SIAM J. Discrete Math. © 2006 Society for Industrial and Applied Mathematics
Vol. 19, No. 4, pp. 1065–1073

BISUBMODULAR FUNCTION MINIMIZATION*

SATORU FUJISHIGE† AND SATORU IWATA‡

Abstract. This paper presents the first combinatorial polynomial algorithm for minimizing bisubmodular functions, extending the scaling algorithm for submodular function minimization due to Iwata, Fleischer, and Fujishige. Since the rank functions of delta-matroids are bisubmodular, the scaling algorithm naturally leads to the first combinatorial polynomial algorithm for testing membership in delta-matroid polyhedra.

Key words. bisubmodular function, delta-matroid, scaling algorithm

AMS subject classification. 90C27

DOI. 10.1137/S0895480103426339

1. Introduction.

Let V be a finite nonempty set of cardinality n and 3^V denote the set of ordered pairs of disjoint subsets of V. Two binary operations $\sqcup$ and $\sqcap$ on 3^V are defined by

$$(X_1, Y_1) \sqcup (X_2, Y_2) = ((X_1 \cup X_2)\backslash(Y_1 \cup Y_2), (Y_1 \cup Y_2)\backslash(X_1 \cup X_2)),$$
$$(X_1, Y_1) \sqcap (X_2, Y_2) = (X_1 \cap X_2, Y_1 \cap Y_2).$$

A function $f : 3^V \to \mathbf{R}$ is called *bisubmodular* if it satisfies

$$f(X_1, Y_1) + f(X_2, Y_2) \geq f((X_1, Y_1) \sqcup (X_2, Y_2)) + f((X_1, Y_1) \sqcap (X_2, Y_2))$$

for any (X_1, Y_1) and (X_2, Y_2) in 3^V. This paper presents the first combinatorial polynomial algorithm for minimizing bisubmodular functions.

Examples of bisubmodular functions include the rank functions of delta-matroids introduced independently by Bouchet [3] and Chandrasekaran–Kabadi [6]. A delta-matroid is a set system $(V, \mathcal{F})$ with $\mathcal{F}$ being a nonempty family of subsets of V that satisfies the following exchange property:

$$\forall F_1, F_2 \in \mathcal{F}, \forall v \in F_1 \triangle F_2, \exists u \in F_1 \triangle F_2 : F_1 \triangle \{u, v\} \in \mathcal{F},$$

where $\triangle$ denotes the symmetric difference. A slightly restricted set system with an additional condition $\emptyset \in \mathcal{F}$ had been introduced by Dress–Havel [11]. A member of $\mathcal{F}$ is called a feasible set of the delta-matroid. Note that the base and the independent-set families of a matroid satisfy this exchange property. Thus, a delta-matroid is a generalization of a matroid.

Chandrasekaran–Kabadi [6] showed that the rank function $\varrho : 3^V \to \mathbf{Z}$ defined by

$$\varrho(X, Y) = \max\{|X \cap F| - |Y \cap F| \mid F \in \mathcal{F}\}$$

*Received by the editors April 25, 2003; accepted for publication (in revised form) September 22, 2005; published electronically February 3, 2006. A preliminary version of this paper appeared in *Proceedings of the Eighth Conference on Integer Programming and Combinatorial Optimization*, LNCS 2081, Springer-Verlag, Berlin, 2001, pp. 160–169.

http://www.siam.org/journals/sidma/19-4/42633.html

†Research Institute for Mathematical Sciences, Kyoto University, Kyoto 606-8502, Japan (fujishig@kurims.kyoto-u.ac.jp).

‡Department of Mathematical Informatics, University of Tokyo, Tokyo 113-8656, Japan (iwata@mist.i.u-tokyo.ac.jp).

is bisubmodular. The convex hull of the characteristic vectors of the feasible sets is described by

$$\mathrm{P}(\varrho) = \{x \mid x \in \mathbf{R}^V,\, \forall (X,Y) \in 3^V : x(X) - x(Y) \le \varrho(X,Y)\},$$

which is called the delta-matroid polyhedron. This fact follows from the greedy algorithm [3, 6] for optimizing a linear function over the feasible sets.

Given a vector $x \in \mathbf{R}^V$, one can test if x belongs to $\mathrm{P}(\varrho)$ by minimizing a bisubmodular function $f(X,Y) = \varrho(X,Y) - x(X) + x(Y)$. Even for such a special case of bisubmodular function minimization, no combinatorial algorithm was known to run in polynomial time. This is in contrast with the matroid polyhedron, for which Cunningham [7] devised a combinatorial strongly polynomial algorithm for testing membership.

A simple example of a delta-matroid is a matching delta-matroid [4], whose feasible sets are the perfectly matchable vertex subsets of an undirected graph. The corresponding delta-matroid polyhedron is the matchable set polytope [2]. For this special case, Cunningham–Green-Krótki [10] developed an augmenting path algorithm for solving the separation problem in polynomial time with the aid of the scaling technique.

