A Capacity Scaling Algorithm for M-Convex Submodular Flow

Satoru IWATA, Satoko MORIGUCHI, and Kazuo MUROTA

METR 2003–42 December 2003

A Capacity Scaling Algorithm for M-Convex Submodular Flow

Satoru IWATA [*] Satoko MORIGUCHI [†] Kazuo MUROTA [‡]

Abstract

This paper presents a faster algorithm for the M-convex submodular flow problem, which is a generalization of the minimum-cost flow problem with an M-convex cost function for the flow-boundary, where an M-convex function is a nonlinear nonseparable discrete convex function on integer points. The algorithm extends the capacity scaling approach for the submodular flow problem by Fleischer, Iwata and McCormick (2002) with the aid of a novel technique of changing the potential by solving maximum submodular flow problems.

Key words. discrete optimization; discrete convex function; submodular flow; algorithm

[*]Graduate School of Information Science and Technology, University of Tokyo, Tokyo 113-8656, Japan. E-mail: iwata@mist.i.u-tokyo.ac.jp

[†]Graduate School of Information Science and Engineering, Tokyo Institute of Technology, Tokyo 152-8552, Japan. E-mail: Satoko.Moriguchi@is.titech.ac.jp

[‡]Graduate School of Information Science and Technology, University of Tokyo, Tokyo 113-8656, Japan; PRESTO JST, Saitama, Japan. E-mail: murota@mist.i.u-tokyo.ac.jp

1 Introduction

In recent research towards a unified framework of discrete convex analysis [26], the concept of M-convex functions was proposed by Murota [23, 24] as an extension of that of valuations on matroids invented by Dress and Wenzel [3]. The concept of L-convex functions, which generalize the Lovász extensions of submodular set functions [19], was also proposed by Murota [24]. These two concepts of discrete convexity are conjugate to each other, and a Fenchel-type duality theorem holds for M- and L-convex/concave functions [23, 24, 25]. The Fenchel-type duality theorem, which is a fundamental result in discrete convex analysis, is equivalent to the optimality characterization of the M-convex submodular flow problem. In fact, the original proof in [25] of the Fenchel-type duality theorem is based on a cycle-canceling algorithm that solves the M-convex submodular flow problem. In other words, an M-convex submodular flow algorithm naturally provides a method for finding both optima in the min-max relation of the Fenchel-type duality theorem.

The M-convex submodular flow problem is a minimum-cost flow problem with an additional M-convex cost function for the flow-boundary. This is a common generalization of the submodular flow problem of Edmonds and Giles [4] and the valuated matroid intersection problem of Murota [21, 22]. Applications include an economic problem of finding a competitive equilibrium of an economy with indivisible commodities [27]. In this economic problem, consumers' utility functions are assumed to have gross substitutes property, which corresponds to M-concavity. The conjugacy between M-convexity and L-convexity corresponds to the relationship between commodities and prices. The dual problem to an M-convex submodular flow problem generalizes the convex cost integer dual network flow problem [1], which arises in many applications such as the isotonic regression, the image segmentation [12] and the convex cost closure set [13].

The running time bound of the cycle-canceling algorithm described in [25] is not polynomial but pseudopolynomial. The first polynomial-time algorithm for the M-convex submodular flow problem is the conjugate scaling algorithm of Iwata and Shigeno [18]. This algorithm extends the cost-scaling primal-dual algorithm due to Cunningham and Frank [2], which is the first polynomial-time combinatorial algorithm for the submodular flow problem. The conjugate scaling algorithm, however, calls submodular function minimization [16, 28] repeatedly, which results in its high cost running time. In fact, the conjugate scaling algorithm using the current fastest submodular function minimization algorithm [15] requires $\mathrm{O}(n^8(\log L)^2 \log K)$ evaluations of the M-convex cost function, where n is the number of vertices, L is the maximum absolute value of the capacity, and K is the maximum absolute value of the cost.

On the other hand, the capacity scaling approach of Edmonds and Karp [5] does not admit a straightforward generalization to the submodular flow problem. This is because a naive scaling of a submodular function destroys the submodularity. The first capacity scaling algorithm for the submodular flow problem was obtained by Iwata [14] with a subsequent improvement in time complexity by Fleischer, Iwata and McCormick [6] based on a technique developed in the cut canceling submodular flow algorithm of Iwata, McCormick and Shigeno [17]. These algorithms make use of additional arcs, called relaxation arcs, to retain submodularity under the scaling.

This paper aims at extending the capacity scaling approach to obtain an efficient algorithm for the M-convex submodular flow problem. An earlier attempt in this direction by Moriguchi and Murota [20] led to a capacity-scaling successive-shortest-path algorithm applicable only to a subclass of M-convex functions that are closed under scaling. In this paper, we extend the relaxation arc technique by introducing a new procedure that updates the potential by solving a maximum submodular flow problem. The resulting algorithm provides an optimal solution under the assumption that an oracle for computing M-convex function values is available. The number of oracle calls is bounded by $\mathrm{O}(n^6(\log L)^2\log(nK))$. This is not only the fastest known algorithm but also the first combinatorial polynomial-time algorithm that solves the M-convex submodular flow problem without relying on submodular function minimization.

A natural further question that may arise is whether there exists a strongly polynomial time algorithm for the M-convex submodular flow problem or not. However, this is known to be impossible. It is shown in [11] that there exist no strongly polynomial time algorithms to solve the resource allocation problem with a separable convex cost function. Since this problem is a special case of the M-convex function minimization, it follows that there exist no strongly polynomial time algorithms to solve M-convex function minimization. Furthermore, the M-convex function minimization is a special case of the M-convex submodular flow problem. Therefore, the M-convex submodular flow problem does not admit strongly polynomial time algorithms.

The outline of this paper is as follows. Section 2 describes the M-/L-convex functions and the M-convex submodular flow problem. The successive shortest path algorithm for the M-convex submodular flow problem is described in Section 3. Section 4 presents our capacity scaling algorithm and its time complexity analysis.

2 M-convex Submodular Flow Problem

2.1 M-convex Functions and Base Polyhedra

Let V be a finite set, and $\chi_v \in \{0,1\}^V$ denote the characteristic vector of $v \in V$. The characteristic vector of $X \subseteq V$ is denoted by χ_X. For a vector $x \in \mathbf{Z}^V$ and an element $v \in V$, $x(v)$ means the component of x with index v. For a vector $x \in \mathbf{Z}^V$ and a set $X \subseteq V$, we write $x(X) = \sum_{v \in X} x(v)$. We write the positive and negative supports of a vector x by

$$\mathrm{supp}^+(x) = \{v \in V \mid x(v) > 0\}, \quad \mathrm{supp}^-(x) = \{v \in V \mid x(v) < 0\}.$$

For a function $f : \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$, we use the notation

$$\mathrm{dom}\, f = \{x \in \mathbf{Z}^V \mid f(x) < +\infty\}$$

for the effective domain of f.

A function $f : \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ is said to be M-convex if it satisfies

(M-EXC) $\forall x, y \in \mathrm{dom}\, f$, $\forall u \in \mathrm{supp}^+(x - y)$, $\exists v \in \mathrm{supp}^-(x - y)$ such that

$$f(x) + f(y) \geq f(x - \chi_u + \chi_v) + f(y + \chi_u - \chi_v).$$

It follows from (M-EXC) that $B = \mathrm{dom}\, f$ satisfies the following property:

(B-EXC) $\forall x, y \in B$, $\forall u \in \mathrm{supp}^+(x - y)$, $\exists v \in \mathrm{supp}^-(x - y)$ such that

$$x - \chi_u + \chi_v \in B,\ y + \chi_u - \chi_v \in B.$$

Note that (B-EXC) implies $x(V) = y(V)$ for any $x, y \in B$. A nonempty set $B \subseteq \mathbf{Z}^V$ with the property (B-EXC) is called an M-convex set.

A set function $\rho : 2^V \to \mathbf{Z}$ is said to be submodular if it satisfies

$$\rho(X) + \rho(Y) \geq \rho(X \cup Y) + \rho(X \cap Y) \quad (X, Y \subseteq V).$$

With a submodular function ρ, we associate the base polyhedron $\mathbf{B}(\rho)$ defined by

$$\mathbf{B}(\rho) = \{x \in \mathbf{R}^V \mid x(V) = \rho(V); \forall X \subseteq V : x(X) \leq \rho(X)\}.$$

An M-convex set is essentially the same as the set of integer points of the base polyhedron. More precisely, a bounded set $B \subseteq \mathbf{Z}^V$ is M-convex if and only if $B = \mathbf{B}(\rho) \cap \mathbf{Z}^V$ for some submodular function $\rho : 2^V \to \mathbf{Z}$. Accordingly, a member of an M-convex set is referred to as a base.

For any base $x \in B$ and $u, v \in V$ with $u \neq v$, the integer defined by

$$\sigma(x, v, u) = \max\{\alpha \in \mathbf{Z} \mid x - \alpha(\chi_u - \chi_v) \in B\}$$

is called the exchange capacity. The exchangeability graph for a base $x \in B$ is a directed graph with the vertex set V and the arc set $E_x = \{(u, v) \mid \sigma(x, v, u) > 0\}$.

