Thus there is no $x \in X$ such that $g_1(x) < g_1(x^*)$ and $g_2(x) < g_2(x^*)$ by (10) and $\alpha_1, \alpha_2 > 0$, implying that there is no $x \in X$ such that $\max\{g_1(x), g_2(x)\} < \max\{g_1(x^*), g_2(x^*)\}$ holds. □

To illustrate the situations corresponding to Lemma 4(i), (ii) and (iii), it is useful to consider the set

$$Z = \{(g_1(x), g_2(x)) | x \in X\} \quad .$$

the set Z is obtained from set Y by an affine transformation and is similar to Y in shape. Set Z is illustrated in Fig. 3 as the shaded area. The thick piecewise linear

curve in Fig. 3 corresponds to the efficient set Y_0. Figures 3 (a), (b) and (c) respectively illustrate the set Z in which Lemma 4 (i), (ii) and (iii) hold. The straight line passing through the origin in Figs. 3(a) ~ (c) separates the set Z into two subsets; one in which $g_1(x) \leq g_2(x)$ holds and the other in which $g_1(x) \geq g_2(x)$ holds. If Lemma 4(i) holds, all $(g_1(x^k), g_2(x^k))$, $k = 1,\ldots,N+1$, lie below the straight line (see Fig. 3(a)) among which $(g_1(x^1), g_2(x^1))$ is nearest to the line and hence x^1 is optimal. The case of Lemma 4(ii) is similarly illustrated in Fig. 3(b). If Lemma 4(iii) holds, the problem is reduced to find k^* such that $(g_1(x^{k^*}), g_2(x^{k^*}))$ is above the straight line and $(g_1(x^{k^*+1}), g_2(x^{k^*+1}))$ is below it. An optimal solution x^* is the one such that $(g_1(x^*), g_2(x^*))$ is the intersection point of the edge connecting $(g_1(x^{k^*}), g_2(x^{k^*}))$ and $(g_1(x^{k^*+1}), g_2(x^{k^*+1}))$ and the straight line (i.e., $g_1(x^*) = g_2(x^*)$).

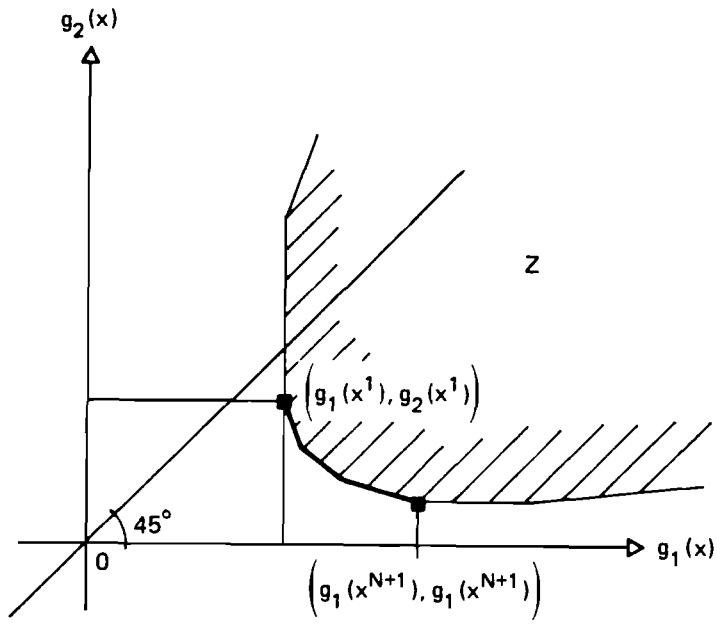

Figure 3 (a). The case in which Lemma 4(i) holds

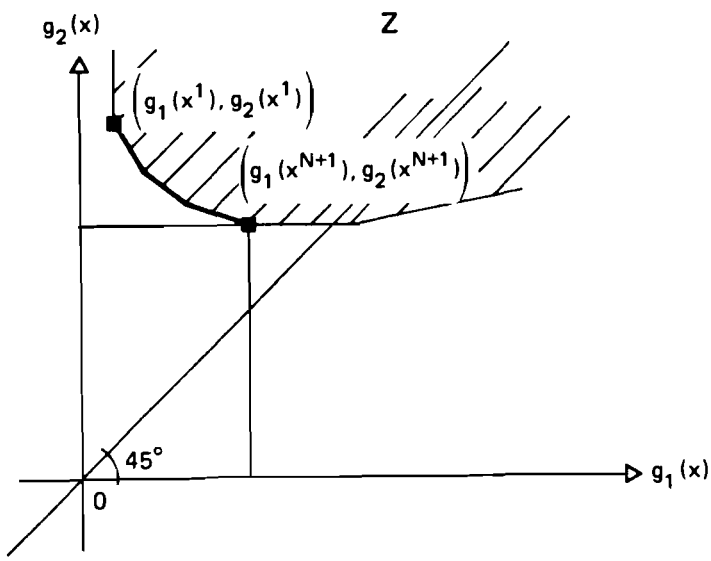

Figure 3 (b). The case in which Lemma 4 (ii) holds.

By Lemma 2 (i), the condition of Lemma 4 (i) (resp. (ii)) is tested simply by taking any λ with $\lambda < \lambda_{(1)}$ (resp. $\lambda > \lambda_{(N)}$) and obtaining an optimal solution x of $P(\lambda)$. If neither the condition of Lemma 4 (i) nor (ii) holds, we need to find k^* satisfying (20). For this, we only have to know $\lambda_{(k^*)}$. Once $\lambda_{(k^*)}$ is obtained, x^{k^*} and x^{k^*+1} are obtained by solving $P(\lambda_{(k^*)} - \varepsilon)$ and $P(\lambda_{(k^*)} + \varepsilon)$ respectively, where ε is a sufficiently small positive number satisfying $\lambda_{(k^*)} - \lambda_{(k^*-1)} < \varepsilon$ and $\lambda_{(k^*+1)} - \lambda_{(k^*)} < \varepsilon$. The following lemma is useful for finding λ with $\lambda < \lambda_{(1)}$ or $\lambda > \lambda_{(N)}$ and for estimating the above ε.

Lemma 5. Let

$$\alpha_1 = \max\{\max\{|a(e)| \mid e \in E,\ a(e) \text{ is finite}\},\ \max\{|b(e)| \mid e \in E,\ b(e) \text{ is finite}\}\} , \tag{23}$$

$$\alpha_2 = \max\{|c_i(e)| \mid i = 1,2,\ e \in E\} , \tag{24}$$

and

$$M = (n + 2m)m\alpha_1\alpha_2 . \tag{25}$$

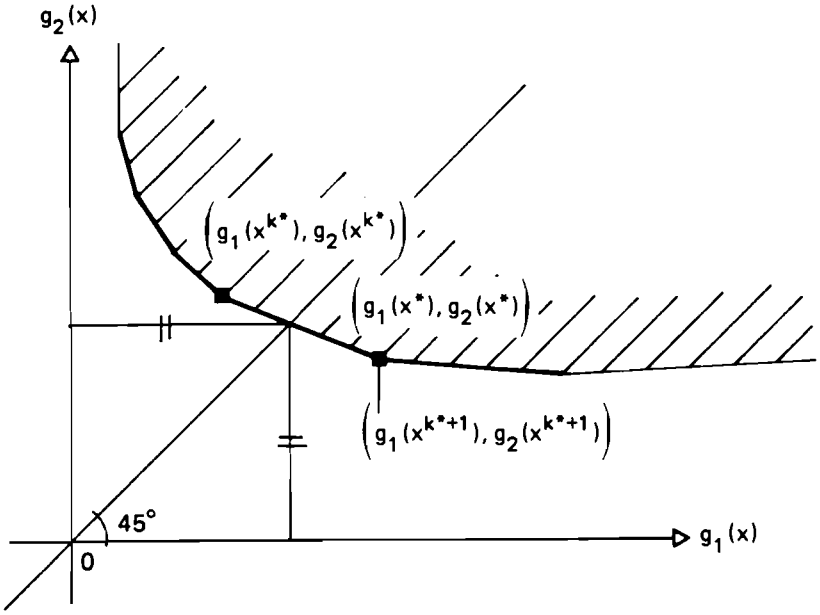

Figure 3 (c). The case in which Lemma 4 (iii) holds

Figure 3. Illustration of set Z.

Then

$$\lambda_{(1)} \geq M^{-1}, \quad \lambda_{(N)} \leq M \tag{26}$$

and

$$\lambda_{(k+1)} - \lambda_{(k)} \geq 1/M^2, k = 1,\ldots,N-1 \tag{27}$$

hold.

