Fast Scaling Algorithms for M-convex Function Minimization with Application to the Resource Allocation Problem

Akiyoshi SHIOURA

February, 2002

Abstract

M-convex functions, introduced by Murota (1996, 1998), enjoy various desirable properties as "discrete convex functions." In this paper, we propose two new polynomial-time scaling algorithms for the minimization of an M-convex function. Both algorithms apply a scaling technique to a greedy algorithm for M-convex function minimization, and run as fast as the previous minimization algorithms. We also specialize our scaling algorithms for the resource allocation problem which is a special case of M-convex function minimization.

Keywords: discrete optimization, convex function, minimization, base polyhedron, resource allocation.

Akiyoshi SHIOURA: Graduate School of Information Sciences, Tohoku University, Sendai, 980-8579, Japan, `shioura@dais.is.tohoku.ac.jp`.

1 Introduction

In the area of discrete optimization, one of the important topics is to identify the discrete structure that guarantees the success of greedy algorithms. As an attempt to do this, various researchers have proposed discrete analogues of convex functions, or "discrete convex" functions (e.g., [4, 15]). Among them, the concept of M-convex functions, introduced by Murota [18, 19, 20, 21], affords a nice framework for well-solved discrete optimization problems with nonlinear objective functions such as the nonlinear resource allocation problem [12, 14] and the convex cost flow problem [1, 24].

Let V be a nonempty finite set. A function $f : \mathbf{Z}^V \to \mathbf{R} \cup \{+\infty\}$ is said to be M-convex if the effective domain $\mathrm{dom}\, f \subseteq \mathbf{Z}^V$ given by

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

is nonempty and f 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

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

where $\chi_w \in \{0,1\}^V$ is the characteristic vector of $w \in V$, and

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

M-convex function is a generalization of separable convex function over the base polyhedron of a submodular system [7] as well as valuated matroid by Dress–Wenzel [2, 3]. Also, M-convex functions enjoy various desirable properties as "discrete convexity" such as extendibility to ordinary convex functions, conjugacy, duality, etc.

In this paper, we consider the minimization of an M-convex function. It is a fundamental problem concerning M-convex functions, and several algorithms have been proposed so far. The local minimality implies the global minimality for M-convex functions. Therefore, a minimizer of an M-convex function can be found by a greedy (or descent) algorithm, which may require exponential time.

The first polynomial-time algorithm is given by Shioura [26]. It is shown that a given vector $x \in \mathrm{dom}\, f$ and a minimizer of f can be separated by using local information around the vector x, which we call "the minimizer cut property." Based on this, Shioura developed an $\mathrm{O}(n^4(\log L)^2)$-time algorithm, where the values n, L are given by

$$n = |V|, \qquad L = \max\{\|x - y\|_\infty \mid x, y \in \mathrm{dom}\, f\}. \tag{1.1}$$

Later, Moriguchi–Murota–Shioura [16] showed a proximity theorem and proposed a scaling approach for M-convex function minimization. Although the algorithm of Moriguchi et al. can be applied only to a restricted class of M-convex functions, it runs in $\mathrm{O}(n^3 \log(L/n))$ time. The scaling approach of Moriguchi et al. was polished up to a polynomial-time algorithm applicable to general M-convex functions by Tamura [27]. Tamura showed a common generalization of the minimizer cut property by Shioura [26] and the proximity theorem by Moriguchi et al. [16], which we call "the minimizer cut property with scaling." Based on this property and intricate analysis, Tamura proved that his scaling algorithm finds a minimizer in $\mathrm{O}(n^3 \log(L/n))$ time. Tamura's algorithm is the fastest so far for the minimization of a general M-convex function.