Lemma 2.1 (no-short cut lemma, cf. [9, Lemma 4.5]). *Suppose that B is an M-convex set, $x \in B$ and that $u_1, v_1, u_2, v_2, \ldots, u_k, v_k$ are distinct elements of V. If $x - \chi_{u_i} + \chi_{v_i} \in B$ for $i = 1, \ldots, k$ and $x - \chi_{u_i} + \chi_{v_j} \notin B$ for any $i < j$, then $y = x - \sum_{i=1}^{k}(\chi_{u_i} - \chi_{v_i}) \in B$.* □

The following local characterization of global minimality is fundamental. We denote by $\arg\min f$ the set of the global minimizers of f.

Theorem 2.2 ([23], cf. [26]). *Let $f : \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ be an M-convex function and $x_* \in \operatorname{dom} f$. Then $x_* \in \arg\min f$ if and only if $f(x_*) \leq f(x_* - \chi_u + \chi_v)$ for all $u, v \in V$.* □

For a function f and a vector $p = (p(v) \mid v \in V)$, we denote by $f[-p]$ the function defined by

$$f[-p](x) = f(x) - \langle p, x\rangle \qquad (x \in \mathbf{Z}^V),$$

where $\langle p, x\rangle = \sum_{v\in V} p(v)x(v)$. This is M-convex for M-convex f. The minimizers of an M-convex function form an M-convex set. An immediate consequence of this and the M-convexity of $f[-p]$ is the following lemma, which is used in the design of our algorithm in Section 4.

Lemma 2.3. *Let $f : \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ be an M-convex function. Then* $\arg\min f[-p]$ *is an M-convex set for each $p \in \mathbf{Z}^V$.* □

For a function $f : \mathbf{Z}^V \to \mathbf{Z}\cup\{+\infty\}$, the discrete Legendre-Fenchel transform is defined by

$$g(p) = \sup\{\langle p, x\rangle - f(x) \mid x \in \mathbf{Z}^V\} \qquad (p \in \mathbf{Z}^V),$$

which is also called the conjugate of f. The conjugate function $g : \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ of an M-convex function f is another kind of discrete convex function, called an L-convex function, and its conjugate coincides with f.

For vectors $p, q \in \mathbf{Z}^V$, we write $p \vee q$ and $p \wedge p$ for their componentwise maximum and minimum. A function $g : \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ is called L-convex if it satisfies

(SBF) $g(p) + g(q) \geq g(p \vee q) + g(p \wedge p) \quad (p, q \in \mathbf{Z}^V)$,
(TRF) $\exists r \in \mathbf{Z}$ such that $g(p + \mathbf{1}) = g(p) + r \quad (p \in \mathbf{Z}^V)$,

where $\mathbf{1} = (1, 1, \ldots, 1) \in \mathbf{Z}^V$.

Global optimality for L-convex function minimization is characterized by local optimality.

Theorem 2.4 ([26]). *For an L-convex function g and $p \in \operatorname{dom} g$, we have*

$$g(p) \leq g(q) \quad (q \in \mathbf{Z}^V) \quad \Longleftrightarrow \quad \begin{cases} g(p) \leq g(p + \chi_X) & (X \subseteq V), \\ g(p) = g(p + \mathbf{1}). \end{cases}$$

□

Our algorithm presented in Section 4 relies on the following property of an L-convex function.

Lemma 2.5 ([26]). *An L-convex function $g : \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ satisfies the discrete midpoint convexity*

$$g(p) + g(q) \geq g\left(\left\lceil \frac{p+q}{2} \right\rceil\right) + g\left(\left\lfloor \frac{p+q}{2} \right\rfloor\right) \quad (p, q \in \mathbf{Z}^V),$$

where $\lceil\cdot\rceil$ and $\lfloor\cdot\rfloor$ mean the rounding-up and -down of a vector to the nearest integer vector. □

2.2 M-convex Submodular Flow Problem

The submodular flow problem is a combinatorial optimization problem that has good properties such as availability of optimality criteria in terms of potentials (dual variables) and negative cycles, integrality of primal and dual optimal solutions, and solvability by efficient algorithms. The submodular flow problem can be generalized to the M-convex submodular flow problem which also retains the aforementioned good properties. In this paper we consider integer-flow problems with integer costs.

Let $G = (V, A)$ be a directed graph with vertex set V of cardinality n and arc set A of cardinality m. Assume that G does not contain parallel arcs, and hence $m \leq n^2$. Consider lower and upper capacity bounds $b, c : A \to \mathbf{Z}$ and the cost function $d : A \to \mathbf{Z}$. For each vertex $v \in V$, let $\delta^+ v$ (resp., $\delta^- v$) denote the set of arcs leaving (resp., entering) v. For each vertex subset $X \subseteq V$, let $\Delta^+ X$ (resp., $\Delta^- X$) denote the set of arcs leaving (resp., entering) X. For each arc $a \in A$, $\partial^+ a$ designates the initial vertex of a, and $\partial^- a$ the terminal vertex of a. The boundary $\partial\xi$ of flow $\xi : A \to \mathbf{Z}$ is defined by

$$\partial\xi(v) = \sum\{\xi(a) \mid a \in \delta^+ v\} - \sum\{\xi(a) \mid a \in \delta^- v\} \qquad (v \in V),$$

which represents the net flow leaving vertex v.

Suppose that $B \subseteq \mathbf{Z}^V$ is an M-convex set. The submodular flow problem can be formulated as follows.

Submodular flow problem SFP

$$\begin{aligned}
\text{Minimize} \quad & \sum_{a \in A} d(a)\xi(a) && (2.1)\\
\text{subject to} \quad & b(a) \leq \xi(a) \leq c(a) \quad (a \in A), && (2.2)\\
& \partial\xi \in B, && (2.3)\\
& \xi(a) \in \mathbf{Z} \quad (a \in A). && (2.4)
\end{aligned}$$

A feasible flow for the submodular flow problem SFP means a function $\xi : A \to \mathbf{Z}$ that satisfies (2.2) and (2.3). The problem SFP is said to be feasible if it admits a feasible flow.

Theorem 2.6 ([7], cf. [9, Theorem 5.1]). *A submodular flow problem* SFP *for a bounded M-convex set B is feasible if and only if*

$$b(\Delta^+ X) - c(\Delta^- X) \leq \rho(X) \qquad (X \subseteq V),$$

where ρ is the submodular set function that determines B by $B = \mathbf{B}(\rho) \cap \mathbf{Z}^V$. □

The feasibility of an SFP can be checked efficiently [7, 10, 31], provided that an oracle for exchange capacity is available.

The M-convex submodular flow problem is a generalization of the submodular flow problem obtained by introducing a cost function for the flow boundary $\partial\xi$ rather than merely imposing the feasibility constraint $\partial\xi \in B$. More precisely, with an M-convex function $f : \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ we add a new term $f(\partial\xi)$ to the objective function.

M-convex submodular flow problem MCSFP

$$\begin{aligned}
\text{Minimize} \quad & \sum_{a\in A} d(a)\xi(a) + f(\partial\xi) & (2.5)\\
\text{subject to} \quad & b(a) \le \xi(a) \le c(a) \quad (a \in A), & (2.6)\\
& \partial\xi \in \operatorname{dom} f, & (2.7)\\
& \xi(a) \in \mathbf{Z} \quad (a \in A). & (2.8)
\end{aligned}$$

Note that the M-convex submodular flow problem with a $\{0, +\infty\}$-valued f reduces to the submodular flow problem SFP.

A feasible flow for the M-convex submodular flow problem MCSFP means a function $\xi : A \to \mathbf{Z}$ that satisfies (2.6) and (2.7). The problem MCSFP is called feasible if it admits a feasible flow. Since $\operatorname{dom} f$ is an M-convex set, the feasibility of an MCSFP is reduced to the feasibility of the corresponding SFP. The exchange capacity in $\operatorname{dom} f$ can be computed efficiently by repeated applications of the function evaluation oracle. Thus the feasibility of an MCSFP can be checked efficiently, provided that an initial base $x \in \operatorname{dom} f$ is available.

In the rest of this paper, we assume that the given MCSFP is feasible. This assumption implies that

$$\operatorname{dom} f \subseteq \{x \in \mathbf{Z}^V \mid x(V) = 0\}. \qquad (2.9)$$

For the sake of simplicity, we further assume that $\operatorname{dom} f$ is bounded, which implies the existence of a submodular function $\rho : 2^V \to \mathbf{Z}$ such that $\operatorname{dom} f = \mathbf{B}(\rho) \cap \mathbf{Z}^V$.

The following optimality condition is known for the M-convex submodular flow problem MCSFP. By a potential we mean a vector $p \in \mathbf{Z}^V$. The reduced cost $d_p : A \to \mathbf{Z}$ is defined by $d_p(a) = d(a) + p(\partial^+ a) - p(\partial^- a)$ for $a \in A$.