A bisubmodular function also generalizes a submodular (set) function. Let 2^V denote the family of all the subsets of V. A function $g : 2^V \to \mathbf{R}$ is called *submodular* if it satisfies

$$g(Z_1) + g(Z_2) \ge g(Z_1 \cup Z_2) + g(Z_1 \cap Z_2)$$

for any $Z_1, Z_2 \subseteq V$. For a submodular function g, we define a bisubmodular function $f : 3^V \to \mathbf{R}$ by

$$f(X,Y) = g(X) + g(V\backslash Y) - g(V).$$

If (X,Y) is a minimizer of f, then both X and $V\backslash Y$ are minimizers of g. Thus, bisubmodular function minimization (BSFM) is a generalization of submodular function minimization.

The first polynomial algorithm for submodular function minimization is given by Grötschel–Lovász–Schrijver [16]. They also give the first strongly polynomial algorithms in [17]. Their algorithms rely on the ellipsoid method, which is not efficient in practice. Two combinatorial strongly polynomial algorithms have been devised independently by Schrijver [23] and Iwata–Fleischer–Fujishige [18]. Both of these new algorithms are based on the combinatorial pseudopolynomial algorithm given by Cunningham [8]. The algorithm of Schrijver [23] directly achieves a strongly polynomial bound, whereas Iwata–Fleischer–Fujishige [18] develop a scaling algorithm with weakly polynomial time complexity and then convert it to a strongly polynomial one.

In the present paper, we extend the scaling algorithm of Iwata–Fleischer–Fujishige [18] to solve the minimization problem for integer-valued bisubmodular functions. The resulting algorithm runs in $\mathrm{O}(n^5 \log M)$ time, where M designates the maximum value of f. This bound is weakly polynomial. A strongly polynomial version will be presented in a forthcoming paper by McCormick–Fujishige [21].

As a generalization of the delta-matroid polyhedron, a *bisubmodular polyhedron*

$$\mathrm{P}(f) = \{x \mid x \in \mathbf{R}^V,\, \forall (X,Y) \in 3^V : x(X) - x(Y) \le f(X,Y)\}$$

is associated with a general bisubmodular function $f : 3^V \to \mathbf{R}$, where we assume $f(\emptyset,\emptyset) = 0$. The linear optimization problem over the bisubmodular polyhedron can be solved by the greedy algorithm of Dunstan–Welsh [12]. Based on the validity of this greedy algorithm, Qi [22] established a connection between bisubmodular functions and their convex extensions. This generalizes a result of Lovász [19] on submodular functions. Qi [22] also mentioned that the connection provides a polynomial algorithm for bisubmodular function minimization using the ellipsoid method. In contrast, our combinatorial algorithm directly deals with the separation problem for bisubmodular polyhedra.

The concept of delta-matroid is extended to that of jump system by Bouchet–Cunningham [5]. A jump system is a set of lattice points satisfying a certain axiom. Examples include the set of degree sequences of a graph [9]. Lovász [20] investigated the membership problem in jump systems and proved a min-max theorem for a fairly wide class of jump systems. The lattice points contained in an integral bisubmodular polyhedron form a jump system, called a convex jump system, and conversely the convex hull of a jump system is an integral bisubmodular polyhedron. An example of a convex jump system is the so-called b-matching degree sequence polyhedron [9]. A very recent paper of Zhang [24] presented an augmenting path separation algorithm for this polyhedron. The present paper provides a more general algorithmic approach to the membership problem in convex jump systems.

2. Bisubmodular polyhedra.

This section provides a preliminary on bisubmodular polyhedra. See [14, section 3.5 (b)] for more detail and background.

For any vector $y \in \mathbf{R}^V$, we denote $\|y\| = \sum_{v \in V} |y(v)|$. Concerning the shortest distance from a given vector x° to the bisubmodular polyhedron $\mathrm{P}(f)$, we have the following min-max relation, which is essentially equivalent to an earlier result of Cunningham–Green-Krótki [9].

THEOREM 2.1 (Fujishige [15]). *For any bisubmodular function $f : 3^V \to \mathbf{R}$ and any vector $x^\circ \in \mathbf{R}^V$,*

$$\min\{\|x - x^\circ\| \mid x \in \mathrm{P}(f)\} = \max\{x^\circ(X) - x^\circ(Y) - f(X, Y) \mid (X, Y) \in 3^V\}$$

holds. Moreover, if f and x° are integer valued, then there is an integral vector x that attains the minimum in the left-hand side.

When $x^\circ = \mathbf{0}$, the min-max relation characterizes the minimum value of f. Our combinatorial algorithm is built on this characterization.

We now turn to the greedy algorithm for computing an extreme point of a bisubmodular polyhedron. See also [1] for related structural results on extreme points of bisubmodular polyhedra.