Proof. (26) is proved by showing that for any $\lambda \in [0, \infty)$ and any $x \in X^*(\lambda)$

$$\max\{|f_j(x)| \mid x \in X, j = 1,2\} \leq M \tag{28}$$

holds. After proving this, (26) follows by Lemma 9.2 in the paper by Katoh and Ibaraki [11]. Note that $P(\lambda)$ for a fixed λ is a linear program and the constraints (7) and (8) of $P(\lambda)$ can be written in the form $A\bar{x} = b$ by introducing $2m$ slack variables for $2m$ inequalities of (8), where A is $(n + 2m) \times (3m)$ matrix, $\bar{x}$ is a $3m$-dimensional vector and b is a $(n + 2m)$-dimensional column vector each of whose element is either $0, a(e)$ or $b(e)$. It is well known in the theory of linear program

that there exists an optimal solution x of $P(\lambda)$ such that x is a restriction of $\tilde{x}$ to nonslack variables where $\tilde{x}$ is a basic feasible solution of $A\tilde{x} = b$. $\tilde{x}$ is written by

$$\tilde{x} = B^{-1}b = \frac{B^{adj}b}{\det(B)} \quad , \tag{29}$$

where B is a $(n + 2m) \times (n + 2m)$ nonsingular square submatrix of A, B^{-1} is the inverse matrix of B, B^{adj} is the adjoint of B and $\det(B)$ is the determinant of B. It is well known (see Chapter 4 of the book by Lawler [13] or Chapter 13 of the book by Papadimitriou and Steiglitz [18]) that matrix A is *totally unimodular*, i.e., every square submatrix C of A has the determinant of either 0, +1 or −1. Hence $\det(B)$ is equal to either +1 or −1. Each element of B^{adj}, is, by definition, equal to an determinant of $(n + 2m - 1) \times (n + 2m - 1)$ submatrix of A, which is also equal to either 0, +1, or −1 by the total unimodularity of A. Since each element of A is also equal to either 0, +1 or −1, $\tilde{x}$ is an integer vector and the absolute value of each element of $\tilde{x}$ is at most $(n + 2m)a_1$. Therefore

$$|\sum_{e \in E} c_i(e)x(e)| \leq (n + 2m)ma_1a_2, \quad i = 1,2$$

follows. This proves (26), since by Lemma 2(i) the value $f_i(x')$, $i = 1,2$ is the same for all $x' \in x^*(\lambda)$ if λ is not a joint point, and by Lemma 1 (iii) it is represented by the convex combination of $f_i(x^k)$ and $f_i(x^{k+1})$ if λ is a joint point $\lambda_{(k)}$.

Now we shall prove (27). For k which $1 \leq k \leq N-1$, consider x^k, x^{k+1} and x^{k+2}, all of which can be assumed to be integer vectors as proved above. Since $x, x' \in X^*(\lambda_{(k)})$ and $x', x'' \in X^*(\lambda_{(k+1)})$ hold by Lemma 1 (i), it holds that

$$f_1(x^k) + \lambda_{(k)}f_2(x^k) = f_1(x^{k+1}) + \lambda_{(k)}f_2(x^{k+1})$$

and

$$f_1(x^{k+1}) + \lambda_{(k+1)}f_2(x^{k+1}) = f_1(x^{k+2}) + \lambda_{(k+1)}f_2(x^{k+2}) \quad .$$

Thus, we have

$$\lambda_{(k)} = \frac{f_1(x^{k+1}) - f_1(x^k)}{f_2(x^k) - f_2(x^{k+1})} \text{ and } \lambda_{(k+1)} = \frac{f_1(x^{k+2}) - f_1(x^{k+1})}{f_2(x^{k+1}) - f_2(x^{k+2})} \quad .$$

Then

$$\lambda_{(k+1)} - \lambda_{(k)} = \frac{(f_1(x^{k+1}) - f_1(x^k))(f_2(x^{k+1}) - f_2(x^{k+2})) - (f_1(x^{k+2}) - f_1(x^{k+1}))(f_2(x^k) - f_2(x^{k+1}))}{(f_2(x^k) - f_2(x^{k+1}))(f_2(x^{k+1}) - f_2(x^{k+2}))}$$

Since $f_1(x)$ and $f_2(x)$ take integer values if x is an integer vector, and $f_1(x^k) < f_1(x^{k+1}) < f_1(x^{k+2})$ and $f_2(x^k) > f_2(x^{k+1}) > f_2(x^{k+2})$ hold by Lemma 2 (ii), the numerator is not less than 1. Since

$$-M \leq f_2(x^{k+2}) < f_2(x^{k+1}) < f_2(x^k) \leq M$$

holds by (26),

$$(f_2(x^k) - f_2(x^{k+1}))(f_2(x^{k+1}) - f_2(x^{k+2})) \leq M^2$$

follows. This proves $\lambda_{(k+1)} - \lambda_{(k)} \leq 1/M^2$. □

By Lemma 5, it is easy to test whether the condition of Lemma 4 (i) or (ii) holds. For this purpose, we have only to solve $P(\lambda)$ for some λ with $0 < \lambda < 1/M$ and for some $\lambda > M$. If the condition of Lemma 4 (iii) holds, we must find k^* satisfying (20), x^{k^*}, x^{k^*+1} and μ of (21) to compute x^* by (22). One possible approach to do this is to employ the binary search for determining $\lambda \in (\lambda_{(k^*-1)}, \lambda_{(k^*)})$ and $\lambda' \in (\lambda_{(k^*)}, \lambda_{(k^*+1)})$ over the interval $[\underline{\lambda}, \overline{\lambda}]$ where $\underline{\lambda}$ and $\overline{\lambda}$ are appropriate numbers satisfying $\underline{\lambda} < 1/M$ and $\overline{\lambda} > M$ respectively. By Lemma 5, such binary search may be terminated until the interval length is reduced to less than $1/M^2$ (though the details are omitted). Therefore such method requires $O(n^2(m + n \log n) \log n \cdot \log M)$ time. This is polynomial in the input size because $\log M$ is polynomial in the input size by (25). However, it is not strongly polynomial because of the term $\log M$. The following section alternatively presents a strongly polynomial algorithm for finding $\lambda_{(k^*)}$ with k^* satisfying (20). Once it is obtained, x^{k^*} and x^{k^*+1} are computed by solving $P(\lambda_{(k^*)} - \varepsilon)$ and $P(\lambda_{(k^*)} + \varepsilon)$ respectively. Here ε satisfies $0 < \varepsilon < 1/M^2$. This is justified since (27) implies $\lambda_{(k^*)} - \varepsilon \in (\lambda_{(k^*-1)}, \lambda_{(k^*)})$ and $\lambda_{(k^*)} + \varepsilon \in (\lambda_{(k^*)}, \lambda_{(k^*+1)})$.

3. The Outline of the Algorithm

As discussed at the end of the previous section, it is easy to test whether the condition of Lemma 4(i) or (ii) holds. Thus we assume in this section that the condition of Lemma 4(iii) holds and we shall focus on how to compute $\lambda_{(k^*)}$ with k^* satisfying (20).

The idea of the algorithm is similar to the one given by Megiddo [14] which was developed for solving fractional programs. The similar idea was also used by Gusfield [10] to determine the curve of the objective cost for parametric combinatorial problems. We apply their ideas to find $\lambda_{(k^*)}$. The algorithm applies the algo-

rithm of Galil and Tardos (the *GT-algorithm*) to solve $P(\lambda_{(k^*)})$ without knowing the exact value of $\lambda_{(k^*)}$. The computation path of the GT-algorithm may contain conditional jump operations, each of which selects proper computation path depending upon the outcome of comparing two numbers. Notice that the GT-algorithm contains arithmetic operations of only additions, subtractions, multiplications and divisions, and comparisons of the numbers generated from the given problem data, and that when applying the GT-algorithm to solve $P(\lambda)$ with λ treated as unknown parameter, the numbers generated in the algorithm are all linear functions of λ or constants not containing λ. Note that comparisons are necessary at conditional jumps. If a comparison for a conditional jump operation is made between two linear functions of $\lambda_{(k^*)}$, the condition can be written in the form of

$$\lambda_{(k^*)} > \hat{\lambda}, \lambda_{(k^*)} = \hat{\lambda} \text{ or } \lambda_{(k^*)} < \hat{\lambda} \tag{30}$$

for an appropriate critical constant $\hat{\lambda}$, which can be determined by solving the linear equation in $\lambda_{(k^*)}$ constructed from the compared two linear functions. Here $\hat{\lambda}$ is assumed to be positive since otherwise $\hat{\lambda} < \lambda_{(k^*)}$ is clearly concluded.