Theorem 2.7 (potential criterion, [25], cf. [26]). *A feasible flow ξ is optimal if and only if there exists a potential $p \in \mathbf{Z}^V$ and a base $x \in \operatorname{dom} f$ such that*

(A) $d_p(a) > 0 \Longrightarrow \xi(a) = b(a)$,

$d_p(a) < 0 \Longrightarrow \xi(a) = c(a)$,

(B) $x \in \arg\min f[-p]$,

(C) $x = \partial\xi$. □

An alternative representation of $\arg\min f[-p]$ in condition (B) above, which is M-convex by Lemma 2.3, is obtained as follows. Let g be the conjugate of the M-convex function f. Since $\operatorname{dom} f$ is bounded, g is finite-valued. Furthermore, (2.9) implies that $g(p + \mathbf{1}) = g(p)$ for any $p \in \mathbf{Z}^V$.

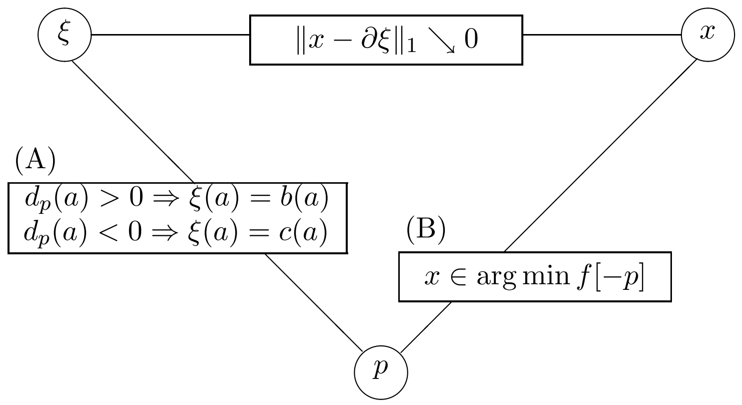
ξ
$\|x - \partial\xi\|_1 \searrow 0$
x
(A)
$d_p(a) > 0 \Rightarrow \xi(a) = b(a)$
$d_p(a) < 0 \Rightarrow \xi(a) = c(a)$
(B)
$x \in \arg\min f[-p]$
p

Figure 1: Relationship among variables in the successive shortest path algorithm

Lemma 2.8. *Let $g_p : 2^V \to \mathbf{Z}$ be the submodular set function defined by*

$$g_p(X) = g(p + \chi_X) - g(p) \quad (X \subseteq V).$$

Then, we have

$$\arg\min f[-p] = \mathbf{B}(g_p) \cap \mathbf{Z}^V.$$

Proof. By the definition of the conjugate, we have $f(x) + g(p) \geq \langle p, x \rangle$ for any $x \in \mathbf{Z}^V$ and any $p \in \mathbf{Z}^V$. It also implies that

$$x \in \arg\min f[-p] \iff f(x) + g(p) = \langle p, x \rangle \iff p \in \arg\min g[-x].$$

On the other hand, it follows from the optimality criterion for L-convex functions (Theorem 2.4) that

$$p \in \arg\min g[-x] \quad \iff \quad \begin{cases} g(p + \chi_X) - g(p) \geq x(X) & (X \subseteq V), \\ g(p + \mathbf{1}) - g(p) = x(V). \end{cases}$$

Combining the above shows that $\arg\min f[-p]$ coincides with the M-convex set $\mathbf{B}(g_p) \cap \mathbf{Z}^V$. □

3 Successive Shortest Path Algorithm

This section describes the successive shortest path (SSP) algorithm for the M-convex submodular flow problem MCSFP, presented by Moriguchi-Murota [20]. As with the successive shortest path algorithm for minimum cost flows, this algorithm is not polynomial, but only pseudopolynomial, since the number of iterations is linear in

$$L = \max\{\max_{a \in A} |b(a)|, \ \max_{a \in A} |c(a)|, \ \max_{x,y \in \mathrm{dom}\, f} \|x - y\|_\infty\}.$$

In Section 4, we shall improve upon the SSP algorithm with the use of a capacity-scaling technique to obtain a polynomial-time algorithm. We shall also use the SSP at the final stage of our capacity-scaling algorithm.

The SSP algorithm is based on the optimality criterion in Theorem 2.7 which guarantees the existence of a potential satisfying the three conditions, (A), (B), and (C). The algorithm maintains a flow $\xi : A \to \mathbf{Z}$, a base $x \in \mathrm{dom}\, f$, and a potential $p \in \mathbf{Z}^V$ and works with an auxiliary graph $G(\xi, x)$. For these variables, the algorithm always maintains the conditions (A) and (B). In general stages of the algorithm, the flow boundary $\partial\xi$ differs from the base x, but their discrepancy is decreased so that the condition (C) holds at the termination of the algorithm. See Figure 1, where the consistency among the variables is illustrated.

The auxiliary graph $G(\xi, x)$ is defined as follows. The vertex set is V and the arc set consists of two disjoint parts, A_ξ and E_x, i.e., $G(\xi, x) = (V, A_\xi \cup E_x)$. The set A_ξ is given by $A_\xi = F_\xi \cup B_\xi$ with

$$\begin{aligned} F_\xi &= \{a \mid a \in A, \xi(a) < c(a)\}, \\ B_\xi &= \{\overline{a} \mid a \in A, b(a) < \xi(a)\} \quad (\overline{a}\text{: reorientation of } a), \end{aligned}$$

whereas E_x is the arc set of the exchangeability graph for $x \in \mathrm{dom}\, f$, i.e.,

$$E_x = \{(u, v) \mid x - \chi_u + \chi_v \in \mathrm{dom}\, f\}.$$

Note that A_ξ represents arcs in the residual graph for a flow ξ employed in the ordinary minimum cost flow problem. Furthermore, each arc a of the auxiliary graph has a length $l(a)$ defined by

$$l(a) = \begin{cases} d(a) & (a \in F_\xi), \\ -d(\overline{a}) & (a \in B_\xi, \overline{a} \in A), \\ \Delta f(x; v, u) & (a = (u, v) \in E_x), \end{cases}$$

where

$$\Delta f(x; v, u) = f(x - \chi_u + \chi_v) - f(x).$$

Given a potential $p \in \mathbf{Z}^V$, we define the reduced length $l_p : A_\xi \cup E_x \to \mathbf{Z}$ with respect to p as

$$l_p(a) = l(a) + p(\partial^+ a) - p(\partial^- a) \quad (a \in A_\xi \cup E_x).$$

In terms of this reduced length, our conditions (A) and (B) can be put in a more compact form:

$$l_p(a) \geq 0 \quad (a \in A_\xi \cup E_x). \tag{3.1}$$

Indeed, it is easy to see that the condition (A) is equivalent to $l_p(a) \geq 0$ for all $a \in A_\xi$. The condition (B) is equivalent to $l_p(a) \geq 0$ for all $a \in E_x$, since

$$x \in \arg\min f[-p] \iff \Delta f(x; v, u) + p(u) - p(v) \geq 0 \quad (u, v \in V)$$

by Theorem 2.2. Thus, the algorithm always maintains the nonnegativity of the reduced length (3.1) in the auxiliary graph $G(\xi, x)$.

The optimality of the flow constructed by the algorithm is guaranteed by the following lemma, a reformulation of the sufficiency part of the optimality criterion given in Theorem 2.7.

Lemma 3.1. *Suppose that* (3.1) *is satisfied by a flow* ξ*, a base* x*, and a potential* p*. If* $x = \partial\xi$*, then* ξ *is optimal to* MCSFP. □

The algorithm reduces the discrepancy between x and $\partial\xi$ by augmenting a unit flow along a path from a vertex in $S^+ = \{v \in V \mid x(v) > \partial\xi(v)\}$ to a vertex in $S^- = \{v \in V \mid x(v) < \partial\xi(v)\}$. Let P be a shortest path from S^+ to S^-, with a minimum number of arcs, with respect to the reduced length l_p. Since l_p is nonnegative, such a shortest path can be found by Dijkstra's algorithm. If there exists no path from $S^+ \neq \emptyset$ to $S^- \neq \emptyset$, then the problem is infeasible. This is because, for the set X of the vertices unreachable from S^+, we have $b(\Delta^+X) - c(\Delta^-X) = \xi(\Delta^+X) - \xi(\Delta^-X) = \partial\xi(X) > x(X) = \rho(X)$, where ρ is the submodular set function that determines $\mathrm{dom}\, f$ by $\mathrm{dom}\, f = \mathbf{B}(\rho) \cap \mathbf{Z}^V$. By Theorem 2.6, this implies that the problem is infeasible. Because of our feasibility assumption, we may assert that there exists a path from S^+ to S^- unless they are empty.

The flow augmentation along P means the operation that updates (ξ, x) as follows.

- If $a \in F_\xi \cap P$, then $\xi(a) := \xi(a) + 1$.
- If $a \in B_\xi \cap P$, then $\xi(\overline{a}) := \xi(\overline{a}) - 1$.
- If $a \in E_x \cap P$, then $x(\partial^+ a) := x(\partial^+ a) - 1,\ x(\partial^- a) := x(\partial^- a) + 1$.