Let $\sigma : V \to \{+, -\}$ be a sign function. For any subset $U \subseteq V$, we denote by $U|\sigma$ the pair $(X, Y) \in 3^V$ with $X = \{u \mid u \in U, \sigma(u) = +\}$ and $Y = \{u \mid u \in U, \sigma(u) = -\}$. We also write $f(U|\sigma) = f(X, Y)$ for any function $f : 3^V \to \mathbf{R}$, and $x(U|\sigma) = x(X) - x(Y)$ for any vector $x \in \mathbf{R}^V$.

Let $L = (v_1, \cdots, v_n)$ be a linear ordering of V. For each $j = 1, \ldots, n$, let $L(v_j) = \{v_1, \ldots, v_j\}$. The *greedy algorithm* with respect to L and a sign function σ assigns $y(v) := \sigma(v)\{f(L(v)|\sigma) - f(L(v)\backslash\{v\}|\sigma)\}$ for each $v \in V$. Then the resulting vector $y \in \mathbf{R}^V$ is an extreme point of the bisubmodular polyhedron $\mathrm{P}(f)$.

Given a weight function $w : V \to \mathbf{R}$, construct a linear ordering $L = (v_1, \ldots, v_n)$ and a sign function σ that satisfies $|w(v_1)| \geq \cdots \geq |w(v_n)|$ and $w(v) = \sigma(v)|w(v)|$ for each $v \in V$. Then the vector y generated by the greedy algorithm with respect to L and σ maximizes the linear function $\sum_{v \in V} w(v)y(v)$ over the bisubmodular polyhedron $\mathrm{P}(f)$.

3. Scaling algorithm. This section presents a scaling algorithm for minimizing an integer-valued bisubmodular function $f : 3^V \to \mathbf{Z}$, provided that an oracle for evaluating the function value is available.

The scaling algorithm works with a positive parameter δ. The algorithm keeps a vector $x \in \mathrm{P}(f)$ as a convex combination of extreme points of $\mathrm{P}(f)$ indexed by finite set I. Namely, $x = \sum_{i \in I} \lambda_i y_i$ with $\lambda_i > 0$ for each $i \in I$ and $\sum_{i \in I} \lambda_i = 1$. Each extreme point y_i is generated by the greedy algorithm with respect to L_i and σ_i. It also keeps a pair of functions $\varphi : V \times V \to \mathbf{R}$ and $\psi : V \times V \to \mathbf{R}$. The function φ is skew-symmetric, i.e., $\varphi(u,v) + \varphi(v,u) = 0$ for any $u, v \in V$, while ψ is symmetric, i.e., $\psi(u,v) = \psi(v,u)$ for any $u, v \in V$. These functions are called *δ-feasible* if they satisfy $-\delta \le \varphi(u,v) \le \delta$ and $-\delta \le \psi(u,v) \le \delta$ for any $u, v \in V$. The boundaries $\partial\varphi$ and $\partial\psi$ are defined by $\partial\varphi(u) = \sum_{v \in V} \varphi(u,v)$ and $\partial\psi(u) = \sum_{v \in V} \psi(u,v)$.

The algorithm starts with an extreme point $x \in \mathrm{P}(f)$ generated by the greedy algorithm with respect to a linear ordering L and a sign function σ. The initial value of δ is given by $\delta := \|x\|/n^2$, and the initial values of φ and ψ are zero.

Each scaling phase starts by cutting the value of δ in half. Then it modifies φ and ψ to make them δ-feasible. This can be done by setting each $\varphi(u,v)$ and $\psi(u,v)$ to the closest values in the interval $[-\delta, \delta]$.

The rest of the scaling phase aims at decreasing $\|z\|$ for $z = x + \partial\varphi + \partial\psi$. It uses three procedures: Augment, Double-Exchange, and Tail-Exchange. Procedure Augment decreases $\|z\|$ by δ, whereas Double-Exchange and Tail-Exchange modify x, φ, ψ keeping z invariant. The scaling phase terminates when none of these procedures are applicable. Then $\|z\|$ is shown to be small enough to keep moderate the number of applications of Augment in the next scaling phase. Furthermore, if $\delta < 1/3n^2$, then the algorithm detects a pair (X, Y) that minimizes f.