The main aim of this paper is to propose two scaling-based fast algorithms for the minimization of an M-convex function. As in the algorithm by Moriguchi et al. [16], both of our algorithms apply a scaling technique to a greedy algorithm. Our algorithms are developed on the basis of two different properties; the one is based on the minimizer cut property, and the other on the minimizer cut property with scaling. We show by simple analysis that our scaling algorithms run in $\mathrm{O}((n^3+n^2\log(L/n))(\log(L/n)/\log n))$ time and in $\mathrm{O}(n^3\log(L/n))$ time, respectively. Hence, our first algorithm is the fastest if $L = \mathrm{O}(n^n)$, and our second algorithm is as fast as Tamura's.

As a special case of M-convex function minimization, we also consider the minimization of a separable convex function over a base polyhedron, which is often called the resource allocation problem under submodular constraint [12, 14]. Various polynomial-time algorithms have been proposed for this problem [8, 9, 11, 17]. Currently, the fastest algorithm is the corrected version of Hochbaum's scaling algorithm [11] by Moriguchi–Shioura [17] and runs in $\mathrm{O}(n(\log n + F\log(B/n))\log(B/n))$ time, where F is the running time of the membership test in a submodular polyhedron, and B is a certain parameter associated with the constraint of the problem. In this paper, we specialize our scaling algorithms to the resource allocation problem, and show that the resulting algorithms run in $\mathrm{O}(n(\log n + F\log(B/n))\log(B/n))$ time and in $\mathrm{O}(n^2(\log n + F)\log(B/n^2))$ time, respectively.

The organization of this paper is as follows. Section 2 is devoted to review a greedy algorithm for the minimization of an M-convex function. Applying a scaling technique to the greedy algorithm, we propose two scaling algorithms in Sections 3 and 4. Finally, we specialize our scaling algorithms to the resource allocation problem in Section 5.

2 A Greedy Algorithm for M-convex Function Minimization

We review a greedy algorithm for M-convex function minimization.

We denote the set of reals and integers by $\mathbf{R}$ and $\mathbf{Z}$, respectively. Also, we denote by $\mathbf{Z}_{++}$ the set of positive integers. Throughout this paper, we assume that $f : \mathbf{Z}^V \to \mathbf{R} \cup \{+\infty\}$ is an M-convex function with bounded $\mathrm{dom}\, f$, and that we are given a vector $x_0 \in \mathrm{dom}\, f$ and an oracle for computing a function value of f in unit time. We denote by $\arg\min f$ the set of minimizers of f.

The greedy algorithm, also called the modified steepest descent algorithm [16], iteratively reduces a set containing a minimizer of f by using the following property.

Theorem 2.1 (minimizer cut property [26, Theorem 2.2]). *Let $f : \mathbf{Z}^V \to \mathbf{R} \cup \{+\infty\}$ be an M-convex function with $\arg\min f \neq \emptyset$, $x \in \mathrm{dom}\, f$ and $u \in V$. Suppose that $v \in V$ satisfies*

$$f(x + \chi_v - \chi_u) = \min_{w \in V} f(x + \chi_w - \chi_u). \tag{2.1}$$

Then, there exists $x^* \in \arg\min f$ satisfying $x^*(v) \geq x(v) + 1 - \chi_u(v)$.

The greedy algorithm also uses the following facts on M-convex functions.

Proposition 2.2. *Let $f : \mathbf{Z}^V \to \mathbf{R} \cup \{+\infty\}$ be an M-convex function.*
(i) *The effective domain $S = \mathrm{dom}\, f$ satisfies the following property:*

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

$$x - \chi_u + \chi_v \in S, \qquad y + \chi_u - \chi_v \in S.$$

In particular, we have $x(V) = y(V)$ for any $x, y \in \mathrm{dom}\, f$.
(ii) *Given a vector $l \in \mathbf{Z}^V$, define a function $f_l : \mathbf{Z}^V \to \mathbf{R} \cup \{+\infty\}$ by*

$$f_l(x) = \begin{cases} f(x) & (x \geq l), \\ +\infty & (\mathit{otherwise}). \end{cases} \tag{2.2}$$

Then, f_l is M-convex if $\mathrm{dom}\, f_l = \{x \in \mathbf{Z}^V \mid x \in \mathrm{dom}\, f,\ x \geq l\}$ is nonempty.