An important observation here is that condition (30) can be tested without knowing the value of $\lambda_{(k^*)}$. For this, solve $P(\hat{\lambda} - \varepsilon), P(\hat{\lambda})$ and $P(\hat{\lambda} + \varepsilon)$ by the GT-algorithm, where $\hat{\lambda}$ is now a known constant, and ε is a positive constant satisfying $\varepsilon < 1/2M^2$. Let x, x', x'' be the obtained solution of $P(\hat{\lambda} - \varepsilon), P(\hat{\lambda})$ and $P(\hat{\lambda} + \varepsilon)$ respectively. First we test whether $\hat{\lambda}$ is a joint point or not, based on the following lemma.

Lemma 6. Let x, x' and x'' be those defined above. Then $\hat{\lambda}$ is a joint point if and only if the following linear equation in λ has the unique solution λ' equal to $\hat{\lambda}$.

$$f_1(x) + \lambda f_2(x) = f_1(x'') + \lambda f_2(x'') \quad . \tag{31}$$

Proof. If $\hat{\lambda}$ is a joint point, say $\lambda_{(k)}$, $\hat{\lambda} - \varepsilon$ and $\hat{\lambda} + \varepsilon$ lie in the intervals $(\lambda_{(k-1)}, \lambda_{(k)})$ and $(\lambda_{(k)}, \lambda_{(k+1)})$ respectively, by Lemma 5. Thus, from definition of a joint point, $f_1(x) + \lambda f_2(x)$ (resp. $f_1(x'') + \lambda f_2(x'')$) defines the value $v(\lambda)$ of (14) for $\lambda \in [\lambda_{(k-1)}, \lambda_{(k)}]$ (resp. $[\lambda_{(k)}, \lambda_{(k+1)}]$). Thus (31) has a unique solution $\lambda = \lambda_{(k)}$.

If $\hat{\lambda}$ is not a joint point, let $\hat{\lambda}$ belong to $(\lambda_{(k-1)}, \lambda_{(k)})$ for some k with $2 \le k \le N + 1$. The following five cases are possible.

Case 1. $\lambda_{(k-1)} < \hat{\lambda} - \varepsilon$ and $\hat{\lambda} + \varepsilon < \lambda_{(k)}$. In this case $f_1(x) = f_1(x'')$ and

$f_2(x) = f_2(x'')$ hold by Lemma 2, and (31) has no unique solution.

Case 2. $\hat{\lambda} - \varepsilon < \lambda_{(k-1)}$ and $\hat{\lambda} + \varepsilon < \lambda_{(k)}$. By $\varepsilon < 1/2M^2$ and Lemma 5, $\hat{\lambda} - \varepsilon > \lambda_{(k-2)}$ holds and the equation (31) has the unique solution $\lambda' = \lambda_{(k-1)}$ which is not equal to $\hat{\lambda}$.

Case 3. $\hat{\lambda} - \varepsilon > \lambda_{(k-1)}$ and $\hat{\lambda} + \varepsilon > \lambda_{(k)}$. This case is treated in a manner similar to Case 2.

Case 4. $\hat{\lambda} - \varepsilon = \lambda_{(k-1)}$ and $\hat{\lambda} + \varepsilon < \lambda_{(k)}$. x^1 satisfies

$$f_2(x) \leq f_2(x'')$$

since $v(\lambda)$ of (14) is concave. If $f_2(x) < f_2(x'')$, the equation of (31) has the unique solution $\lambda' = \lambda_{(k-1)} \neq \hat{\lambda}$. If $f_2(x) = f_2(x'')$, (31) has no unique solution.

Case 5. $\hat{\lambda} - \varepsilon > \lambda_{(k-1)}$ and $\hat{\lambda} + \varepsilon = \lambda_{(k)}$. This case is analogous to Case 4.

Note that the case of $\hat{\lambda} - \varepsilon \leq \lambda_{(k-1)}$ and $\hat{\lambda} + \varepsilon \geq \lambda_{(k)}$ is not possible because of $\lambda_{(k)} - \lambda_{(k-1)} \geq 1/M^2$ by Lemma 5 and $\varepsilon < 1/2M^2$ by assumption. □

After computing x, x' and x'' defined above, the algorithm proceeds as follows. If one of x, x' and x'' (say, $\hat{x}$) satisfies

$$(g_1(\hat{x}) \equiv)\alpha_1 f_1(\hat{x}) + \beta_1 = \alpha_2 f_2(\hat{x}) + \beta_2 (\equiv g_2(\hat{x})) \quad , \tag{32}$$

$\hat{x}$ is an optimal solution of BMCP since $f(\hat{x})$ is weakly efficient by Lemma 3 (i) and hence there is no $x \in X$ such that $f_1(x) < f_1(\hat{x})$ and $f_2(x) < f_2(\hat{x})$ hold. So, assume in what follows that none of x, x', x'' satisfies (32). Depending upon whether $\hat{\lambda}$ is a joint point or not, consider the following two cases.

Case 1. $\hat{\lambda}$ is not a joint point. We then compare the two values $g_1(x')$ and $g_2(x')$. Two subcases are possible.

Subcase 1A. $g_1(x') < g_2(x')$. Then $\lambda^* > \hat{\lambda}$ is concluded, and the algorithm chooses the computation path corresponding to $\lambda^* > \hat{\lambda}$.

Subcase 1B. $g_1(x') > g_2(x')$. Then $\lambda^* < \hat{\lambda}$ is concluded, and the algorithm chooses the computation path corresponding to $\lambda^* < \hat{\lambda}$.

Case 2. $\hat{\lambda}$ is a joint point. Then we consider the following three subcases.

Subcase 2A. $g_1(x'') < g_2(x'')$. By Lemma 2 (ii), $g_1(x) < g_2(x)$ follows. This implies $\lambda^* > \hat{\lambda} + \varepsilon$ and the algorithm chooses the computation path corresponding to

$\lambda^* > \hat{\lambda}$.

Subcase 2B. $g_1(x) > g_2(x)$. Similarly to Subcase 2A, $g_1(x'') > g_2(x'')$ follows. This implies $\lambda^* < \hat{\lambda} - \varepsilon$ and the algorithm chooses the computation path corresponding to $\lambda^* < \hat{\lambda}$.

Subcase 2C. $g_1(x) < g_2(x)$ and $g_1(x'') > g_2(x'')$.
Then $\hat{\lambda}$ is the desired joint point $\lambda_{(k^*)}$ by Lemma 4 (iii). By Lemma 5 and $0 < \varepsilon < 1/M^2$, $x \in X^*_{k^*}$ and $x'' \in X^*_{k^*+1}$ follow. Therefore, by Lemma 4 (iii), an optimal solution x^* of BMCP is found by (21) and (22) after letting $x^{k^*} = x$ and $x^{k^*+1} = x''$.

With this observation the algorithm starts with the initial interval $(\underline{\lambda}, \overline{\lambda})$, where $\underline{\lambda}$ and $\overline{\lambda}$ are typically determined by $\underline{\lambda} = 1/(M+1)$, $\overline{\lambda} = M + 1$, and every time it performs the conditional jump operation, the critical value $\hat{\lambda}$ is computed, and $P(\hat{\lambda} - \varepsilon)$, $P(\hat{\lambda})$ and $P(\hat{\lambda} + \varepsilon)$ are solved. Depending upon the cases explained above, the length of the interval may be reduced in such a way that the desired joint point $\lambda_{(k^*)}$ exists in the reduced interval. It will be shown in the next section that Subcase 2C always occurs during the course of the algorithm, which proves the correctness of our algorithm. Since the GT-algorithm requires $O(n^2(n \log n + m)\log n)$ jump operations, and at each jump operation at most three minimum cost circulation problems, i.e., $P(\hat{\lambda} - \varepsilon), P(\hat{\lambda})$ and $P(\hat{\lambda} + \varepsilon)$, are solved by calling the GT-algorithm, the entire algorithm requires $O(n^4(n \log n + m)^2 \log^2 n)$ time in total.

4. Description and Analysis of the Algorithm

The algorithm for solving Problem BMCP is described as follows:

Procedure SOLVEBMCP

Input: A directed graph $G = (V,E)$ with costs $c_1(e), c_2(e)$, lower and upper compacities $a(e)$ and $b(e)$ for each $e \in E$, and the weights α_1, α_2, β_1 and β_2.

Output: An optimal solution of Problem BMCP.

Step 0: [Initialization]. Compute M by (23), (24), (25). Let $\underline{\lambda} = 1/(M+1)$, $\overline{\lambda} = M + 1$ and $\varepsilon = 1/(2M^2 + 1)$.