Given δ-feasible φ and ψ, the algorithm constructs an auxiliary directed graph $G(\varphi, \psi)$ as follows. Let V^+ and V^- be two copies of V. For each $v \in V$, we denote its copies by $v^+ \in V^+$ and $v^- \in V^-$. The vertex set of $G(\varphi, \psi)$ is $V^+ \cup V^-$. For any subset $U \subseteq V$, define $U^+ = \{u^+ \mid u \in U\}$ and $U^- = \{u^- \mid u \in U\}$. The arc set $A(\varphi, \psi) = A(\varphi) \cup A(\psi)$ of $G(\varphi, \psi)$ is defined by

$$A(\varphi) = \{(u^+, v^+) \mid u \ne v,\, \varphi(u,v) \le 0\} \cup \{(u^-, v^-) \mid u \ne v,\, \varphi(u,v) \ge 0\},$$
$$A(\psi) = \{(u^+, v^-) \mid \psi(u,v) \le 0\} \cup \{(u^-, v^+) \mid \psi(u,v) \ge 0\}.$$

Note that, for any $u, v \in V$ and any sign function η on V, the graph $G(\varphi, \psi)$ has a directed path from $u^{\eta(u)}$ to $v^{\eta(v)}$ if and only if $G(\varphi, \psi)$ has a directed path from $v^{-\eta(v)}$ to $u^{-\eta(u)}$.

Let $S = \{v \mid v \in V, z(v) \le -\delta\}$ and $T = \{v \mid v \in V, z(v) \ge \delta\}$. A simple directed path in $G(\varphi, \psi)$ from $S^+ \cup T^-$ to $S^- \cup T^+$ is called a *δ-augmenting path.* If there exists a δ-augmenting path P, the algorithm applies the following δ-augmentation to φ and ψ.

```
Augment($\delta, P, \varphi, \psi$):
    • For each $(u^+, v^+)$ in $P$, $\varphi(u,v) := \varphi(u,v) + \delta/2$ and $\varphi(v,u) := \varphi(v,u) - \delta/2$.
    • For each $(u^-, v^-)$ in $P$, $\varphi(u,v) := \varphi(u,v) - \delta/2$ and $\varphi(v,u) := \varphi(v,u) + \delta/2$.
    • For each $(u^+, v^-)$ in $P$, $\psi(u,v) := \psi(u,v) + \delta/2$ and $\psi(v,u) := \psi(v,u) + \delta/2$.
    • For each $(u^-, v^+)$ in $P$, $\psi(u,v) := \psi(u,v) - \delta/2$ and $\psi(v,u) := \psi(v,u) - \delta/2$.
```

Note that a δ-augmentation does not change x, maintains δ-feasibility, and decreases $\|z\|$ by δ.

After each δ-augmentation, the algorithm computes an expression of x as a convex combination of affinely independent extreme points of $\mathrm{P}(f)$ chosen from among $\{y_i \mid$

```
Double-Exchange$(i, u, v)$;
$\beta := f(L_i(u)\backslash\{v\}|\sigma_i) - f(L_i(u)|\sigma_i) + \sigma_i(v)y_i(v)$;
$\alpha := \min\{\delta,\ \lambda_i\beta\}$;
If $\alpha < \lambda_i\beta$ then
    $k \leftarrow$ a new index;
    $I := I \cup \{k\}$;
    $\lambda_k := \lambda_i - \alpha/\beta$;
    $\lambda_i := \alpha/\beta$;
    $y_k := y_i$;
    $L_k := L_i$;
Update $L_i$ by interchanging $u$ and $v$;
$y_i := y_i + \beta(\sigma_i(u)\chi_u - \sigma_i(v)\chi_v)$;
$x := x + \alpha(\sigma_i(u)\chi_u - \sigma_i(v)\chi_v)$;
If $\sigma_i(u) = \sigma_i(v)$ then
    $\varphi(u, v) := \varphi(u, v) - \sigma_i(u)\alpha$;
    $\varphi(v, u) := \varphi(v, u) + \sigma_i(u)\alpha$;
Else
    $\psi(u, v) := \psi(u, v) - \sigma_i(u)\alpha$;
    $\psi(v, u) := \psi(v, u) - \sigma_i(u)\alpha$.
```

FIG. 3.1. *Algorithmic description of Procedure* Double-Exchange(i, u, v).

$i \in I\}$. This can be done by a standard linear programming technique using Gaussian elimination.

If there is no δ-augmenting path, let $X^+ \subseteq V^+$ and $Y^- \subseteq V^-$ be the sets of vertices reachable by directed paths from $S^+ \cup T^-$. Then we have $S \subseteq X$, $T \subseteq Y$, and $X \cap Y = \emptyset$. For each $i \in I$, consider a pair of disjoint subsets $W_i = \{u \mid u^{\sigma_i(u)} \in X^+ \cup Y^-\}$ and $R_i = \{u \mid u^{\sigma_i(u)} \in X^- \cup Y^+\}$. Note that $W_i \cup R_i = X \cup Y$. We now introduce two procedures: Double-Exchange and Tail-Exchange.

Procedure Double-Exchange(i, u, v) is applicable if u immediately succeeds v in L_i and either $u \in W_i$ and $v \notin W_i$ or $u \notin R_i$ and $v \in R_i$ hold. Such a triple (i, u, v) is called *active*. The first step of the procedure is to compute

$$\beta := f(L_i(u)\backslash\{v\}|\sigma_i) - f(L_i(u)|\sigma_i) + \sigma_i(v)y_i(v).$$

Then it interchanges u and v in L_i and updates y_i as $y_i := y_i + \beta(\sigma_i(u)\chi_u - \sigma_i(v)\chi_v)$. The resulting y_i is an extreme point generated by the new linear ordering L_i and sign function σ_i.