The greedy algorithm is described as follows. Let $x_0 \in \mathrm{dom}\, f$, and L and n be the values given by (1.1). We maintain a vector $l \in \mathbf{Z}^V$ to represent the set

$$S(l) \equiv \{x \in \mathbf{Z}^V \mid x \geq l\}$$

which always contains a minimizer of f. We also maintain a vector $x \in S(l) \cap \mathrm{dom}\, f$.

Algorithm GREEDY
Step 0: Put $x := x_0$, $l(w) := x_0(w) - L$ $(w \in V)$.
Step 1: If $x = l$, then output the current x and stop. [x is a minimizer of f]
Step 2: Choose any $u \in V$ with $x(u) > l(u)$.
Step 3: Find $v \in V$ satisfying (2.1).
Step 4: Put $l(v) := x(v) + 1 - \chi_u(v)$ and $x := x + \chi_v - \chi_u$. Go to Step 2. □

We have $S(l) \cap \arg\min f \neq \emptyset$ at Step 0. In each iteration, we reduce the set $S(l)$ by applying Theorem 2.1 to the M-convex function f_l given by (2.2). Proposition 2.2 (i) implies that if $x = l$ then x is a unique vector in $S(l) \cap \mathrm{dom}\, f$. Hence, the output of GREEDY is a minimizer of f. To analyze the number of iterations, we consider the value $\sum_{w \in V}\{x(w) - l(w)\}$, which is at most nL and decreases at least one in each iteration. Hence, GREEDY terminates in nL iterations.

In the following two sections, we apply a scaling technique to Algorithm GREEDY and develop two variants of polynomial-time scaling algorithms.

3 The First Scaling Algorithm

Our first algorithm SCALING1 uses a procedure called SCALEDGREEDY1$(\alpha,\ x,\ l)$. The input of Procedure SCALEDGREEDY1$(\alpha,\ x,\ l)$ is a scaling parameter $\alpha \in \mathbf{Z}_{++}$ and vectors $x, l \in \mathbf{Z}^V$ satisfying

$$x \in S(l) \cap \mathrm{dom}\, f, \qquad S(l) \cap \arg\min f \neq \emptyset. \tag{3.1}$$

Procedure SCALEDGREEDY1$(\alpha,\ x,\ l)$ consists of several phases called "Phase-u," in each of which we fix the element $u \in V$ in Theorem 2.1 and reduce the set $S(l)$ by applying Theorem 2.1 to the

M-convex function f_l given by (2.2). For a vector $x \in \mathrm{dom}\, f$ and $u, v \in V$, we define the *exchange capacity* $\widehat{c}(x; v, u)$ by

$$\widehat{c}(x; v, u) = \sup\{\beta \in \mathbf{Z} \mid x + \beta(\chi_v - \chi_u) \in \mathrm{dom}\, f\}.$$

By Proposition 2.2 (i), we have $x + \beta(\chi_v - \chi_u) \in \mathrm{dom}\, f$ for any $\beta \in \mathbf{Z}$ with $0 \le \beta \le \widehat{c}(x; v, u)$.

Procedure ScaledGreedy1(α, x, l)
Step 0: Put $V' := V$.
Step 1: If $V' = \emptyset$, then output the current x and l, and stop.
Step 2: Choose any $u \in V'$.
Step 3: [Phase-u starts]
  Step 3-1: Find $v \in V$ satisfying (2.1).
  Step 3-2: [Phase-u ends] If $v = u$ or $x(u) = l(u)$, then put $l(u) := x(u)$, $V' := V' \setminus \{u\}$. Go to Step 2.
  Step 3-3: [full iteration] If $x + \alpha(\chi_v - \chi_u) \in \mathrm{dom}\, f$ and $x(u) - \alpha \ge l(u)$, then put $l(v) := x(v) + 1$, $x := x + \alpha(\chi_v - \chi_u)$, and $V' := V' \setminus \{v\}$. Go to Step 3-1.
  Step 3-4: [partial iteration] Compute the value $\alpha' = \min\{\widehat{c}(x; v, u), x(u) - l(u)\}$. Put $l(v) := x(v) + 1$, $x := x + \alpha'(\chi_v - \chi_u)$, and $V' := V' \setminus \{v\}$. Go to Step 3-1. □