Step 1: [Test the conditions of Lemma 4 (i) and (ii)]. Compute optimal solutions x' and x'' of Problems $P(\underline{\lambda})$ and $P(\overline{\lambda})$ respectively by applying the GT-algorithm. If

x' (resp. x'') satisfies $g_1(x') > g_2(x')$ (resp. $g_1(x'') < g_2(x'')$), output x' (resp. x'') as an optimal solution of BMCP and halt. Otherwise go to Step 2.

Step 2: [Test the condition of Lemma 4 (iii)].

(i) Follow the GT-algorithm applied to $P(\lambda_{(k^*)})$ treating $\lambda_{(k^*)}$ as unknown constant satisfying $\underline{\lambda} < \lambda_{(k^*)} < \bar{\lambda}$. If the GT-algorithm halts, go to Step 3. Else at the next conditional jump operation, do the following.

(ii) Let the condition of the jump operation given by

$$p_1(\lambda_{(k^*)}) < p_2(\lambda_{(k^*)}), p_1(\lambda_{(k^*)}) = p_2(\lambda_{(k^*)}) \text{ or } p_1(\lambda_{(k^*)}) > p_2(\lambda_{(k^*)}) \quad , \tag{33}$$

where $p_1(\lambda_{(k^*)})$ and $p_2(\lambda_{(k^*)})$ are linear functions in $\lambda_{(k^*)}$. Solve equation

$$p_1(\lambda_{(k^*)}) = p_2(\lambda_{(k^*)}) \quad . \tag{34}$$

(iii) If equation (34) has no solution λ^* satisfying $\underline{\lambda} < \lambda^* < \bar{\lambda}$, i.e., $p_1(\lambda^*) < p_2(\lambda^*)$ (or $p_1(\lambda^*) > p_2(\lambda^*)$) holds for all such λ^*, then choose the corresponding computation path at the current conditional jump operation. Go to (viii).

(iv) If equation (34) holds for all λ^* with $\underline{\lambda} < \lambda^* < \bar{\lambda}$, choose $p_1(\lambda^*) = p_2(\lambda^*)$ as the proper computation path, and go to (viii).

(v) If equation (34) has the unique solution $\hat{\lambda}$ such that $\underline{\lambda} < \hat{\lambda} < \bar{\lambda}$, the conditions of (33) are transformed to

$$\lambda_{(k^*)} < \hat{\lambda} \text{ (resp. } \lambda_{(k^*)} > \hat{\lambda}), \lambda_{(k^*)} = \hat{\lambda} \text{ or } \lambda_{(k^*)} > \hat{\lambda} \text{ (resp. } \lambda_{(k^*)} < \hat{\lambda}) \quad .$$

Solve $P(\hat{\lambda} - \varepsilon), P(\hat{\lambda})$ and $P(\hat{\lambda} + \varepsilon)$ by applying the GT-algorithm, and let x, x' and x'' be optimal solutions of these problems respectively.

(vi) If one of x, x', x'' satisfies (32), output it as an optimal solution of BMCP and halt.

(vii) Test whether $\hat{\lambda}$ is a joint point or not based on Lemma 6.

(vii-a) If $\hat{\lambda}$ is not a joint point, determine $\lambda_{(k^*)} > \hat{\lambda}$ or $\lambda_{(k^*)} < \hat{\lambda}$ according to Subcases 1A and 1B given prior to the description of the algorithm, and choose the proper computation path corresponding to $\lambda_{(k^*)} > \hat{\lambda}$ or $\lambda_{(k^*)} < \hat{\lambda}$ respectively. If $\lambda_{(k^*)} > \hat{\lambda}$, let $\underline{\lambda} = \hat{\lambda}$. Otherwise let $\bar{\lambda} = \hat{\lambda}$. Go to (viii).

(vii-b) If $\hat{\lambda}$ is a joint point, determine $\lambda_{(k^*)} > \hat{\lambda} + \varepsilon$, $\lambda_{(k^*)} < \hat{\lambda} - \varepsilon$ or $\lambda_{(k^*)} = \hat{\lambda}$ according to Subcases 2A, 2B and 2C given prior to the description of the algorithm, respectively. If $\lambda_{(k^*)} > \hat{\lambda} + \varepsilon$ (resp. $\lambda_{(k^*)} < \hat{\lambda} - \varepsilon$), let $\underline{\lambda} = \hat{\lambda} + \varepsilon$ (resp.

$\bar{\lambda} = \hat{\lambda} - \varepsilon$), choose the proper computation path according to $\lambda_{(k^*)} > \hat{\lambda}$ (resp. $\lambda_{(k^*)} < \hat{\lambda}$) and go to (viii). If $\lambda_{(k^*)} = \hat{\lambda}$, compute μ satisfying (21) and then x^* by (22) after letting $x^{k^*} = x$ and $x^{k^*+1} = x''$. Halt.

(viii) Return to the conditional jump operation of the GT-algorithm in Step 2, from where it exited to find the proper computation path.

Step 3: Halt. □

The correctness of the algorithm is almost clear by the discussion given in the previous section. What remains is to prove that the algorithm always halts either in Step 1, Step 2 (vi) or Step 2 (vii). Assume otherwise. Note that Algorithm SOLVEBMCP always halts because it follows the GT-algorithm. Assume that SOLVEBMCP halts in Step 3 and consider the interval $(\underline{\lambda}, \bar{\lambda})$ generated when it halts in Step 3. It follows from the discussion given in Section 3 that $\underline{\lambda} < \lambda_{(k^*)} < \bar{\lambda}$ holds. When Algorithm SOLVEBMCP halts, it has obtained a solution which is optimal to $P(\lambda)$ for all $\lambda \in (\underline{\lambda}, \bar{\lambda})$, since if the GT-algorithm is applied to solve $P(\lambda)$ for any $\lambda \in (\underline{\lambda}, \bar{\lambda})$, it follows the same computation path irrespective of choice of λ from the interval $(\underline{\lambda}, \bar{\lambda})$. However, by Lemma 2(iii), an optimal solution of $P(\lambda')$ with $\underline{\lambda} < \lambda' < \lambda_{(k^*)}$ is not optimal to $P(\lambda'')$ with $\lambda_{(k^*)} < \lambda'' < \bar{\lambda}$. This is a contradiction.

The running time of the algorithm can be derived in a manner similar to [14]. At each jump operation, a linear equation (34) is solved and if it has the unique solution $\hat{\lambda}$ with $\underline{\lambda} < \hat{\lambda} < \bar{\lambda}$, three problems $P(\hat{\lambda} - \varepsilon), P(\hat{\lambda}), P(\hat{\lambda} + \varepsilon)$ are solved. So Step 2 (v) requires $O(n^2(n \log n + m)\log n)$ time. The other part of Step 2 is dominated by this. Since the total number of jump operations in the GT-algorithm is $O(n^2(n \log n + m)\log n)$, Step 2 is repeated $O(n^2(n \log n + m)\log n)$ times. So, Step 2 requires $O(n^4(n \log n + m)^2 \log^2 n)$ time in total. Since Step 0 requires constant time and Step 1 requires $O(n^2(n \log n + m)\log n)$ time, Algorithm SOLVEBMCP requires $O(n^4(n \log n + m)^2 \log^2 n)$ time in total.

Theorem 1. Algorithm SOLVEBMCP correctly computes an optimal solution of Problem BMCP in $O(n^4(n \log n + m)^2 \log^2 n)$ time.

The algorithm is in fact strongly polynomial, since the running time depends only on the numbers of vertices and edges in a graph, and if the input data are all rational numbers, the size of the numbers generated in the algorithm is clearly polynomial in n, m and the size of the input numbers.

This running time is improved to $O(\min\{n^6\log^3 n, n^4(n\log n + m)\log^5 n\})$ in the following section by utilizing the idea of simulating the parallel shortest path algorithm in a serial manner. Such idea of simulating parallel algorithms for the purpose of the speed-up of algorithms was originated by Megiddo [15]. The application of his idea to our problem, however, seems to be new.

5. Time Reduction

In order to reduce the running time of Algorithm SOLVEBMCP, the following remarks are useful.

Remark 1. In the GT-algorithm, the shortest path algorithm is applied $O(n^2\log n)$ times as a subroutine. Since the best known shortest path algorithm with a single source node, which is due to Fredman and Tarjan [6], requires $O(n\log n + m)$ time, the GT-algorithm requires $O(n^2(n\log n + m)\log n)$ time in total.

Remark 2. When the GT-algorithm is applied to solve $P(\lambda)$, comparisons with two numbers containing λ are made only when the shortest path algorithm is applied.