If $\lambda_i\beta \le \delta$, Double-Exchange$(i, u, v)$ is called *saturating*. Otherwise, it is called *nonsaturating*. In the nonsaturating case, the procedure adds to I a new index k with y_k, σ_k and L_k being the previous y_i, σ_i and L_i, and assigns $\lambda_k := \lambda_i - \delta/\beta$ and $\lambda_i := \delta/\beta$. In both cases, x moves to $x := x + \alpha(\sigma_i(u)\chi_u - \sigma_i(v)\chi_v)$ with $\alpha = \min\{\delta, \lambda_i\beta\}$. In order to keep z invariant, the procedure finally modifies φ or ψ appropriately. If $\sigma_i(u) = \sigma_i(v)$, it updates $\varphi(u, v) := \varphi(u, v) - \sigma_i(u)\alpha$ and $\varphi(v, u) := \varphi(v, u) + \sigma_i(u)\alpha$. On the other hand, if $\sigma_i(u) \neq \sigma_i(v)$, then $\psi(u, v) := \psi(u, v) - \sigma_i(u)\alpha$ and $\psi(v, u) := \psi(v, u) - \sigma_i(u)\alpha$. A formal description of Double-Exchange is given in Figure 3.1.

LEMMA 3.1. *As a result of nonsaturating* Double-Exchange(i, u, v), *a new vertex joins* $X \cup Y$ *or a* δ*-augmenting path appears in* $G(\varphi, \psi)$.

```
Tail-Exchange($i, v$);
$\sigma_i(v) := -\sigma_i(v)$;
$\beta := f(V|\sigma_i) - f(V\backslash\{v\}|\sigma_i) - \sigma_i(v)y_i(v)$;
$\alpha := \min\{\delta,\ \lambda_i\beta\}$;
If $\alpha < \lambda_i\beta$ then
    $k \leftarrow$ a new index;
    $I := I \cup \{k\}$;
    $\lambda_k := \lambda_i - \alpha/\beta$;
    $\lambda_i := \alpha/\beta$;
    $y_k := y_i$;
    $L_k := L_i$;
$y_i := y_i + \sigma_i(v)\beta\chi_v$;
$x := x + \sigma_i(v)\alpha\chi_v$;
$\psi(v,v) := \psi(v,v) - \sigma_i(v)\alpha$.
```

FIG. 3.2. *Algorithmic description of Procedure* Tail-Exchange(i, v).

Proof. The nonsaturating Double-Exchange(i, u, v) updates φ or ψ so that a new arc $(u^{\sigma_i(u)}, v^{\sigma_i(v)})$ should appear in $G(\varphi, \psi)$. Recall that either $u \in W_i$ and $v \notin W_i$ or $u \notin R_i$ and $v \in R_i$ holds. If $u \in W_i$ and $v \notin R_i \cup W_i$, the new arc makes $v^{\sigma_i(v)}$ reachable from $S^+ \cup T^-$. If $v \in R_i$ and $u \notin W_i \cup R_i$, the new arc makes $u^{\sigma_i(u)}$ reachable to $S^- \cup T^+$. Thus, in these cases, a new vertex v or u is added to $X \cup Y$. Finally, if $u \in W_i$ and $v \in R_i$, the new arc yields a δ-augmenting path. ∎

Procedure Tail-Exchange(i, v) is applicable if v is the last element in L_i and $v \in R_i$. Such a pair (i, v) is also called *active*. The first step of the procedure is to reverse the sign $\sigma_i(v)$. It then computes

$$\beta := f(V|\sigma_i) - f(V\backslash\{v\}|\sigma_i) - \sigma_i(v)y_i(v)$$

and updates $y_i := y_i + \sigma_i(v)\beta\chi_v$. The resulting y_i is an extreme point generated by L_i and the new σ_i.

If $\lambda_i\beta \leq \delta$, Tail-Exchange$(i, v)$ is called *saturating*. Otherwise, it is called *nonsaturating*. In the nonsaturating case, the procedure adds to I a new index k with y_k, σ_k and L_k being the previous y_i, σ_i and L_i, and assigns $\lambda_k := \lambda_i - \delta/\beta$ and $\lambda_i := \delta/\beta$. In both cases, x moves to $x := x + \sigma_i(v)\alpha\chi_v$ with $\alpha = \min\{\delta, \lambda_i\beta\}$. In order to keep z invariant, the procedure finally modifies ψ as $\psi(v, v) := \psi(v, v) - \sigma_i(v)\alpha$. A formal description of Tail-Exchange is given in Figure 3.2.