Lemma 3.1. *Let $\alpha \in \mathbf{Z}_{++}$ and $x, l \in \mathbf{Z}^V$ be vectors satisfying the condition* (3.1).
(i) *When* ScaledGreedy1(α, x, l) *terminates, the vectors $x, l \in \mathbf{Z}^V$ satisfy the condition* (3.1) *and the inequality $x(w) - l(w) \le \alpha - 1$ for all $w \in V$.*
(ii) *The running time of* ScaledGreedy1(α, x, l) *is* $\mathrm{O}(n \sum_{w \in V}\{x(w) - l(w)\}/\alpha + (n + \log_2 \alpha) n^2)$.

Proof. (i): The condition (3.1) is satisfied in each iteration, and if $w \in V$ is deleted from V' in some iteration, then $0 \le x(w) - l(w) \le \alpha - 1$ holds in the following iterations. Hence, we have the claim.

(ii): We denote by $x_0, l_0 \in \mathbf{Z}^V$ the vectors x, l given as the input of ScaledGreedy1(α, x, l). It suffices to show that the running time of Phase-u is $\mathrm{O}(n\{x_0(u) - l_0(u)\}/\alpha + (n + \log_2 \alpha)n)$. We classify the iterations in Phase-u into two types: we call an iteration *full* if Step 3-3 is performed, and *partial* if Step 3-4 is performed.

We first consider full iterations in Phase-u. Each full iteration takes $\mathrm{O}(n)$ time. We have $x(u) = x_0(u)$ and $l(u) = l_0(u)$ at the beginning of Phase-u. The value $l(u)$ remains the same, and $x(u)$ does not increase and is at least $l(u)$ during Phase-u. Moreover, $x(u)$ decreases by α in each full iteration, implying that the number of full iterations is at most $\{x_0(u) - l_0(u)\}/\alpha$. Hence, it takes $\mathrm{O}(n\{x_0(u) - l_0(u)\}/\alpha)$ time for full iterations.

We then analyze the total running time for partial iterations in Phase-u.

Claim. Let $x, y \in \mathrm{dom}\, f$ and $u \in V$. Suppose $x(u) > y(u)$ for $u \in V$ and $x(w) \le y(w)$ for all $w \in V \setminus \{u\}$. Then, we have $y + \chi_v - \chi_u \notin \mathrm{dom}\, f$ for all $w \in V \setminus \{u\}$ with $x + \chi_w - \chi_u \notin \mathrm{dom}\, f$.

[Proof of Claim] Let $w \in V \setminus \{u\}$ satisfy $x + \chi_w - \chi_u \notin \mathrm{dom}\, f$, and suppose, to the contrary, that $y' = y + \chi_w - \chi_u \in \mathrm{dom}\, f$ holds. Since $w \in \mathrm{supp}^+(y' - x)$ and $\mathrm{supp}^-(y' - x) = \{u\}$, the property (B-EXC) for $y', x \in \mathrm{dom}\, f$ implies $x + \chi_w - \chi_u \in \mathrm{dom}\, f$, a contradiction. [End of Claim]

For each $w \in V \setminus \{u\}$, the value $x(w)$ do not decrease in Phase-u, which, together with Claim above, implies that $x(w)$ increases in at most one partial iteration, i.e., we have at most $n-1$ partial iterations in Phase-u. We can compute the exchange capacity $\widehat{c}(x; v, u)$ in $\mathrm{O}(\log_2 \alpha)$ time by binary search since $x + \alpha(\chi_v - \chi_u) \notin \mathrm{dom}\, f$. Hence, partial iterations take $\mathrm{O}((n + \log_2 \alpha)n)$ time in total. $\square$

We now give the description of Algorithm Scaling1. The scaling parameter α is initially set to $n^{\lceil \log_n (L/n) \rceil}$ ($\simeq L/n$), and divided by n at the end of each iteration.