We modify Algorithm SOLVEBMCP in such a way that instead of using $O(n\log n + m)$ shortest path algorithm, we employ a parallel shortest path algorithm such as Dekel, Nassimi and Sahni's [3] and Kučera's[12] in a serial manner when SOLVEBMCP follows the GT-algorithm in Step 2(i). We still use $O(n\log n + m)$ shortest path algorithm in other parts of SOLVEBMCP such as Step 1, Step 2 (v). The idea of the time reduction is based on Megiddo [15]. We shall explain how it is attained. Let P denote the number of processors and let τ_P denote the number of steps required on a P-processor machine. Dekel, Nassimi and Sahni's scheme requires $P = O(n^3)$ and $\tau_P = O(\log^2 n)$ while Kučera's scheme requires $P = O(n^4)$ and $\tau_P = O(\log n)$. We simulate these algorithms serially. According to some fixed permutation, we visit one processor at a time and perform one step in each cycle. At each processor, when two linear functions $p_1(\lambda), p_2(\lambda)$ are compared, we execute Step 2 (ii), (iii) and (iv). If the equation $p_1(\lambda) = p_2(\lambda)$ has a unique solution $\hat{\lambda}$ with $\underline{\lambda} < \hat{\lambda} < \bar{\lambda}$, such critical value $\hat{\lambda}$ is stored and we proceed to the next processor without executing Step 2 (v), (vi) and (vii). After one step of the multiprocessor, we have at most P such critical values. Let $\hat{\lambda}_1, \hat{\lambda}_2, \cdots, \hat{\lambda}_P$ denote such critical values. We then compute

$$\lambda' = \max\{\hat{\lambda}_i \mid 1 \le i \le P,\ \hat{\lambda}_i < \lambda_{(k')}\} \quad ,$$

$$\lambda'' = \max\{\hat{\lambda}_i \mid 1 \le i \le P,\ \hat{\lambda}_i > \lambda_{(k^*)}\} \quad ,$$

or in the meantime we may find the desired joint point $\lambda_{(k^*)}$ among those critical values. As explained in [15], this is done by performing a binary search that requires $O(P)$ time for median findings in subsets of the set of critical values, and $O(\log P)$ applications of the GT-algorithm. We explain in more details how λ' is computed (the case λ'' is similarly treated). Each time the median $\hat{\lambda}_i$ is found from among the remaining critical values, we execute Step 2 (v) , (vi) and (vii) with $\hat{\lambda}$ replaced by $\hat{\lambda}_i$. In Step 2 (vii), it may happen that $\hat{\lambda}_i$ is concluded as the desired joint point $\lambda_{(k^*)}$. Otherwise $\hat{\lambda}_i < \lambda_{(k^*)}$ or $\hat{\lambda}_i > \lambda_{(k^*)}$ is concluded, and half of the remaining critical points are discarded. Since the remaining subset during binary search is halved each time, the time required to find all medians is

$$O(P + P/2 + P/4 + \cdots) = O(P) \quad ,$$

as shown in [15]. Since we need $O(\log P)$ applications of the median finding in order to find λ', it requires

$$O(n^2(n \log n + m) \log n \cdot \log P)$$

time. Hence, each step of the multiprocessor requires

$$O(P + n^2(n \log n + m) \log n \cdot \log P)$$

time. After λ' and λ'' are computed, we can choose the proper computation path at each processor. Since the above process is repeated τ_P times in total, each application of the parallel shortest path algorithm requires

$$O((P + n^2(n \log n + m) \log n \log P)\, \tau_P)$$

time. Since the shortest path problem is solved $O(n^2 \log n)$ times as mentioned in Remark 1, the total running time is

$$O(n^2 \log n\, (P + n^2(n \log + m) \log n \log P) \tau_P) \quad .$$

If Dekel, Nassimi and Sahni's scheme is employed, this becomes

$$O(n^4(n \log n + m) \log^5 n) \quad ,$$

while if Kučera's scheme is employed, it becomes

$$O(n^6 \log^3 n) \quad .$$

Therefore, depending on how dense the graph is, we may choose the better one. We then have the following theorem.

Theorem 2. The modified SOLVEBMCP solves Problem BMCP in $O(\min\{n^4(n \log n + m)\log^5 n, n^6 \log^3 n\})$ time.

6. Extensions

In this section, we shall show how our approach is generalized to other types of problems which are variants of Problem BMCP studied so far. One of such problems is the minimum-cost circulation problem with one additional linear constraint studied by Brucker [2]. This is described as follows.

$$\begin{aligned} &\text{SMCPLC: minimize } f_1(x) \\ &\qquad \text{subject to the constraints of (7) and (8) and} \\ &\qquad\qquad f_2(x) \le d \quad . \end{aligned} \tag{35}$$

Here, f_1 and f_2 are those defined in (9), and d is a given constant. The above problem is solved as follows. It is easy to see that there exists an optimal solution x^* of SMCPLC such that $f(x^*)$ is efficient. Define $x^k, k = 1,\ldots,N+1$, by

$$x^k \in X_k^* \quad , \tag{36}$$

as before. If $f_2(x^1) \le d, x^1$ is optimal to SMCPLC. If $f_2(x^{N+1}) > d$, there is no feasible solution to SMCPLC. So assume

$$f_2(x^{N+1}) \le d < f_2(x^1) \quad . \tag{37}$$

Let

$$\lambda_{(k_1)} = \min\{\lambda_{(k)} | 1 \le k \le N, f_2(x^{k+1}) \le d\} \quad , \tag{38}$$

$$\lambda_{(k_2)} = \max\{\lambda_{(k)} | 1 \le k \le N, f_2(x^{k+1}) > d\} \quad . \tag{39}$$

Lemma 7. Let μ satisfy

$$\mu f_2(x^{k_1+1}) + (1-\mu) f_2(x^{k_2+1}) = d \quad . \tag{40}$$

Then

$$x^* = \mu x^{k_1+1} + (1-\mu) x^{k_2+1} \tag{41}$$

is optimal to SMCPLC.

Proof. By (9), x^* satisfies $f_2(x^*) = d$, and $f(x^*)$ is efficient since $f(x^{k_1+1})$ and $f(x^{k_2+1})$ are adjacent efficient vertices by (38), (39) and Lemma 3 (see Fig. 4). Thus, there is no $x \in X$ such that $f_1(x) < f_1(x^*)$ and $f_2(x) \leq f_2(x^*)$ hold. This proves the lemma. □

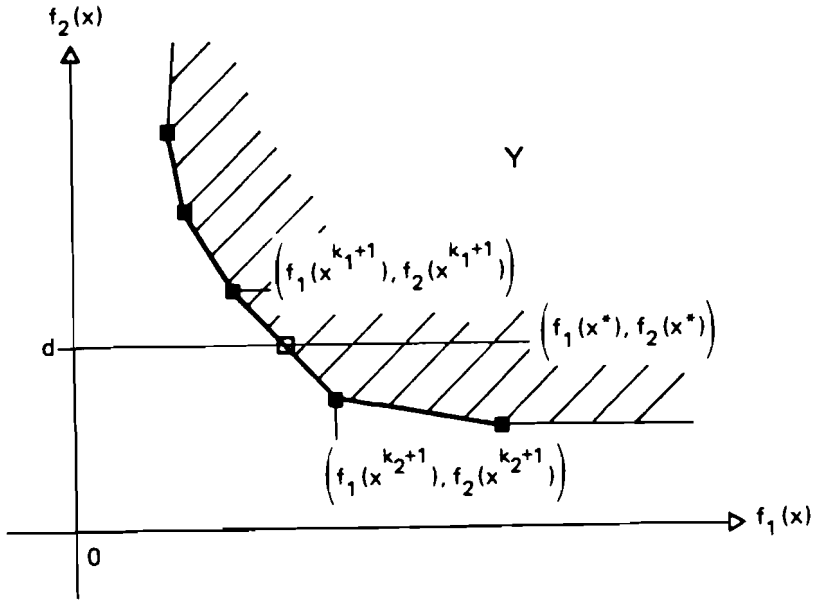

Fig. 4

Figure 4. Illustration of the set Y used in Lemma 7.