This flow augmentation decreases the discrepancy $\|x - \partial\xi\|_1$. Obviously, ξ remains to satisfy (2.6). Also x remains to be a base, i.e., a member of $\mathrm{dom}\, f$, because of the no-short cut lemma (Lemma 2.1); see [20] for details. Let $q(v)$ denote the shortest path distance from S^+ to each $v \in V$. To restore the condition (3.1), we also modify the potential

$$p(v) := p(v) + \min\{q(v), \sum_{a \in P} l_p(a)\} \qquad (v \in V).$$

Given an initial triple of a base x, a flow $\xi : A \to \mathbf{Z}$ and a potential $p \in \mathbf{Z}^V$ satisfying the capacity constraint (2.6) and the nonnegativity (3.1) of $l_p(a)$ for all $a \in A_\xi \cup E_x$, the algorithm SSP, described in Figure 2, repeats the flow augmentation until S^+ becomes empty. Then S^- is also empty, and the condition (C) holds, which means ξ is optimal to MCSFP by Theorem 2.7.

We now analyze the running time of algorithm SSP assuming that the value of f can be evaluated within time F. The total number of the flow augmentation is bounded by $\|\partial\xi - x\|_1$ for the initial ξ and x, which, in turn, is bounded by $\mathrm{O}(n^2 L)$. The construction of $G(\xi, x)$ takes $\mathrm{O}(F \cdot n^2)$ time since the length of an arc of E_x can be computed with an evaluation of f. The Dijkstra method takes $\mathrm{O}(n^2)$ time. Thus the total time complexity of algorithm SSP is $\mathrm{O}(F \cdot n^4 L)$.

```
Algorithm: SSP($x, \xi, p$)
Input: $b, c, d, f$ for MCSFP and $x, \xi, p$ satisfying (2.6) and (3.1).
Output: An optimal flow $\xi$ for MCSFP.

begin
    while $S^+ \neq \emptyset$ do
    begin
        compute the shortest path distance $q(v)$ from $S^+$ to each $v \in V \setminus S^+$
        in $G(\xi, x)$ with respect to the reduced length $l_p$;
        let $P$ be the one with the minimum number of arcs among the
        shortest paths from $S^+$ to $S^-$;
        for $v \in V$ do $p(v) := p(v) + \min\{q(v), \sum_{a \in P} l_p(a)\}$;
        for $a \in F_\xi \cap P$ do $\xi(a) := \xi(a) + 1$;
        for $a \in B_\xi \cap P$ do $\xi(\overline{a}) := \xi(\overline{a}) - 1$;
        for $a \in E_x \cap P$ do $x(\partial^+ a) := x(\partial^+ a) - 1, x(\partial^- a) := x(\partial^- a) + 1$
    end;
    return $\xi$
end
```

Figure 2: Successive shortest path algorithm

4 A Capacity Scaling Algorithm

4.1 Algorithm Outline

In this section, we present a capacity scaling algorithm for the M-convex submodular flow problem as an improvement of the SSP algorithm. The running time bound of the new algorithm is polynomial in n, $\log L$, and $\log K$,where

$$K = \max\{\max_{a \in A} |d(a)|, \max_{x,y \in \mathrm{dom}\, f} |f(x) - f(y)|\}.$$

Note that K and L are used as measures of the problem size. We do not need to evaluate their values in our algorithm.

The algorithm performs a number of scaling phases for different values of a scaling parameter α that represents a scale for the arc capacity and $\mathrm{dom}\, f$. A scaling phase with a specific value of α is called the α-scaling phase. The scaling parameter α is initially set to a sufficiently large number and reduced by a factor of two in each scaling phase. At the end of the α-scaling phase, the algorithm obtains a flow and a potential that satisfy the optimality criterion in Theorem 2.7 only approximately, with precision α, so to speak. We refer to this condition as α-optimality, of which we give the precise definition later. At the final stage, after the execution of the 1-scaling phase, the algorithm calls algorithm SSP to end up with an optimal flow.

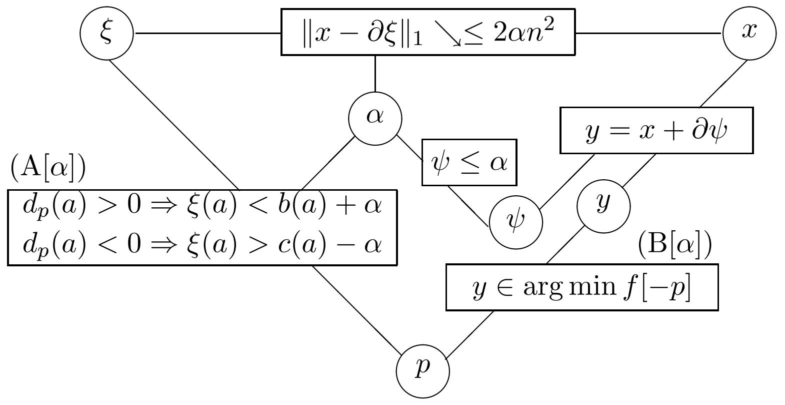
ξ
$\|x - \partial\xi\|_1 \searrow \leq 2\alpha n^2$
x
α
$y = x + \partial\psi$
$\psi \leq \alpha$
(A[α])
$d_p(a) > 0 \Rightarrow \xi(a) < b(a) + \alpha$
$d_p(a) < 0 \Rightarrow \xi(a) > c(a) - \alpha$
ψ
y
(B[α])
$y \in \arg\min f[-p]$
p

Figure 3: Relationship among variables in a scaling phase

Just as the SSP algorithm, the capacity scaling algorithm maintains a flow $\xi : A \to \mathbf{Z}$, a vector $x \in \mathbf{Z}^V$ and a potential $p \in \mathbf{Z}^V$ and works with an auxiliary graph. The capacity constraint (2.6) for the flow ξ is always retained. The flow boundary $\partial\xi$ may differ from the vector x, but their discrepancy is controlled by the scaling parameter α so that $\|x - \partial\xi\|_1$ becomes sufficiently small, i.e., at most $2\alpha n^2$, at the end of the α-scaling phase.

To incorporate the scaling approach into the SSP framework, our algorithm adds a complete directed graph on V with arc set $D = \{(u, v) \mid u \neq v \in V\}$ to the given graph (V, A), and considers a flow $\psi : D \to \mathbf{Z}$ on it. The added arcs are called the relaxation arcs and the flow ψ is called the relaxation flow. The relaxation arcs have capacity α, so that

$$0 \leq \psi(a) \leq \alpha \quad (a \in D). \tag{4.1}$$

Accordingly, the capacity functions b and c are extended to D as $b(a) = 0$ and $c(a) = \alpha$ for all $a \in D$. The algorithm always retains this capacity constraints for ψ. We impose that, for any distinct $u, v \in V$, at least one of $\psi(u, v)$ and $\psi(v, u)$ should be zero. Thus the algorithm maintains a pair of flows ξ on A and ψ on D. In accordance with the introduction of the relaxation flow ψ, the algorithm uses another vector y to represent $x + \partial\psi$, i.e., it always keeps the relationship

$$y = x + \partial\psi. \tag{4.2}$$

In the algorithm, we always keep y as a base in $\mathrm{dom}\, f$.

We now define α-optimality, which is a relaxation of the optimality criterion in Theorem 2.7. A triple (ξ, x, ψ) is said to be α-optimal if it satisfies

(A[α]**)** $d_p(a) > 0 \Longrightarrow b(a) \leq \xi(a) < b(a) + \alpha,$

$d_p(a) < 0 \Longrightarrow c(a) \geq \xi(a) > c(a) - \alpha,$

(B[α]**)** $y = x + \partial\psi \in \arg\min f[-p],$

(C[α]) $\|x - \partial\xi\|_1 \leq 2\alpha n^2$.

Each scaling phase of our algorithm aims at finding an α-optimal (ξ, x, ψ). Once the algorithm obtains an α-optimal (ξ, x, ψ), it cuts the value of α in half and goes to the next phase. The relationship among the variables in each α-scaling phase is summarized in Figure 3.

The auxiliary graph is defined with reference to a pair of flows, ξ and ψ, and a base y (not x), as follows. The vertex set is V and the arc set consists of three disjoint parts, A_ξ, E_y, and D_ψ; i.e., $G(\xi, y, \psi) = (V, A_\xi \cup E_y \cup D_\psi)$. The set A_ξ is given by $A_\xi = F_\xi \cup B_\xi$ with

$$\begin{aligned} F_\xi &= \{a \mid a \in A, \xi(a) < c(a)\}, \\ B_\xi &= \{\overline{a} \mid a \in A, b(a) < \xi(a)\} \quad (\overline{a}\text{: reorientation of } a), \end{aligned}$$

whereas D_ψ is given by

$$D_\psi = \{a \mid a \in D, \psi(a) < \alpha\},$$

and E_y is the arc set of the exchangeability graph for $y \in \operatorname{dom} f$.