Algorithm Scaling1
Step 0: Compute the value L given by (1.1). Put $x := x_0$, $l(w) := x_0(w) - L$ $(w \in V)$, and $\alpha := n^{\lceil \log_n (L/n) \rceil}$.
Step 1: If $\alpha < 1$ then output x and stop. [The current x is optimal]
Step 2: Use Procedure ScaledGreedy1(α, x, l) to obtain vectors $x', l' \in \mathbf{Z}^V$.
Step 3: Put $x := x'$, $l := l'$ and $\alpha := \alpha/n$. Go to Step 1. $\square$

Theorem 3.2. *Suppose that a vector $x_0 \in \mathrm{dom}\, f$ is given. Then, Algorithm* Scaling1 *finds a minimizer of f in $\mathrm{O}((n^3 + n^2 \log_2(L/n))\{\log_2(L/n)/\log_2 n\})$ time.*

Proof. We first show the correctness of the algorithm. The condition (3.1) is satisfied at Step 0. By Lemma 3.1 (i), the condition (3.1) is also satisfied at the beginning of each iteration. If $\alpha = 1$, then the output x', l' of ScaledGreedy1(α, x, l) satisfy $x' = l'$ by Lemma 3.1 (i). Hence, we have $x' \in \arg\min f$, i.e., the output of Scaling1 is a minimizer of f.

We then analyze the running time. By Lemma 3.1 (i), we have $x(w) - l(w) \leq n\alpha$ $(w \in V)$ at the beginning of Step 2 in Algorithm Scaling1. From this inequality and Lemma 3.1 (ii) follows that each iteration takes $\mathrm{O}(n^3 + n^2 \log_2(L/n))$ time. The number of iterations of Scaling1 is $\mathrm{O}(\log_n(L/n)) = \mathrm{O}(\log_2(L/n)/\log_2 n)$, and the value L can be computed in $\mathrm{O}(n^2 \log_2 L)$ time by using the fact that $\mathrm{dom}\, f$ satisfies the property (B-EXC) (see, e.g., [26]). This concludes the proof. $\square$

4 The Second Scaling Algorithm

Our second algorithm Scaling2 uses a procedure called ScaledGreedy2(α, x, l), which also maintains the set $S(l)$ containing a minimizer and reduces $S(l)$ by exploiting the following property.

Theorem 4.1 (minimizer cut property with scaling [27, Theorem 2.6]). *Let $f : \mathbf{Z}^V \to \mathbf{R} \cup \{+\infty\}$ be an M-convex function with $\arg\min f \neq \emptyset$. Also, let $x \in \mathrm{dom}\, f$, $u \in V$, and $\alpha \in \mathbf{Z}_{++}$. Suppose that $v \in V$ satisfies*

$$f(x + \alpha(\chi_v - \chi_u)) = \min_{w \in V} f(x + \alpha(\chi_w - \chi_u)). \tag{4.1}$$

Then, there exists $x^* \in \arg\min f$ satisfying $x^*(v) \geq x(v) + \alpha(1 - \chi_u(v)) - (n-1)(\alpha - 1)$.

Procedure ScaledGreedy2(α, x, l)
Step 0: Put $V' := V$.
Step 1: If $V' = \emptyset$, then output the current x and l, and stop.