By the lemma, all what we do is to compute $\lambda_{(k_1)}$ and $\lambda_{(k_2)}$. Once $\lambda_{(k_1)}$ and $\lambda_{(k_2)}$ are obtained, x^{k_1+1} (resp. x^{k_2+1}) are computed by solving $P(\lambda_{(k_1)} + \varepsilon)$ (resp. $P(\lambda_{(k_2)} + \varepsilon)$, where ε satisfies $0 < \varepsilon < 1/M^2$. We shall explain only how $\lambda_{(k_1)}$ is computed (the case of $\lambda_{(k_2)}$ is similarly treated). This is done in a manner similar to the way of finding $\lambda_{(k^*)}$ in Algorithm SOLVEBMCP given in Section 4. Following the GT-algorithm to solve $P(\lambda_{(k_1)})$ without knowing the exact value of $\lambda_{(k_1)}$, every time comparison is made at conditional jump operation, we compute a critical value $\hat{\lambda}$ by solving the linear equation in $\lambda_{(k_1)}$ formed by the compared two numbers containing $\lambda_{(k_1)}$. We first test whether $\hat{\lambda}$ is a joint point or not, using Lemma 6. If $\hat{\lambda}$ is a joint point (say $\lambda_{(k)}$), we solve $P(\hat{\lambda} + \varepsilon)$ to obtain x^{k+1} and compute $f_2(x^{k+1})$. According to whether $f_2(x^{k+1}) \leq d$ or not, $\hat{\lambda} \geq \lambda_{(k_1)}$, or $\hat{\lambda} < \lambda_{(k_1)}$ is concluded respectively, and the proper computation path is chosen. If $\hat{\lambda}$ is not a joint point,

we solve $P(\hat{\lambda})$ to obtain $\hat{x} \in X^*(\hat{\lambda})$ and compute $f_2(\hat{x})$. According to whether $f_2(\hat{x}) \leq d$ or not, $\hat{\lambda} > \lambda_{(k_1)}$ or $\hat{\lambda} < \lambda_{(k_1)}$ is concluded respectively, and the proper computation path is chosen. In any case, by the discussion similar to the one in Sections 3 and 4, we finally obtain $\lambda_{(k_1)}$. We do not give the details of the algorithm since it is almost the same as SOLVEBMCP. In addition, we can also apply the idea of the time reduction given in Section 5 and hence the following theorem holds.

Theorem 3. Problem SMCPLC can be solved in

$$O(\min\{n^6 \log^3 n,\ n^4(n \log n + m)\log^5 n\}) \text{ time} \quad .$$

We now turn our attention to another type of problem to which the idea similar to the one given in Sections 3 and 4 can also be applied. Recall that Problem BMCP in (11) arises in interactive multicriteria decision making. Consider the situation in which only aspiration level $q = (q_1, q_2)$ is specified by the decision maker and q is unattainable. In this case, the distance between $f(x)$ and q can be considered to represent a measure of regret resulting from unattainability of $f(x)$ to q, instead of considering the achievement function such as s in (3)(see Figure 5). The following weighted l_p-norm has been considered in the literature to measure such distance (see [19, 27]).

$$d(f(x), q) = \sum_{i=1}^{2} (\alpha_i |f_i(x) - q_i|^p)^{\frac{1}{p}}, \quad \alpha_i > 0 \quad . \tag{42}$$

Here p is a positive integer and α_i are given constants. For the ease of exposition, we assume

$$q_1 \leq f_1(x^1) \text{ and } q_2 \leq f_2(x^{N+1}) \quad , \tag{43}$$

where x^k are those defined in (36). The other case such as $q_1 > f_1(x^1)$ or $q_2 > f_2(x^{N+1})$ are treated later. We therefore consider the following problem:

$$\begin{aligned} &\text{BMCP2: minimize } d(f(x), q) \\ &\qquad \text{subject to the constraints (7) and (8).} \end{aligned} \tag{44}$$

Since $q \notin Y$ is assumed, it is clear by (43) that $f(x^*)$ is efficient for any optimal solution x^* of BMCP2. The efficient set Y_0 is represented by a function of one parameter y_1 with $f_1(x^1) \leq y_1 \leq f_1(x^{N+1})$ as follows.

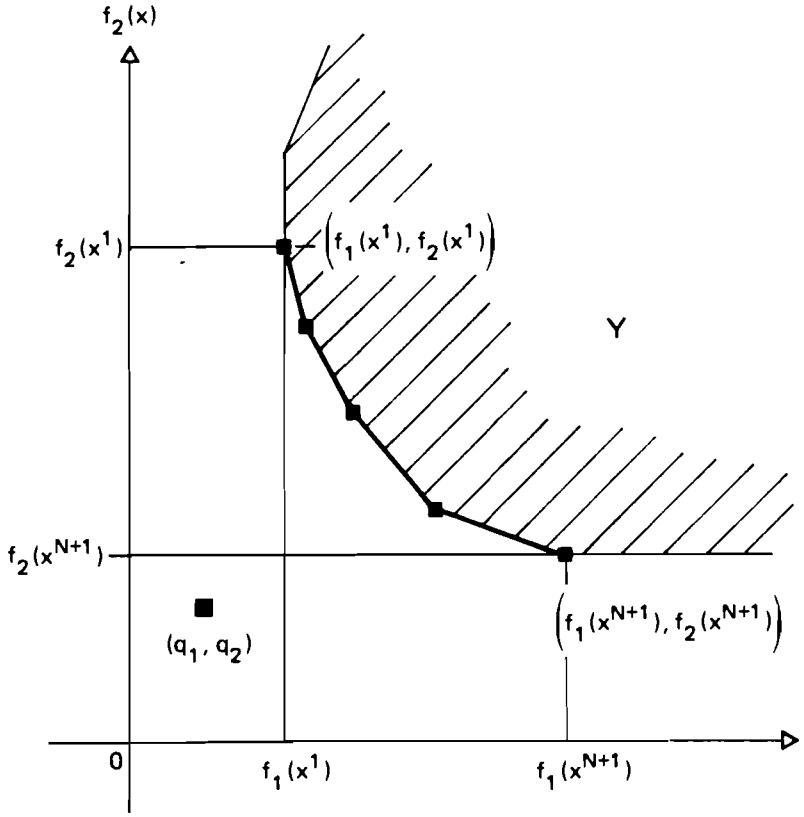

Figure 5. Illustration of point q and the set Y.

$$\{(y_1, \lambda_{(k)}^{-1}(f_1(x^k) - y_1) + f_2(x^k)) | f_1(x^k) \le y_1 \le f_1(x^{k+1}), \quad k = 1, \cdots, N\} \quad . \tag{45}$$

By (43) and (45), the objective function $d(f(x),q)$ is then represented by

$$(\alpha_1(y_1 - q_1)^p + \alpha_2(\lambda_{(k)}^{-1}(f_1(x^k) - y_1) + f_2(x^k) - q_2)^p)^{\frac{1}{p}} \quad , \tag{46}$$

if $f(x)$ is on the efficient edge between the vertices $f(x^k)$ and $f(x^{k+1})$. Define, for y_1 with $f_1(x^k) \le y_1 \le f_1(x^{k+1})$,

$$g(y_1,k) \equiv \alpha_1(y_1 - q_1)^p + \alpha_2(\lambda_{(k)}^{-1}(f_1(x^k) - y_1) + f_2(x^k) - q_2)^p \quad , \tag{47}$$

and define

$$g(y_1) \equiv \begin{cases} +\infty & \text{if } y_1 < f_1(x^1) \\ g(y_1,k) & \text{if } f_1(x^k) \le y_1 \le f_1(x^{k+1}),\ k = 1,\ldots,N \\ +\infty & \text{if } y_1 > f_1(x^{N+1}) \end{cases} \quad . \tag{48}$$

Since $g(y_1)$ is clearly convex and Problem BMCP2 is equivalent to minimizing $g(y_1)$, Problem BMCP2 is reduced to find y_1^* such that

$$\partial g(y_1^*) = 0 \quad , \tag{49}$$

where $\partial g(y_1)$ denotes the subgradient of $g(y_1)$. Let for $(y_1, y_2) \in Y_0$ and $\lambda > 0$

$$\delta(y_1, y_2, \lambda) \equiv \alpha_1 p (y_1 - q_1)^{p-1} - \lambda^{-1} \alpha_2 p (y_2 - q_2)^{p-1} \quad . \tag{50}$$

From (46), (47) and (48), we have

$$\partial g(y_1) = \begin{cases} \{\mu \mid -\infty < \mu \le \delta(f_1(x^1), f_2(x^1), \lambda_{(1)})\}, & y_1 = f_1(x^1) \\ \{\mu \mid \delta(f_1(x^k), f_2(x^k), \lambda_{(k-1)}) \le \mu \le \delta(f_1(x^k), f_2(x^k), \lambda_{(k)})\}, & \\ & y_1 = f_1(x^k),\ k = 2, \ldots, N \\ \{\mu \mid \delta(f_1(x^{N+1}), f_2(x^{N+1}), \lambda_{(N)}) \le \mu < +\infty\}, & y_1 = f_1(x^{N+1}) \\ \delta(y_1, \lambda_{(k)}^{-1}(f_1(x^k) - y_1) + f_2(x^k), \lambda_{(k)}), & f_1(x^k) < y_1 < f_1(x^{k+1}),\ k = 1, \ldots, N \end{cases} \tag{51}$$

Lemma 8.