On the arc set $A_\xi \cup D_\psi$, we define a function $r : A_\xi \cup D_\psi \to \mathbf{Z}$, representing residual capacities, as

$$r(a) = \begin{cases} c(a) - \xi(a) & (a \in F_\xi), \\ \xi(\overline{a}) - b(\overline{a}) & (a \in B_\xi, \overline{a} \in A), \\ \alpha - \psi(a) & (a \in D_\psi). \end{cases}$$

The arc sets F_ξ, B_ξ, A_ξ and D_ψ that consist of arcs with residual capacities at least α are denoted by $F_\xi(\alpha), B_\xi(\alpha), A_\xi(\alpha)$ and $D_\psi(\alpha)$, respectively, i.e.,

$$\begin{aligned} F_\xi(\alpha) &= \{a \mid a \in F_\xi, r(a) \geq \alpha\}, \\ B_\xi(\alpha) &= \{a \mid a \in B_\xi, r(a) \geq \alpha\}, \\ A_\xi(\alpha) &= \{a \mid a \in A_\xi, r(a) \geq \alpha\} = F_\xi(\alpha) \cup B_\xi(\alpha), \\ D_\psi(\alpha) &= \{a \mid a \in D_\psi, \psi(a) = 0\}. \end{aligned}$$

Furthermore, on the arc set $A_\xi \cup E_y$, we define a function $l : A_\xi \cup E_y \to \mathbf{Z}$, representing arc lengths, as

$$l(a) = \begin{cases} d(a) & (a \in F_\xi), \\ -d(\overline{a}) & (a \in B_\xi, \overline{a} \in A), \\ \Delta f(y; v, u) & (a = (u, v) \in E_y). \end{cases}$$

Given a potential $p \in \mathbf{Z}^V$, we define the reduced length $l_p : A_\xi \cup E_y \to \mathbf{Z}$ with respect to p as

$$l_p(a) = l(a) + p(\partial^+ a) - p(\partial^- a) \quad (a \in A_\xi \cup E_y).$$

In terms of this reduced length, our conditions (A[α]) and (B[α]) can be put in a more compact form:

$$l_p(a) \geq 0 \quad (a \in A_\xi(\alpha) \cup E_y). \tag{4.3}$$

The algorithm always maintains this nonnegativity of the reduced length for the arcs in $A_\xi(\alpha) \cup E_y$.

We now intend to reduce the discrepancy between x and $\partial\xi$ by augmenting α units of flow along a path P from

$$S^+(\alpha) = \{v \in V \mid x(v) - \partial\xi(v) \geq \alpha\}$$

to

$$S^-(\alpha) = \{v \in V \mid x(v) - \partial\xi(v) \leq -\alpha\},$$

while maintaining (A[α]) and (B[α]). Such a path that allows α-units of flow augmentation should consist of arcs in $A_\xi(\alpha) \cup D_\psi(\alpha)$. In order to satisfy (A[α]) after the augmentation, we further impose that $l_p(a) = 0$ on arcs $a \in A_\xi(\alpha) \cap P$. That is, the path P consists of arcs in $A^* \cup D^*$, where $A^* = F^* \cup B^*$ with

$$F^* = \{a \mid a \in F_\xi(\alpha),\ l_p(a) = 0\}, \quad B^* = \{a \mid a \in B_\xi(\alpha),\ l_p(a) = 0\},$$

and $D^* = D_\psi(\alpha)$. We call such a path an α-augmenting path. In Section 4.2, we describe a path search procedure that finds an α-augmenting path by possibly changing ψ, y and p while maintaining (B[α]).

We augment the flow along an α-augmenting path P. The flow augmentation along P means that we update $\xi(a)$ and $\psi(a)$ for each arc $a \in P$ as follows.

- If $a \in F^*$, then $\xi(a) := \xi(a) + \alpha$.
- If $a \in B^*$, then $\xi(\overline{a}) := \xi(\overline{a}) - \alpha$.
- If $a \in D^*$, then $\psi(a) := \alpha - \psi(\overline{a})$, $\psi(\overline{a}) := 0$.

We also update x to keep $y = x + \partial\psi$ unchanged. This flow augmentation decreases the discrepancy between y and $\partial\xi + \partial\psi$ at both endpoints of the path P. The discrepancy is unchanged at the other vertices. Since $x - \partial\xi = y - (\partial\xi + \partial\psi)$, the discrepancy $\|x - \partial\xi\|_1$ is reduced by 2α. The flow augmentation does not violate (B[α]) as y and p are unchanged. The condition (A[α]) is also preserved since $l_p(a) = 0$ for arcs $a \in A_\xi(\alpha) \cap P$ and an arc with residual capacity newly at least α is the reverse arc of an arc $a \in P$.

The algorithm repeats the path search and flow augmentation until we obtain an α-optimal (ξ, x, ψ). Then we go to the next scaling phase by cutting the value of α in half.

At the start of a new α-scaling phase, we modify ψ as

$$\psi(a) := \min\{\psi(a), \alpha\} \quad (a \in D)$$

to satisfy the capacity constraints (4.1) for the new value of α, and modify x to $y - \partial\psi$ to maintain (4.2). We also modify ξ as follows to resolve the possible violation of (A[α]).

- If $a \in F_\xi(\alpha)$ and $l_p(a) < 0$, then $\xi(a) := \xi(a) + \alpha$.
- If $a \in B_\xi(\alpha)$ and $l_p(a) < 0$, then $\xi(\overline{a}) := \xi(\overline{a}) - \alpha$.

The algorithm also updates the auxiliary graph $G(\xi, y, \psi)$ after this adjustment.

As an initial flow, we adopt any ξ that satisfies (2.6) and (A) in Theorem 2.7 for $p = \mathbf{0}$. We also adopt any $x \in \arg\min f$ as an initial base. For these ξ and x, the initial value of α is set to be $2^{\lfloor \log M \rfloor}$, where $M = \|x - \partial\xi\|_\infty / n$. Since we assume MCSFP is feasible, we have a feasible flow ξ°, which satisfies $\|x - \partial\xi^\circ\|_\infty \leq L$ and $\|\partial\xi^\circ - \partial\xi\|_\infty \leq 2(n-1)L$. This implies $M = \|x - \partial\xi\|_\infty / n \leq 2L$. We initialize $p = \mathbf{0}, \psi = \mathbf{0}$ and $y = x$.

The algorithm Capacity_Scaling is described in Figure 4.

4.2 Path Search

In each phase of Capacity_Scaling, we find an α-augmenting path by calling procedure Path_Search of Figure 5 repeatedly.

At the start of procedure Path_Search, we call procedure Potential_Update to find a potential p which satisfies

$$|p(u) - p(v)| \leq (n-1)K \qquad (u, v \in V) \tag{4.4}$$

in addition to (A[α]) and (B[α]) for the current ξ and y. Let $G_\alpha(\xi, y)$ be a graph with the vertex set V and the arc set $A_\xi(\alpha) \cup E_y$, i.e., $G_\alpha(\xi, y) = (V, A_\xi(\alpha) \cup E_y)$. At the start of procedure Potential_Update, we identify the strongly connected components of $G_\alpha(\xi, y)$ and its source components, i.e., components without entering arcs. We choose a vertex v_C from each source component C. Let S be the collection of v_C for all source components. We compute the shortest path distance $q(v)$ from S to each $v \in V$ in $G_\alpha(\xi, y)$ with respect to the arc length l. We adopt this q as the new potential p. We summarize the procedure Potential_Update in Figure 6.

Procedure Path_Search is a variant of the ordinary graph search on the graph $(V, A^* \cup D^*)$. Let R denote the set of the vertices reached from $S^+(\alpha)$ at a certain point in the search, and put $W = V \setminus R$. If there exists an arc in $A^* \cup D^*$ from R to W, procedure Path_Search extends the search tree in the usual manner. Otherwise, in order to create a new arc that can be used to extend the search tree while maintaining both conditions (A[α]) and (B[α]), it modifies p, y and ψ to $p' = p + \pi\chi_W$, $y' = y - \partial\varphi$, and $\psi' = \psi - \varphi$ with a nonnegative integer π and an integer-flow φ on

$$\vec{D} = \{(u, v) \mid (u, v) \in D, u \in R, v \in W\}$$

determined by procedure Cut_Cancel.