LEMMA 3.2. *As a result of nonsaturating* Tail-Exchange(i, v), *a δ-augmenting path appears in $G(\varphi, \psi)$.*

Proof. Suppose the algorithm applies Tail-Exchange(i, v) with $\sigma_i(v) = \tau$. Then $v^{-\tau}$ is reachable from $S^+ \cup T^-$ and v^τ is reachable to $S^- \cup T^+$ in $G(\varphi, \psi)$. The nonsaturating Tail-Exchange(i, v) changes $\sigma_i(v)$ to $-\tau$ and updates $\psi(v, v)$ so that a new arc $(v^{-\tau}, v^\tau)$ appears in $G(\varphi, \psi)$, which yields a δ-augmenting path from $S^+ \cup T^-$ to $S^- \cup T^+$. ∎

If there is no δ-augmenting path and neither Double-Exchange nor Tail-Exchange is applicable, the algorithm terminates the scaling phase. Then it goes to the next scaling phase unless $\delta < 1/3n^2$. If $\delta < 1/3n^2$, the algorithm terminates by returning the current (X, Y) as a minimizer of f.

A formal description of our scaling algorithm BSFM is now given in Figure 3.3.

```
BSFM($f$):
Initialization:
    $L \leftarrow$ a linear ordering on $V$;
    $\sigma \leftarrow$ a sign function on $V$;
    $x \leftarrow$ an extreme vector in $\mathrm{P}(f)$ generated by $L$ and $\sigma$;
    $I := \{\ell\}$, $y_\ell := x$, $\lambda_\ell := 1$, $L_\ell := L$;
    $\varphi := \mathbf{0}$, $\psi := \mathbf{0}$;
    $\delta \leftarrow \|x\|/n^2$;
While $\delta \geq 1/3n^2$ do
    $\delta := \delta/2$;
    For $(u,v) \in V \times V$ do
        Change $\varphi(u,v)$ and $\psi(u,v)$ to the closest values in the interval $[-\delta,\delta]$;
    $S := \{v \mid x(v) + \partial\varphi(v) + \partial\psi(v) \leq -\delta\}$;
    $T := \{v \mid x(v) + \partial\varphi(v) + \partial\psi(v) \geq \delta\}$;
    $X^+ \leftarrow$ the set of vertices in $V^+$ reachable from $S^+ \cup T^-$ in $G(\varphi,\psi)$;
    $Y^- \leftarrow$ the set of vertices in $V^-$ reachable from $S^+ \cup T^-$ in $G(\varphi,\psi)$;
    $Q \leftarrow$ the set of active triples and active pairs;
    While $\exists\delta$-augmenting path or $Q \neq \emptyset$ do
        If $\exists P$: $\delta$-augmenting path then
            Augment($\delta, P, \varphi, \psi$);
            Update $S$, $T$, $X^+$, $Y^-$, $Q$;
            Express $x$ as $x = \sum_{i\in I} \lambda_i y_i$ by possibly smaller affinely independent
                    subset $I$ and positive coefficients $\lambda_i > 0$ for $i \in I$;
        Else
            While $\nexists\delta$-augmenting path and $Q \neq \emptyset$ do
                Find an active $(i,u,v) \in Q$ or active $(i,v) \in Q$;
                Apply Double-Exchange($i,u,v$) or Tail-Exchange($i,v$);
                Update $X^+$, $Y^-$, $Q$;
Return $(X,Y)$;
End.
```

FIG. 3.3. *A scaling algorithm for bisubmodular function minimization.*

4. Validity and complexity.

This section is devoted to the analysis of our scaling algorithm. We first discuss the validity.

LEMMA 4.1. *At the end of each scaling phase, the current $(X,Y) \in 3^V$ and $z = x + \partial\varphi + \partial\psi$ satisfy $\|z\| \leq 2n\delta - f(X,Y)$.*

Proof. At the end of each scaling phase, we have $y_i(X) - y_i(Y) = f(X,Y)$ for each $i \in I$. Hence, x satisfies $x(X) - x(Y) = f(X,Y)$. By the definition of (X,Y), we have $\varphi(u,v) > 0$ for $u \in X$, $v \in V \backslash X$ and $\varphi(u,v) < 0$ for $u \in Y$, $v \in V\backslash Y$. These inequalities imply $\partial\varphi(X) = \sum\{\varphi(u,v) \mid u \in X, v \in V\backslash X\} > 0$ and $\partial\varphi(Y) = \sum\{\varphi(u,v) \mid u \in Y, v \in V\backslash Y\} < 0$. Similarly, we have $\psi(u,v) > 0$ for $u \in X$, $v \in V\backslash Y$ and $\psi(u,v) < 0$ for $u \in Y$, $v \in V\backslash X$, which imply $\partial\psi(X) = \sum\{\psi(u,v) \mid u \in X, v \in V\} > \theta$ and $\partial\psi(Y) = \sum\{\psi(u,v) \mid u \in Y, v \in V\} < \theta$, where $\theta = \sum\{\psi(u,v) \mid u \in X, v \in Y\}$. Since $S \subseteq X$ and $T \subseteq Y$, we have $z(v) \geq -\delta$ for $v \in V\backslash X$ and $z(v) \leq \delta$ for $v \in V\backslash Y$. Therefore, we have $\|z\| \leq -z(X) + z(Y) + 2n\delta \leq -x(X) + x(Y) + 2n\delta = -f(X,Y) + 2n\delta$. ☐