(i) If $\delta(f_1(x^1), f_2(x^1), \lambda_{(1)}) > 0$, any x^1 is optimal to BMCP2.

(ii) If $\delta(f_1(x^{N+1}), f_2(x^{N+1}), \lambda_{(N)}) < 0$, any x^N is optimal to BMCP2.

(iii) If $\delta(f_1(x^{k^*}), f_2(x^{k^*}), \lambda_{(k^*-1)}) \le 0$ and $\delta(f_1(x^{k^*}), f_2(x^{k^*}), \lambda_{(k^*)}) \ge 0$ hold for some k^* with $2 \le k^* \le N$, any x^{k^*} is optimal to BMCP2.

(iv) If $\delta(f_1(x^{k^*}), f_2(x^{k^*}), \lambda_{(k^*)}) \le 0$ and $\delta(f_1(x^{k^*+1}), f_2(x^{k^*+1}), \lambda_{(k^*)}) \ge 0$ hold for some k^* with $2 \le k^* \le N$,

$$x^* = \mu x^{k^*} + (1 - \mu) x^{k^*+1} \tag{52}$$

is optimal to BMCP2, where μ satisfies

$$\begin{aligned} &\alpha_1 (\mu f_1(x^{k^*}) + (1-\mu) f_1(x^{k^*+1}) - q_1)^{p-1} \\ &= \lambda_{(k^*)}^{-1} \alpha_2 (\mu f_2(x^{k^*}) + (1 - \mu) f_2(x^{k^*+1}) - q_2)^{p-1} \quad . \end{aligned} \tag{53}$$

Proof. (i), (ii) and (iii) are obvious from (48) and (49). (iv) is proved as follows. Since $g(y_1)$ is convex, there exists (y_1, y_2) such that (y_1, y_2) is on the edge connecting $(f_1(x^{k^*}), f_2(x^{k^*}))$ and $(f_1(x^{k^*+1}), f_2(x^{k^*+1}))$ and $0 \in \partial g(y_1)$. Since the point (y_1, y_2) on this edge is represented as $(\mu f_1(x^{k^*}) + (1 - \mu) f_1(x^{k^*+1}), \mu f_2(x^{k^*}) + (1 - \mu) f_2(x^{k^*+1}))$ by using the parameter μ with $0 \le \mu \le 1$ and

$\partial g(f_1(x^*)) = 0$ holds by (50) and (53), x^* of (52) is optimal to BMCP2.

Based on Lemma 8, we can construct an algorithm for solving BMCP2 which is similar to the one presented in Sections 4 and 5. The conditions of Lemma 8 (i) and (ii) can be verified simply by solving $P(\lambda)$ for $\lambda < \lambda_{(1)}$ and $\lambda > \lambda_{(N)}$ respectively, which is similar to Lemma 4 (i) and (ii). If none of the conditions of Lemma 8 (i) and (ii) holds, there exists k^* satisfying the condition of Lemma 8 (iii) or (iv). To compute an optimal solution x^* for this case, we compute

$$\lambda' = \max\{\lambda_{(k)} | 1 \le k \le N, \delta(f_1(x^k), f_2(x^k), \lambda_{(k)}) \le 0\} \quad , \tag{54}$$

$$\lambda'' = \min\{\lambda_{(k)} | 1 \le k \le N, \delta(f_1(x^{k+1}), f_2(x^{k+1}), \lambda_{(k)}) \ge 0\} \quad . \tag{55}$$

Suppose that these two values are obtained. We then consider the following two cases.

Case 1. $\lambda' = \lambda''$. Then the condition of Lemma 8 (iv) holds for $\lambda_{(k^*)} = \lambda'$. x^{k^*} and x^{k^*+1} are obtained by solving $P(\lambda_{(k^*)} - \varepsilon)$ and $P(\lambda_{(k^*)} + \varepsilon)$ respectively, where ε satisfies $0 < \varepsilon < 1/M^2$. An optimal solution x^* of BMCP2 is then computed by (52). The value of μ in (52) is obtained by solving the following linear equation in μ which is equivalent to (53).

$$\mu f_1(x^{k^*}) + (1-\mu) f_1(x^{k^*+1}) - q_1 = \left(\frac{\alpha_2}{\alpha_1 \lambda_{(k^*)}}\right)^{\frac{1}{p-1}} (\mu f_2(x^{k^*}) + (1-\mu) f_2(x^{k^*+1}) - q_2) \quad . \tag{56}$$

Case 2. $\lambda' < \lambda''$. Then λ' and λ'' are two consecutive joint points. Otherwise there exists a joint point $\lambda_{(k')}$ with $\lambda' < \lambda_{(k')} < \lambda''$. Since the values of $\delta(f_1(x^k), f_2(x^k), \lambda_{(k)})$ and $\delta(f_1(x^{k+1}), f_2(x^{k+1}), \lambda_{(k)})$ are respectively increasing in k by the convexity of $g(y_1)$, it follows from the maximality and the minimality of λ' and λ'' respectively that

$$\delta(f_1(x^{k'}), f_2(x^{k'}), \lambda_{(k')}) > 0 \text{ and } \delta(f_1(x^{k'+1}), f_2(x^{k'+1}), \lambda_{(k')}) < 0$$

holds. This is, however, impossible by $f_1(x^{k'}) < f_1(x^{k'+1})$ and $f_2(x^{k'}) > f_2(x^{k'+1})$. So let $\lambda' = \lambda_{(k^*-1)}$ and $\lambda'' = \lambda_{(k^*)}$. By the maximality of λ' and the minimality of λ'', we have

$$\delta(f_1(x^{k^*}), f_2(x^{k^*}), \lambda_{(k^*-1)}) < 0 \text{ and } \delta(f_1(x^{k^*}), f_2(x^{k^*}), \lambda_{(k^*)}) > 0 \quad .$$

This is equivalent to the condition of Lemma 8 (iii). Therefore by Lemma 8 (iii) an optimal solution of BMCP2 is obtained by solving $P(\lambda)$ for a λ with

$\lambda' < \lambda < \lambda''$. $\square$.

With this observation, what remains to do is to compute λ' and λ'' of (54) and (55) respectively. This is done in a manner similar to the ways of computing $\lambda_{(k^*)}$ explained in Sections 3 and 4 and of computing $\lambda_{(k_1)}$ and $\lambda_{(k_2)}$ of (38) and (39) explained in this section. Therefore, the details are omitted here.

Finally, we mention the case in which (43) does not hold. If $q_1 > f_1(x^{N+1}), f(x^*)$ may not be efficient for an optimal solution x^* of BMCP2 (see Figure 6). However, if we consider the following new parametric problem for a nonnegative parameter λ,

$$P'(\lambda): \underset{x \in X}{\text{minimize}} \; -f_1(x) + \lambda f_2(x) \quad ,$$

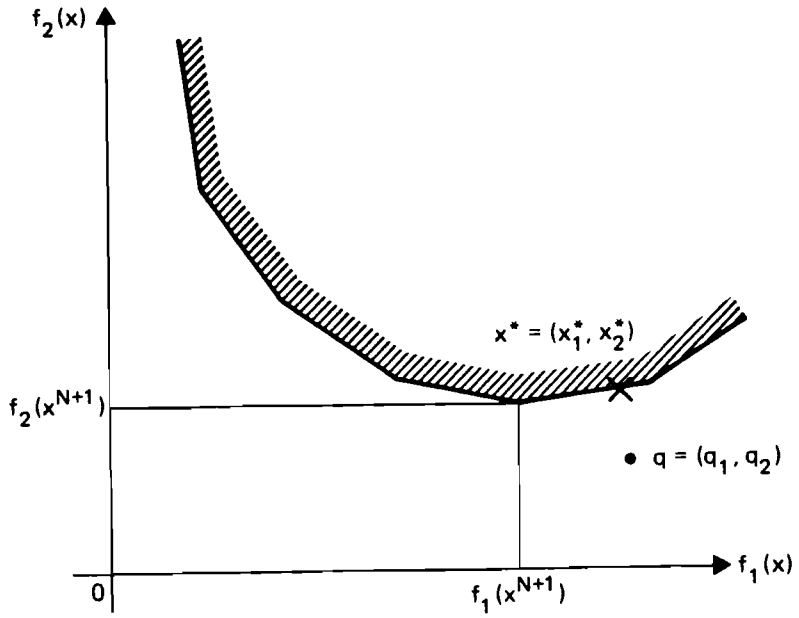

Figure 6. Illustration of the case in which $q_1 > f_1(x^{N+1})$ holds.

we can have the property relating $P'(\lambda)$ to BMCP2 which is similar to Lemma 8 (though the details are omitted here). The case of $q_2 > f_2(x^1)$ can be similarly treated. Thus we assume that

$$q_1 \leq f_1(x^{N+1}) \text{ and } q_2 \leq f_2(x^1) \tag{57}$$

hold.