We now discuss how to determine such an appropriate pair of π and φ that yields a new arc in $A^* \cup D^*$. We impose the condition $0 \leq \varphi(a) \leq \psi(a)$ for $a \in \vec{D}$ to maintain (4.1) for ψ'. We also impose $y - \partial\varphi \in \arg\min f[-p - \pi\chi_W]$ to maintain (B[α]) for (y', p'). For the remaining condition (A[α]), we impose a further condition $\pi \leq l_p(a)$ for each arc a in

$$\vec{A}_\xi(\alpha) = \{a \mid a \in A_\xi(\alpha), \partial^+ a \in R, \partial^- a \in W\}.$$

```
**Algorithm:** CAPACITY_SCALING
**Input:** $b, c, d, f$ for MCSFP.
**Output:** An optimal flow $\xi$ for MCSFP.

begin
    set $p := \mathbf{0}$;
    find $\xi$ which satisfies (A) and (2.6);
    find $x \in \arg\min f$;
    set $\psi := \mathbf{0}$ and $y := x$;
    set $M := \|x - \partial\xi\|_\infty / n$ and $\alpha := 2^{\lfloor \log M \rfloor}$;
    while $\alpha \geq 1$ do
    begin
        for $a \in D$ do $\psi(a) := \min\{\psi(a), \alpha\}$;
        $x := y - \partial\psi$;
        for $a \in F_\xi(\alpha)$ do if $l_p(a) < 0$ then $\xi(a) := \xi(a) + \alpha$;
        for $a \in B_\xi(\alpha)$ do if $l_p(a) < 0$ then $\xi(\overline{a}) := \xi(\overline{a}) - \alpha$;
        while $\|x - \partial\xi\|_1 > 2\alpha n^2$ do
        begin
            $P :=$ PATH_SEARCH$(\xi, y, \psi, p, \alpha)$;
            [This updates $\psi, y, p$.]
            [$P$ : path from $S^+(\alpha)$ to $S^-(\alpha)$ on $(V, A^* \cup D^*)$.]
            for $a \in F^* \cap P$ do $\xi(a) := \xi(a) + \alpha$;
            for $a \in B^* \cap P$ do $\xi(\overline{a}) := \xi(\overline{a}) - \alpha$;
            for $a \in D^* \cap P$ do $\psi(a) := \alpha - \psi(\overline{a})$, $\psi(\overline{a}) := 0$;
            $x := y - \partial\psi$
        end;
        $\alpha := \alpha/2$
    end;
    $\xi :=$ SSP$(y, \xi, p)$;
    return $\xi$
end
```

Figure 4: Capacity scaling algorithm

```
Procedure: PATH_SEARCH($\xi, y, \psi, p, \alpha$)
Output: An $\alpha$-augmenting path $P$ from $S^+(\alpha)$ to $S^-(\alpha)$.

begin
    $p :=$ POTENTIAL_UPDATE($\xi, y, \alpha$);
    $Q := S^+(\alpha)$, $R := S^+(\alpha)$, $T := \emptyset$;
    while $R \cap S^-(\alpha) = \emptyset$ do
    begin
        if $Q \neq \emptyset$ then
        begin
            choose $u \in Q$;
            if there exists $v \in V \setminus R$ with $a = (u, v) \in A^* \cup D^*$ then
                $R := R \cup \{v\}$, $Q := Q \cup \{v\}$, $T := T \cup \{a\}$
            else $Q := Q \setminus \{u\}$
        end
        else
        begin
            $(\pi, \varphi) :=$ CUT_CANCEL($\xi, y, \psi, p, \alpha$);
            for $v \in V \setminus R$ do $p(v) := p(v) + \pi$;
            for $v \in V$ do $y(v) := y(v) - \partial\varphi(v)$;
            for $a \in \vec{D}$ do $\psi(a) := \psi(a) - \varphi(a)$;
            $Q := R$
        end
    end;
    return a path $P$ from $S^+(\alpha)$ to $S^-(\alpha)$ using vertices in $R$ and arcs in $T$
end
```

Figure 5: Procedure PATH_SEARCH

```
Procedure: POTENTIAL_UPDATE($\xi, y, \alpha$)
Output: A potential $p$ which satisfies (A[$\alpha$]), (B[$\alpha$]), and (4.4).

begin
    perform the strongly connected component decomposition of $G_\alpha(\xi, y)$;
    identify the source components;
    choose a vertex $v_C$ from each source component $C$;
    let $S$ be the collection of $v_C$ for all source components;
    compute the shortest path distance $q(v)$ from $S$ to each $v \in V$ in $G_\alpha(\xi, y)$
    with respect to the arc length $l$;
    return $q$
end
```

Figure 6: Procedure POTENTIAL_UPDATE

To sum up, procedure CUT_CANCEL aims at finding (π, φ) such that

$$0 \leq \pi \leq l_p(a) \qquad (a \in \vec{A}_\xi(\alpha)), \tag{4.5}$$

$$0 \leq \varphi(a) \leq \psi(a) \qquad (a \in \vec{D}), \tag{4.6}$$

$$y - \partial\varphi \in \arg\min f[-p - \pi\chi_W], \tag{4.7}$$

and that $D_{\psi'}(\alpha) \cup \{a \mid a \in A_\xi(\alpha), l_{p'}(a) = 0\}$ contains an arc from R to W.

It turns out that such (π, φ) exists and can be determined with the aid of the following maximum submodular flow problem on a bipartite graph $(R, W; \vec{D})$.

Maximum submodular flow problem MaxSFP(π)

$$\text{Maximize} \quad \sum_{a \in \vec{D}} \varphi(a) \tag{4.8}$$

$$\text{subject to} \quad 0 \leq \varphi(a) \leq \psi(a) \quad (a \in \vec{D}), \tag{4.9}$$

$$y - \partial\varphi \in \arg\min f[-p - \pi\chi_W], \tag{4.10}$$

$$\varphi(a) \in \mathbf{Z} \quad (a \in \vec{D}). \tag{4.11}$$

It should be clear that $\varphi : \vec{D} \to \mathbf{Z}$ is the variable to be optimized and π is a parameter that specifies the problem. Recall that $f[-p-\pi\chi_W]$ is an M-convex function for the given $p \in \mathbf{Z}^V$ and $\pi \in \mathbf{Z}$. Since $\arg\min f[-p - \pi\chi_W]$ is an M-convex set by Lemma 2.3, this problem is a maximum integral submodular flow problem. For the maximum submodular flow problem, a number of efficient algorithms are available [7, 8, 10, 31].

If $\vec{A}_\xi(\alpha)$ is nonempty, procedure CUT_CANCEL computes

$$\bar{\pi} = \min\{l_p(a) \mid a \in \vec{A}_\xi(\alpha)\}. \tag{4.12}$$

If MaxSFP($\bar{\pi}$) is feasible, then it returns $\bar{\pi}$ and an optimal flow $\bar{\varphi}$ of MaxSFP($\bar{\pi}$). Consequently, $l_{p'}(a) = 0$ holds for the arc $a \in \vec{A}_\xi(\alpha)$ that attains the minimum on the right-hand side of (4.12).

If MaxSFP($\bar{\pi}$) is infeasible or $\vec{A}_\xi(\alpha)$ is empty, procedure Cut_Cancel looks for π^* such that MaxSFP(π^*) is feasible and MaxSFP($\pi^* + 1$) is infeasible. If $\vec{A}_\xi(\alpha)$ is nonempty and MaxSFP($\bar{\pi}$) is infeasible, such π^* exists between 0 and $\bar{\pi}$ because MaxSFP(0) is feasible. We adopt $\pi^\circ = 0$ and $\pi^\bullet = \bar{\pi}$ as lower and upper bounds on π^*. If $A_\xi(\alpha)$ is empty, then we check the feasibility of MaxSFP(π) for $\pi = 1, 2, 4, 8, \ldots$, until we reach infeasible MaxSFP(π). The last π serves as an upper bound $\pi^\bullet$ and the previous π as a lower bound π° on π^*. It will be shown later in Lemma 4.3 that $\pi^\bullet$ is at most $2nK$.

We then find π^* by the following ordinary bisection procedure starting with lower and upper bounds π° and $\pi^\bullet$. At every iteration, check the feasibility of MaxSFP(π) for $\pi = \lceil(\pi^\circ + \pi^\bullet)/2\rceil$. If it is infeasible, then reset $\pi^\bullet$ to $\lceil(\pi^\circ + \pi^\bullet)/2\rceil$. Otherwise, reset π° to $\lceil(\pi^\circ + \pi^\bullet)/2\rceil$. Repeat this process until $\pi^\bullet = \pi^\circ + 1$. Then π° gives the desired value of π^*. We return π^* and an optimal flow φ^* of MaxSFP(π^*). It will be shown later in Lemma 4.2 that we have $\vec{D} \cap D_{\psi'}(\alpha) \neq \emptyset$ for $\psi' = \psi - \varphi^*$.

We summarize the procedure Cut_Cancel in Figure 7.

As a result of Cut_Cancel, an arc is added to the search tree and the set R becomes larger. This implies the following lemma.

Lemma 4.1. *In one execution of* Path_Search, *procedure* Cut_Cancel *is called at most $n - 1$ times.* □

For the justification of Path_Search, it now remains to show that $\vec{D} \cap D_{\psi'}(\alpha) \neq \emptyset$ for $\psi' = \psi - \varphi^*$ and that $\pi^\bullet \leq 2nK$.