THEOREM 4.2. *The algorithm obtains a minimizer of f at the end of the last scaling phase.*

Proof. Note that $\delta < 1/3n^2$ at the end of the last scaling phase. For each $v \in V$, since $|\partial\varphi(v)| \leq (n-1)\delta$ and $|\partial\psi(v)| \leq n\delta$, we have $|x(v)| \leq |z(v)|+|\partial\varphi(v)|+|\partial\psi(v)| \leq |z(v)|+(2n-1)\delta$. Then it follows from Lemma 4.1 that $\|x\| \leq (2n^2+n)\delta - f(X,Y) < 1 - f(X,Y)$. For any $(X',Y') \in 3^V$, we have $f(X',Y') \geq x(X') - x(Y') \geq -\|x\| > f(X,Y) - 1$. Hence (X,Y) is a minimizer of the integer-valued function f. ∎

We now give a running time bound of our algorithm.

LEMMA 4.3. *Each scaling phase performs* $\mathrm{O}(n^2)$ *augmentations.*

Proof. At the beginning of each scaling phase, the algorithm modifies φ and ψ to make them δ-feasible (for the new δ). This changes $\|z\|$ by at most $2n^2\delta$. Therefore, by Lemma 4.1, the pair (X,Y) obtained at the end of the previous scaling phase must satisfy $\|z\| \leq 2n^2\delta + 4n\delta - f(X,Y)$ after updating φ and ψ at the beginning of the current scaling phase. On the other hand, at the end of the current scaling phase, we have $\|z\| \geq -z(X)+z(Y) \geq -x(X)+x(Y)-2n^2\delta \geq -f(X,Y)-2n^2\delta$. Thus, during the scaling phase, $\|z\|$ decreases by at most $4n\delta + 4n^2\delta$. Since each δ-augmentation decreases $\|z\|$ by δ, the number of δ-augmentations in the scaling phase is at most $4n^2+4n$, which is $\mathrm{O}(n^2)$. ∎

LEMMA 4.4. *The algorithm performs Procedure* Double-Exchange $\mathrm{O}(n^3)$ *times and* Tail-Exchange $\mathrm{O}(n^2)$ *times between δ-augmentations.*

Proof. In Double-Exchange, a vertex in W_i moves in L_i ahead of some vertex not in W_i and/or a vertex in R_i moves behind some vertex not in R_i. Procedure Tail-Exchange changes a vertex of R_i into W_i. No vertex goes out of W_i. A vertex of R_i can be switched to W_i by Tail-Exchange. However, it does not go out of $R_i \cup W_i$. Thus, for each $i \in I$, after at most $\mathrm{O}(n^2)$ applications of Double-Exchange and $\mathrm{O}(n)$ applications of Tail-Exchange to $i \in I$, the subset R_i is empty and $W_i = L_i(w)$ holds for some $w \in V$. At this point, neither Double-Exchange nor Tail-Exchange is applicable to $i \in I$.

After each δ-augmentation, the algorithm updates the convex combination $x = \sum_{i\in I}\lambda_i y_i$ so that $|I| \leq n+1$. A new index is added to I as a result of nonsaturating Double-Exchange(i,u,v) and Tail-Exchange(i,v). It follows from Lemmas 3.1 and 3.2 that this can happen at most $n-1$ times before the algorithm finds a δ-augmenting path or finishes the scaling phase. Hence, $|I|$ is always $\mathrm{O}(n)$, and the algorithm performs Double-Exchange $\mathrm{O}(n^3)$ times and Tail-Exchange $\mathrm{O}(n^2)$ times between δ-augmentations. ∎

Let M be the maximum value of f. Since $f(\emptyset,\emptyset) = 0$, the maximum value M is nonnegative.