Let us consider the case $q_1 > f_1(x^1)$. Since $q_1 \leq f_1(x^{N+1})$ is assumed by (57), there exists an efficient point $(f_1(\hat{x}), f_2(\hat{x}))$ with $f_1(\hat{x}) = q_1$ (see Fig. 7). Since the objective value $d(f(x), q)$ is larger than $d(f(\hat{x}), q)$ for all efficient $f(x)$ with $f_1(x) < q_1$, since if $f(x)$ is efficient and $f_1(x) < q_1$, $|f_1(x) - q_1| > |f_1(\hat{x}) - q_1| = 0$ and $|f_2(x) - q_2| > |f_2(\hat{x}) - q_2|$ hold (the last inequality follows from $f_2(x) > f_2(\hat{x})$ and $f_2(\hat{x}) < q_2$). Thus, we can eliminate all efficient $f(x)$ with $f_1(x) < q_1$, and consider $\hat{x}$ as if it is x^1. Therefore this case can be reduced to $q_1 \leq f_1(x^1)$. The case of $q_2 > f_2(x^{N+1})$ can be reduced to the case of $q_2 \leq f_2(x^{N+1})$ in a similar manner. Finally, notice that the solution $\hat{x}$ considered above is computed by solving the following problem.

minimize $f_2(x)$

subject to (7), (8) and

$$f_1(x) \leq d \quad .$$

As discussed at the beginning of this section, this problem can be solved in $O(\min\{n^6 \log^3 n, n^4(n \log n + m)\log^5 n\})$ time and its optimal solution x^* always satisfies $f_1(x^*) = d$. As a result, we have the following theorem.

Theorem 4. Problem BMCP2 can be solved in $O(\min\{n^6 \log^3 n, n^4(n \log n + m)\log^5 n\})$ time.

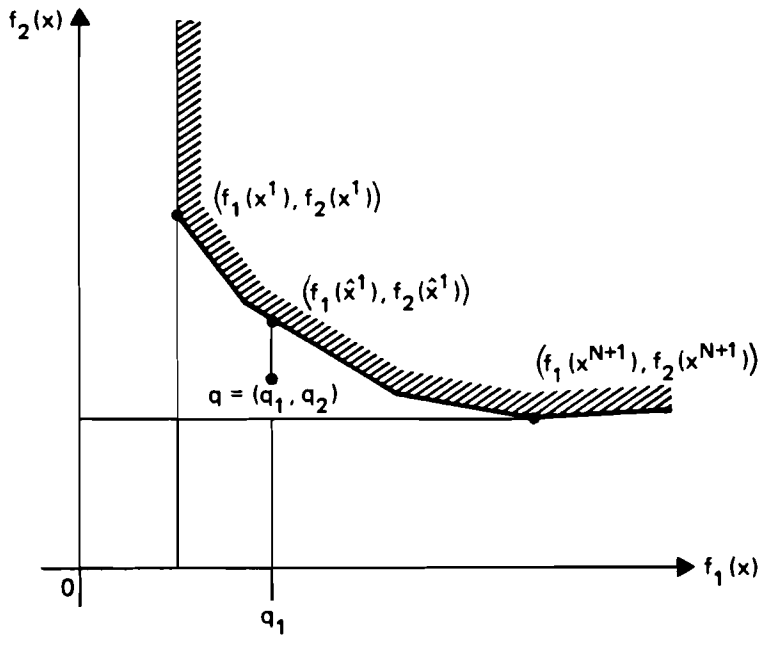

Figure 7. Illustration of the case in which $q_1 \leq f_1(x^{N+1})$ and $q_2 \leq f_2(x^1)$ hold.

References

[1] V.J. Bowman, Jr., On the relationship of the Chebyshev norm and efficient frontier of multiple-criteria objectives, In H. Thiriez and S. Zionts (eds.) Multiple criteria decision making. Springer, Berlin, Heidelberg, New York, *Lecture Notes in Economic and Mathematical Systems*, Vol. 130, 1976.

[2] P. Brucker, Parametric programming and circulation problems with one additional linear constraint, *Osnabrücker Schriften zur Mathematik*, Reihe P, Heft 56, Fachbereich Mathematik, Universität Osnabrück, 1983.

[3] E. Dekel, D. Nassimi and S. Sahni, Parallel matrix and graph algorithms, *SIAM J. Comput.*, 10 (1981), 657-675.

[4] W. Dinkelbach, Über einen Lösungsansatz zum Vectormaximumproblem. In M. Beckman (ed) Unternehmungsforschung Heute. Springer, Berlin, Heidelberg, New York, *Lecture Notes in Operational Research and Mathematical Systems*, Vol. 50, 1-30, 1971.

[5] J. Edmonds and R. Karp, Theoretical improvements in the algorithmic efficiency for network flow problems, *J. ACM* 19 (1972), 248-264.

[6] M.L. Fredman and R.E. Tarjan, Fibonacci heaps and their uses, Proceedings of 25th Annual IEEE Symposium on Foundations of Computer Science, (1984), 225-231.

[7] S. Fujishige, A capacity-rounding algorithm for the minimum-cost circulation problem: a dual framework of the Tardos algorithm, *Mathematical Programming*, 35 (1986), 298-308.

[8] T. Gal, Linear parametric programming - a brief survey, *Mathematical Programming Study*, 21 (1984), 43-68.

[9] Z. Galil and E. Tardos, An $O(n^2(m + n \log n)\log n)$ min-cost flow algorithm, Proceedings of 27th Annual IEEE Symposium on Foundations of Computer Science, (1986), 1-9.

[10] D. Gusfield, Parametric combinatorial computing and a problem of program module distribution, *J. ACM*, 30 (1983), 551-563.

[11] N. Katoh and Y. Ibaraki, A parametric characterization and an ε -approximation scheme for the minimization of a quasiconcave program, *Discrete Applied Mathematics*, 17 (1987) 39-66.

[12] L. Kučera, Parallel computation and conflicts in memory access, *Information Processing Letters*, 14 (1982), 93-96.

[13] E.L. Lawler, Combinatorial Optimization: Networks and Matroids, Holt, Rinehart & Winston, New York, 1976.

[14] N. Megiddo, Combinatorial optimization with rational objective functions, *Math. Oper. Res.*, 4 (1979), 414-424.

[15] N. Megiddo, Applying parallel computation algorithms in the design of serial algorithms, *J. ACM*, 30 (1983), 852-865.

[16] H. Nakayama, On the components in interactive multiobjective programming methods (in M. Grauer, M. Thompson, A.P. Wierzbicki, editors: Plural Rationality and Interactive Decision Processes, Proceedings, 1984), Springer Verlag, Berlin, 1985.

[17] J.B. Orlin, Genuinely polynomial simplex and non-simplex algorithm for the minimum cost flow problem, Working Paper No. 1615-84, A.P. Sloan School of Management, MIT, December 1984.

[18] C.H. Papadimitriou and K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[19] Y. Sawaragi, H. Nakayama and T. Tanino, Theory of multiobjective optimization, Academic Press, New York, 1985.

[20] R.E. Steuer and E.V. Choo, An interactive weighted Chebyshev procedure for multiple objective programming, *Mathematical Programming*, 26 (1983), 326-344.

[21] E. Tardos, A strongly polynomial minimum cost circulation algorithm, *Combinatorica*, 5 (1985), 247-255.

[22] A.P. Wierzbicki, Penalty methods in solving optimization problems with vector performance criteria, Proceedings of the 6th IFAC World Congress, Cambridge-Boston, 1975.

[23] A.P. Wierzbicki, Basic properties of scalarizing functionals for multiobjective optimization, Mathematische Operationsforschung und Statistik. *Optimization*, 8 (1977), 55-60.

[24] A.P. Wierzbicki, On the use of penalty functions in multiobjective optimization, In W. Oettli, F. Steffens, et al., editors: Proceedings of the 3rd Symposium on Operational Research, Universität Mannheim, Athenaum, 1978.

[25] A.P. Wierzbicki, A mathematical basis for satisficing decision making, *Mathematical Modelling*, 3 (1982), 391-405.

[26] A.P. Wierzbicki, On the completeness and constructiveness of parametric characterizations to vector optimization problems, *OR Spektrum*, 8 (1986), 73-87.

[27] A.P. Wierzbicki and A. Lewandowski, Dynamic Interactive Decision Analysis and Support, Working Paper, IIASA, Laxenburg, Austria, 1987.