Step 2: Choose any $u \in V'$.
Step 3: Find $v \in V$ satisfying (4.1).
Step 4: If $v = u$ or $x(u) - \alpha < l(u)$, then put $l(u) := \max\{l(u), x(u) - (n-1)(\alpha-1)\}$ and $V' := V' \setminus \{u\}$. Go to Step 1.
Step 5: Put $l(v) := \max\{l(v), x(v) + \alpha - (n-1)(\alpha-1)\}$, $x := x + \alpha(\chi_v - \chi_u)$, and $V' := V' \setminus \{v\}$. Go to Step 1. □

Lemma 4.2. *Let $\alpha \in \mathbf{Z}_{++}$ and $x, l \in \mathbf{Z}^V$ be vectors satisfying the condition* (3.1)*.*
(i) *When* ScaledGreedy2(α, x, l) *terminates, the vectors $x, l \in \mathbf{Z}^V$ satisfy the condition* (3.1) *and the inequality $x(w) - l(w) \le (n-1)(\alpha-1)$ for all $w \in V$.*
(ii) *The running time of* ScaledGreedy2(α, x, l) *is* $\mathrm{O}(n \sum_{w \in V}\{x(w) - l(w)\}/\alpha)$.

Proof. The claim (i) can be shown in a similar way as Lemma 3.1 (i). To prove (ii), we denote by $x_0, l_0 \in \mathbf{Z}^V$ the vectors x, l given as the input of ScaledGreedy2(α, x, l). For each $w \in V$ the value $x(w)$ decreases by α at most $\lfloor \{x_0(w) - l_0(w)\}/\alpha \rfloor$ time until w is deleted from V' in Step 4 or 5. Since each iteration requires $\mathrm{O}(n)$ time, the claim follows. □

We now give the description of Algorithm Scaling2. The scaling parameter α is initially set to $2^{\lceil \log_2(L/2n) \rceil}$ ($\simeq L/2n$), and divided by two at the end of each iteration.

Algorithm Scaling2
Step 0: Compute the value L given by (1.1). Put $x := x_0$, $l(w) := x_0(w) - L$ ($w \in V$), and $\alpha := 2^{\lceil \log_2(L/2n) \rceil}$.
Step 1: If $\alpha < 1$ then output x and stop. [The current x is optimal]
Step 2: Use Procedure ScaledGreedy2(α, x, l) to obtain vectors $x', l' \in \mathbf{Z}^V$.
Step 3: Put $x := x'$, $l := l'$ and $\alpha := \alpha/2$. Go to Step 1. □

We can prove the following statement in a similar way as Theorem 3.2 by using Lemma 4.2.

Theorem 4.3. *Suppose that a vector $x_0 \in \mathrm{dom}\, f$ is given. Then, Algorithm* Scaling2 *finds a minimizer of f in $\mathrm{O}(n^3 \log_2(L/n))$ time.*

5 Application to the Resource Allocation Problem

We first explain the resource allocation problem. For any $x \in \mathbf{Z}^V$ and any $S \subseteq V$, we define $x(S) = \sum_{w \in S} x(w)$. A function $f : \mathbf{Z} \to \mathbf{R}$ is said to be *convex* if it satisfies $2f(\alpha) \le f(\alpha-1) + f(\alpha+1)$ for all $\alpha \in \mathbf{Z}$. A set function $\rho : 2^V \to \mathbf{Z} \cup \{+\infty\}$ is called *submodular* if it satisfies $\rho(S) + \rho(T) \ge \rho(S \cap T) + \rho(S \cup T)$ ($S, T \subseteq V$). We define

$$\mathrm{P}(\rho) = \{x \in \mathbf{Z}^V \mid x(S) \le \rho(S) \ (S \subseteq V)\},$$

which is called the *submodular polyhedron* associated with the submodular function ρ (see [7]).