THEOREM 4.5. *The scaling algorithm finds a minimizer of f in* $\mathrm{O}(n^5 \log M)$ *time.*

Proof. For the initial $x \in \mathrm{P}(f)$, let $B = \{v \mid x(v) > 0\}$ and $C = \{v \mid x(v) < 0\}$. Then we have $\|x\| = x(B) - x(C) \leq f(B,C) \leq M$. Hence the algorithm performs $\mathrm{O}(\log M)$ scaling phases. It follows from Lemmas 4.3 and 4.4 that each scaling phase performs $\mathrm{O}(n^5)$ function evaluations and arithmetic operations. Therefore the total running time is $\mathrm{O}(n^5 \log M)$. ∎

5. Conclusion.

We have described a combinatorial polynomial algorithm for minimizing integer-valued bisubmodular functions. If we are given a positive lower bound ϵ for the difference between the minimum and the second minimum value of f, a variant of the present algorithm works for any real-valued bisubmodular function f. The only required modification is to change the stopping rule $\delta < 1/3n^2$ to $\delta < \epsilon/3n^2$. The running time is $\mathrm{O}(n^5 \log(M/\epsilon))$. Thus we obtain a polynomial algorithm for testing membership in delta-matroid polyhedra.

One can make this algorithm strongly polynomial with the aid of the generic preprocessing technique of Frank–Tardos [13] that uses the simultaneous Diophantine approximation. However, a more natural strongly polynomial algorithm is desirable. Subsequent to this paper, McCormick and Fujishige [21] have devised such an algorithm for general bisubmodular function minimization.

Acknowledgments. The authors are very grateful to Kazutoshi Ando and an anonymous referee for pointing out errors in earlier versions of this manuscript.

REFERENCES

[1] K. Ando and S. Fujishige, *On structures of bisubmodular polyhedra*, Math. Program., 74 (1996), pp. 293–317.

[2] E. Balas and W. R. Pulleyblank, *The perfectly matchable subgraph polytope of an arbitrary graph*, Combinatorica, 9 (1989), pp. 321–337.

[3] A. Bouchet, *Greedy algorithm and symmetric matroids*, Math. Program., 38 (1987), pp. 147–159.

[4] A. Bouchet, *Matchings and $\triangle$-matroids*, Discrete Appl. Math., 24 (1989), pp. 55–62.

[5] A. Bouchet and W. H. Cunningham, *Delta-matroids, jump systems and bisubmodular polyhedra*, SIAM J. Discrete Math., 8 (1995), pp. 17–32.

[6] R. Chandrasekaran and S. N. Kabadi, *Pseudomatroids*, Discrete Math., 71 (1988), pp. 205–217.

[7] W. H. Cunningham, *Testing membership in matroid polyhedra*, J. Combin. Theory Ser. B, 36 (1984), pp. 161–188.

[8] W. H. Cunningham, *On submodular function minimization*, Combinatorica, 5 (1985), pp. 185–192.

[9] W. H. Cunningham and J. Green-Krótki, *b-matching degree sequence polyhedra*, Combinatorica, 11 (1991), pp. 219–230.

[10] W. H. Cunningham and J. Green-Krótki, *A separation algorithm for the matchable set polytope*, Math. Programming, 65 (1994), pp. 139–150.

[11] A. Dress and T. F. Havel, *Some combinatorial properties of discriminants in metric vector spaces*, Adv. Math., 62 (1986), pp. 285–312.

[12] F. D. J. Dunstan and D. J. A. Welsh, *A greedy algorithm solving a certain class of linear programmes*, Math. Program., 5 (1973), pp. 338–353.

[13] A. Frank and É. Tardos, *An application of simultaneous Diophantine approximation in combinatorial optimization*, Combinatorica, 7 (1987), pp. 49–65.

[14] S. Fujishige, *Submodular Functions and Optimization*, North-Holland, Amsterdam, 1991; 2nd ed., 2005.

[15] S. Fujishige, *A min-max theorem for bisubmodular polyhedra*, SIAM J. Discrete Math., 10 (1997), pp. 294–308.

[16] M. Grötschel, L. Lovász, and A. Schrijver, *The ellipsoid method and its consequences in combinatorial optimization*, Combinatorica, 1 (1981), pp. 169–197.

[17] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, Berlin, 1988.

[18] S. Iwata, L. Fleischer, and S. Fujishige, *A combinatorial strongly polynomial algorithm for minimizing submodular functions*, J. ACM, 48 (2001), pp. 761-777.

[19] L. Lovász, *Submodular functions and convexity*, in Mathematical Programming — The State of the Art, A. Bachem, M. Grötschel and B. Korte, eds., Springer-Verlag, Berlin, 1983, pp. 235–257.

[20] L. Lovász, *The membership problem in jump systems*, J. Combin. Theory Ser. B, 70 (1997), pp. 45–66.

[21] S. T. McCormick and S. Fujishige, *Better algorithms for bisubmodular function minimization*, manuscript, 2003.

[22] L. Qi, *Directed submodularity, ditroids and directed submodular flows*, Math. Program., 42 (1988), pp. 579–599.

[23] A. Schrijver, *A combinatorial algorithm minimizing submodular functions in strongly polynomial time*, J. Combin. Theory Ser. B, 80 (2000), pp. 346–355.

[24] F. Zhang, *A separation algorithm for b-matching degree-sequence polyhedra*, Math. Oper. Res., 28 (2003), pp. 92–102.