Lemma 4.2. *Suppose that φ^* is optimal to* MaxSFP(π^*) *and that* MaxSFP($\pi^* + 1$) *is infeasible. Then there exists an arc $a \in \vec{D}$ such that $\varphi^*(a) = \psi(a)$, and hence $\vec{D} \cap D_{\psi'}(\alpha) \neq \emptyset$ for $\psi' = \psi - \varphi^*$.*

Proof. It suffices to show that, if φ is an optimal flow of MaxSFP(π) and $\varphi(a) < \psi(a)$ for every arc $a \in \vec{D}$, then φ is also feasible for MaxSFP($\pi + 1$). Define

$$\rho_\pi(X) = g(p + \pi\chi_W + \chi_X) - g(p + \pi\chi_W),$$

which is the submodular function associated with the M-convex set $\arg\min f[-p - \pi\chi_W]$; see Lemma 2.8. For $z = y - \partial\varphi \in \arg\min f[-p - \pi\chi_W]$, we have

$$z(X) = y(X) - \partial\varphi(X) \leq \rho_\pi(X) \quad (X \subseteq V). \tag{4.13}$$

Moreover, this inequality is tight for $X = W$, i.e.,

$$z(W) = \rho_\pi(W), \tag{4.14}$$

```
Procedure: Cut_Cancel($\xi, y, \psi, p, \alpha$)
Output: A pair of $\pi$ and $\varphi$ that satisfy (4.5)–(4.7).

begin
    if $\vec{A}_\xi(\alpha) \neq \emptyset$ then
    begin
        $\bar{\pi} := \min\{l_p(a) \mid a \in \vec{A}_\xi(\alpha)\}$;
        if MaxSFP($\bar{\pi}$) is feasible then
        begin
            $\bar{\varphi} :=$ optimal flow for MaxSFP($\bar{\pi}$);
            return $(\bar{\pi}, \bar{\varphi})$
        end
        else $\pi^\circ := 0$, $\pi^\bullet := \bar{\pi}$
    end
    else
    begin
        $\pi^\circ := 0$;
        $\pi := 1$;
        while MaxSFP($\pi$) is feasible do
        begin
            $\pi^\circ := \pi$;
            $\pi := 2\pi$
        end;
        $\pi^\bullet := \pi$
    end;
    repeat
        $\pi := \lceil(\pi^\circ + \pi^\bullet)/2\rceil$;
        if MaxSFP($\pi$) is infeasible then $\pi^\bullet := \pi$
        else $\pi^\circ := \pi$
    until $\pi^\bullet = \pi^\circ + 1$;
    $\pi^* := \pi^\circ$;
    $\varphi^* :=$ optimal flow for MaxSFP($\pi^*$);
    return $(\pi^*, \varphi^*)$
end
```

Figure 7: Procedure Cut_Cancel

which is shown as follows. Let Y be the maximal tight subset of W, where a tight set means a subset X for which the inequality (4.13) holds with equality. Suppose that there exists $v \in W \setminus Y$, and let U be the minimal tight subset of V containing v. Since $Y \cup U$ is also tight, the maximality of Y implies that there exists $u \in U \setminus W$. For $a = (u, v)$ we can increase the value of $\varphi(a)$ without violating the constraint $\varphi(a) \leq \psi(a)$. This does not violate (4.13) either, since there exists no tight X with $u \notin X$ and $v \in X$. This contradicts the optimality of φ.

For any $X \subseteq V$,

$$\begin{aligned} z(X) &= z(X \cup W) + z(X \cap W) - z(W) \\ &\leq \rho_\pi(X \cup W) + \rho_\pi(X \cap W) - \rho_\pi(W) \\ &= g(p + \pi\chi_W + \chi_{X\cup W}) + g(p + \pi\chi_W + \chi_{X\cap W}) \\ &\qquad - g(p + \pi\chi_W) - g(p + (\pi+1)\chi_W) \\ &\leq g(p + (\pi+1)\chi_W + \chi_X) - g(p + (\pi+1)\chi_W) \\ &= \rho_{\pi+1}(X), \end{aligned}$$

where the first inequality is by (4.13) and (4.14) and the second is by the discrete midpoint convexity in Lemma 2.5. This means $y - \partial\varphi \in \arg\min f[-p - (\pi+1)\chi_W]$, which shows that φ is a feasible flow of MaxSFP$(\pi + 1)$. □

We now show an explicit upper bound on $\pi^\bullet$.

Lemma 4.3. *The upper bound $\pi^\bullet$ used in* CUT_CANCEL *satisfies $\pi^\bullet \leq 2nK$.*

Proof. This is immediate from Lemmas 4.4 and 4.5 below. □

Lemma 4.4. *At the beginning of procedure* CUT_CANCEL, *$p(u) - p(v) \leq (n-1)K$ holds for any $u \in R$ and $v \in W$, and hence we have $\bar{\pi} \leq nK$.*

Proof. At the beginning of procedure PATH_SEARCH, p is modified by procedure POTENTIAL_UPDATE to satisfy (4.4). We prove that $p(u) - p(v)$ does not exceed $(n-1)K$ for any $u \in R$ and any $v \in W$ during an execution of procedure PATH_SEARCH. Let $p_i, \bar{\pi}_i, \pi^*_i, R_i$ and W_i denote $p, \bar{\pi}, \pi^*, R$ and W at the beginning of the i-th call of CUT_CANCEL in one execution of PATH_SEARCH. For any $u \in R_i$ and $v \in W_i$, we check how much $p(u) - p(v)$ has changed by the $(i-1)$-th call of CUT_CANCEL. In case of $u \in R_{i-1}$, since $v \in W_i \subseteq W_{i-1}$ and $\pi^*_{i-1} \geq 0$, we have

$$p_i(u) - p_i(v) = p_{i-1}(u) - (p_{i-1}(v) + \pi^*_{i-1}) \leq p_{i-1}(u) - p_{i-1}(v).$$

In the other case, we must have $u \in W_{i-1}$. Since the $(i-1)$-th call of CUT_CANCEL increases both $p(u)$ and $p(v)$ by π^*_{i-1}, we have

$$p_i(u) - p_i(v) = p_{i-1}(u) - p_{i-1}(v).$$

Thus, in either case, $p(u) - p(v)$ does not increase. Therefore, we have $p(u) - p(v) \leq (n-1)K$. This implies $\bar{\pi}_i \leq K + (n-1)K = nK$ for any i. □

Lemma 4.5. *In procedure* Cut_Cancel, *if* $\vec{A}_\xi(\alpha)$ *is empty, then* MaxSFP(π) *is infeasible for* $\pi > nK$.

Proof. Suppose to the contrary that MaxSFP(π) has a feasible flow φ.

We first claim that there exists no arc from R to W in E_z for $z = y - \partial\varphi$. For any pair of $u \in R$ and $v \in W$, define $z' = z - \chi_u + \chi_v$. Since $z \in \arg\min f[-p - \pi\chi_W]$ and $p(u) - p(v) \leq (n-1)K$, we have

$$\begin{aligned} f(z') - f(z) &= f[-p - \pi\chi_W](z') - f[-p - \pi\chi_W](z) + \pi + p(v) - p(u) \\ &> nK - (n-1)K = K, \end{aligned}$$

which implies $z' \notin \operatorname{dom} f$ by the definition of K, and hence $(u, v) \notin E_z$.

Let $\rho : 2^V \to \mathbf{Z}$ be the submodular function associated with the M-convex set $\operatorname{dom} f$. Since there exists no arc in E_z from R to W, we have

$$z(W) = \rho(W) \geq b(\Delta^+ W) - c(\Delta^- W), \tag{4.15}$$

where the inequality follows from Theorem 2.6. Since there exists no arc in $A_\xi(\alpha)$ from R to W, we also have

$$c(a) - \xi(a) < \alpha \quad (a \in \Delta^- W), \tag{4.16}$$

$$\xi(a) - b(a) < \alpha \quad (a \in \Delta^+ W). \tag{4.17}$$

Combining (4.15), (4.16), and (4.17), we obtain

$$\partial\xi(W) - z(W) \leq \alpha(|\Delta^+ W| + |\Delta^- W|) \leq \alpha n(n-1). \tag{4.18}$$

Since $\varphi(a) \leq \psi(a)$ for $a \in \vec{D}$ and $\psi(a) = 0$ for $a \in D$ from W to R, we have

$$\partial\psi(W) \leq \partial\varphi(W). \tag{4.19}$$

Furthermore, it follows from $R \cap S^-(\alpha) = \emptyset$ and $W \cap S^+(\alpha) = \emptyset$ that

$$x(v) - \partial\xi(v) > -\alpha \quad (v \in R), \tag{4.20}$$

$$x(v) - \partial\xi(v) < \alpha \quad (v \in W). \tag{4.21}$$

For $S = \{v \in V \mid x(v) < \partial\xi(v)\}$, we have

$$\begin{aligned} \|x - \partial\xi\|_1 &= -2\sum_{v \in S}(x(v) - \partial\xi(v)) \\ &= -2\left(\sum_{v \in R\cap S}(x(v) - \partial\xi(v)) + \sum_{v \in W\cap S}(x(v) - \partial\xi(v))\right) \\ &\leq 2(\alpha|R| + \alpha|W| + \partial\xi(W) - x(W)) \\ &= 2(\alpha|R| + \alpha|W| + \partial\xi(W) - z(W) - \partial\varphi(W) + \partial\psi(W)) \\ &\leq 2(\alpha n + \alpha n(n-1)) \\ &= 2\alpha n^2, \end{aligned}$$

where the first inequality is due to (4.20) and (4.21) and the second is to (4.18) and (4.19). This contradicts the fact that Path_Search is called only when $\|x - \partial\xi\|_1 > 2\alpha n^2$. ☐

4.3 Time Complexity

We now discuss the running time of our capacity scaling algorithm. The following lemma gives a bound on the number of augmentations in any α-scaling phase.

Lemma 4.6. *The number of augmentations in any α-scaling phase is* $\mathrm{O}(n^2)$.

Proof. When an α-scaling phase ends, we have $\|x - \partial\xi\|_1 \leq 2\alpha n^2$. Then we reduce α by a factor of two and start a new α-scaling phase. At the start of a new α-scaling phase, we modify ψ to satisfy the capacity constraints (4.1) for the new value of α, and modify x to maintain (4.2). We also modify ξ to resolve the possible violation of $(\mathrm{A}[\alpha])$ for arcs a with residual capacity $\alpha \leq r(a) < 2\alpha$. After these modifications, we have

$$\|x - \partial\xi\|_1 \leq 4\alpha n^2 + 2\alpha n^2 + 2\alpha m \leq 8\alpha n^2.$$

Since each augmentation reduces the discrepancy by 2α, the total number of augmentations in any α-scaling phase is at most $4n^2 = \mathrm{O}(n^2)$. □

Since the algorithm calls Path_Search before each augmentation, Lemma 4.6 implies the same $\mathrm{O}(n^2)$ bound on the number of calls of Path_Search in each scaling phase. We now provide the running time bound of Path_Search. Recall that F denotes the time for evaluating f.

Lemma 4.7. *Procedure* Path_Search *runs in* $\mathrm{O}(F \cdot n^4 \log L \log(nK))$ *time.*

Proof. At the start of procedure Path_Search, shortest path distances can be computed in $\mathrm{O}(n^3)$ time by the Ford-Bellman algorithm.

By Lemma 4.1, procedure Path_Search calls Cut_Cancel at most $n-1$ times. In each execution of Cut_Cancel, we need $\mathrm{O}(\log(nK))$ iterations for the bisection procedure since $\pi^\bullet \leq 2nK$ by Lemma 4.3. Each feasibility check in procedure Cut_Cancel requires to solve MaxSFP(π), which takes $\mathrm{O}(n^3 h)$ time by the algorithm of Fujishige and Zhang [10], where h denotes the time to compute an exchange capacity. In our framework of using a function evaluation oracle for the M-convex function, we have $h = \mathrm{O}(F \cdot \log L)$ by the binary search method. After the execution of procedure Cut_Cancel, the exchange capacity oracle is also called $\mathrm{O}(n^2)$ time. Thus, procedure Cut_Cancel runs in $\mathrm{O}(F \cdot n^3 \log L \log(nK))$ time.

□

At the start of algorithm Capacity_Scaling we can find a minimizer of an M-convex function in $\mathrm{O}(F \cdot n^3 \log L)$ time [29, 30]. Since we initially set $\alpha = 2^{\lfloor \log M \rfloor}$ with $M \leq 2L$, after $\mathrm{O}(\log L)$ scaling phases, we have $\alpha = 1$. By Lemmas 4.6 and 4.7, each scaling phase runs in $\mathrm{O}(F \cdot n^6 \log L \log(nK))$ time. Thus, the algorithm performs $\mathrm{O}(\log L)$ scaling phases in $\mathrm{O}(F \cdot n^6 (\log L)^2 \log(nK))$ time.

At the end of the 1-scaling phase, we have $\|y - \partial\xi\|_1 \leq \|x - \partial\xi\|_1 + \|\partial\psi\|_1 \leq 2n^2 + n(n-1) \leq 3n^2$. We then call SSP($\xi, y, p$), which performs at most $3n^2/2$ augmentations to obtain an optimal flow ξ of MCSFP. Since each construction of the auxiliary graph can be done by $\mathrm{O}(n^2)$ evaluations of f, the running time of this postprocess is $\mathrm{O}(F \cdot n^4)$, which is dominated by the above $\mathrm{O}(F \cdot n^6 (\log L)^2 \log(nK))$ bound. Thus we obtain the following theorem.

Theorem 4.8. *Algorithm* Capacity_Scaling *solves the M-convex submodular flow problem* MCSFP *in* $\mathrm{O}(F \cdot n^6 (\log L)^2 \log(nK))$ *time.* $\square$

Acknowledgements

The authors are grateful to Akihisa Tamura for helpful comments on the manuscript. This work is supported by the 21st Century COE Program on Information Science and Technology Strategic Core, a Grant-in-Aid for Scientific Research from the Ministry of Education, Culture, Sports, Science and Technology of Japan, and the Kayamori Foundation of Informational Science Advancement.

References

[1] Ahuja, R. K., Hochbaum, D. S., Orlin, J. B. (2003): Solving the convex cost integer dual network flow problem. Manage. Sci. **49**, 950–964

[2] Cunningham, W. H., Frank, A. (1985): A primal-dual algorithm for submodular flows. Math. Oper. Res. **10**, 251–262

[3] Dress, A. W. M., Wenzel, W. (1992): Valuated matroids. Adv. Math. **93**, 214–250

[4] Edmonds, J., Giles, R. (1977): A min-max relation for submodular functions on graphs. Ann. Discrete Math. **1**, 185–204

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

[6] Fleischer, L., Iwata, S., McCormick, S. T. (2002): A faster capacity scaling algorithm for minimum cost submodular flow. Math. Program., Ser. **A 92**, 119–139

[7] Frank, A. (1984): Finding feasible vectors of Edmonds-Giles polyhedra. J. Comb. Theory, Ser. **B 36**, 221–239

[8] Fujishige, S. (1978): Algorithms for solving the independent flow-problems. J. Oper. Res. Soc. Japan **21**, 189–204

[9] Fujishige, S. (1991): Submodular Functions and Optimization. North-Holland, Amsterdam

[10] Fujishige, S., Zhang, X. (1992): New algorithms for the intersection problem of submodular systems. Japan J. Indust. Appl. Math. **9**, 369–382

[11] Hochbaum, D. S. (1994): Lower and upper bounds for the allocation problem and other nonlinear optimization problems. Math. Oper. Res. **19**, 390–409

[12] Hochbaum, D. S. (2001): An efficient algorithm for image segmentation, Markov random fields and related problems. J. ACM **48**, 686–701

[13] Hochbaum, D. S., Queyranne, M. (2003): Minimizing a convex cost closure set. SIAM J. Discrete Math. **16**, 192–207

[14] Iwata, S. (1997): A capacity scaling algorithm for convex cost submodular flows. Math. Program. **76**, 299–308

[15] Iwata, S. (2003): A faster scaling algorithm for minimizing submodular functions. SIAM J. Comput. **32**, 833-840

[16] Iwata, S., Fleischer, L., Fujishige, S. (2001): A combinatorial strongly polynomial algorithm for minimizing submodular functions. J. ACM **48**, 761–777

[17] Iwata, S., McCormick, S. T., Shigeno, M. (1999): A strongly polynomial cut canceling algorithm for the submodular flow problem. In: G. Cornuéjols, R. E. Burkard and G. J. Woeginger (Eds.), Integer Programming and Combinatorial Optimization, Lecture Notes in Computer Science, **1610**, pp. 259–272. Springer-Verlag

[18] Iwata, S., Shigeno, M. (2003): Conjugate scaling algorithm for Fenchel-type duality in discrete convex optimization. SIAM J. Optim. **13**, 204–211

[19] Lovász, L. (1983): Submodular functions and convexity. In: A. Bachem, M. Grötschel and B. Korte (Eds.), Mathematical Programming — The State of the Art, pp. 235–257. Springer-Verlag

[20] Moriguchi, S., Murota, K. (2003): Capacity scaling algorithm for scalable M-convex submodular flow problems. Optim. Methods Softw. **18**, 207–218

[21] Murota, K. (1996): Valuated matroid intersection, I: optimality criteria. SIAM J. Discrete Math. **9**, 545–561

[22] Murota, K. (1996): Valuated matroid intersection, II: algorithms. SIAM J. Discrete Math. **9**, 562–576

[23] Murota, K. (1996): Convexity and Steinitz's exchange property. Adv. Math. **124**, 272–311

[24] Murota, K. (1998): Discrete convex analysis. Math. Program. **83**, 313–371

[25] Murota, K. (1999): Submodular flow problem with a nonseparable cost function. Combinatorica **19**, 87–109

[26] Murota, K. (2003): Discrete Convex Analysis. SIAM, Philadelphia

[27] Murota, K., Tamura, A. (2003): Application of M-convex submodular flow problem to mathematical economics. Japan J. Indust. Appl. Math. **20**, 257–277

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

[29] Shioura, A. (2003): Fast scaling algorithms for M-convex function minimization with application to resource allocation problem. Discrete Appl. Math. **134**, 303–316

[30] Tamura, A. (2002): Coordinatewise domain scaling algorithm for M-convex function minimization. In: W. J. Cook and A. S. Schulz (Eds.), Integer Programming and Combinatorial Optimization, Lecture Notes in Computer Science, **2337**, pp. 21–35. Springer-Verlag

[31] Tardos, É., Tovey, C. A., Trick, M. A. (1986): Layered augmenting path algorithms. Math. Oper. Res. **11**, 362–370