Given a family of convex functions $f_w : \mathbf{Z} \to \mathbf{R}$ ($w \in V$), a submodular function $\rho : 2^V \to \mathbf{Z} \cup \{+\infty\}$ with $\rho(\emptyset) = 0$ and $\rho(V) < +\infty$, and a vector $l_0 \in \mathbf{Z}^V$, we consider the following problem:

$$
\text{(RAP)} \left|\begin{array}{ll} \text{Minimize} & f(x) = \sum_{w\in V} f_w(x(w)) \\ \text{subject to} & x(V) = \rho(V),\ x \in \mathrm{P}(\rho),\ x \geq l_0, \end{array}\right.
$$

which is called the resource allocation problem under the submodular constraint [12, 14]. In the following, we assume that (RAP) has a feasible solution, where we note that (RAP) is feasible if and only if $l_0 \in \mathrm{P}(\rho)$ [7, Theorem 2.3]. We denote by $Q^*(\rho, l_0)$ the set of optimal solutions of (RAP).

It is well-known that the resource allocation problem (RAP) can be solved by the following greedy algorithm (see, e.g., [5, 12, 14]). For each $w \in V$ and $\beta \in \mathbf{Z}$, we define $\Delta f_w(\beta) = f_w(\beta+1) - f_w(\beta)$.

Algorithm Greedy_RAP
Step 0: Put $x := l_0$.
Step 1: If $x + \chi_w \notin \mathrm{P}(\rho)$ for all $w \in V$, then output the current x and stop. [x is optimal]
Step 2: Find $v \in V$ such that $x+\chi_v \in \mathrm{P}(\rho)$ and $\Delta f_v(x(v)) = \min\{\Delta f_w(x(w)) \mid w \in V, x+\chi_w \in \mathrm{P}(\rho)\}$.
Step 3: Put $x(v) := x(v) + 1$. Go to Step 1. □

Problem (RAP) is related to M-convex functions as follows. Given an instance of (RAP), we define a function $\widetilde{f} : \mathbf{Z}^{\widetilde{V}} \to \mathbf{R} \cup \{+\infty\}$ by

$$
\widetilde{f}(x, x(v_0)) = \begin{cases} \displaystyle\sum_{w\in V} f_w(x(w)) + Mx(v_0) & (x \in \mathrm{P}(\rho),\ x \geq l_0,\ x(V) + x(v_0) = 0), \\ +\infty & (\text{otherwise}), \end{cases} \tag{5.1}
$$

where $(x, x(v_0)) \in \mathbf{Z}^{\widetilde{V}}$, $\widetilde{V} = V \cup \{v_0\}$ and M is a sufficiently large positive number. Then, the function $\widetilde{f}$ is M-convex (see [19, 20, 21]), and $(x, -x(V)) \in \arg\min \widetilde{f}$ if and only if $x \in Q^*(\rho, l_0)$. Based on this fact, we can specialize the scaling algorithms in Sections 3 and 4 to the problem (RAP).

5.1 Specializing the First Scaling Algorithm

We specialize our first scaling algorithm to the problem (RAP). In fact, the resulting algorithm Scaling1_RAP is essentially the same as the corrected version of Hochbaum's scaling algorithm [17].

The minimizer cut property (Theorem 2.1) for the M-convex function $\widetilde{f}$ defined by (5.1) and $u = v_0$ turns into the following property for (RAP):

Theorem 5.1. *Let $x \in \mathrm{P}(\rho)$ be a vector with $x \geq l_0$. Suppose that $v \in V$ satisfies*

$$
x + \chi_v \in \mathrm{P}(\rho), \quad \Delta f_v(x(v)) = \min\{\Delta f_w(x(w)) \mid w \in V,\ x + \chi_w \in \mathrm{P}(\rho)\}. \tag{5.2}
$$

Then, there exists an optimal solution x^* of (RAP) *satisfying $x^*(v) > x(v)$.*

Using this property, we specialize Procedure ScaledGreedy1(α, x, l). The input of the specialized version ScaledGreedy1_RAP(α, l) is a scaling parameter $\alpha \in \mathbf{Z}_{++}$ and a vector $l \in \mathbf{Z}^V$ satisfying

$$
l \in \mathrm{P}(\rho), \quad l \geq l_0, \quad S(l) \cap Q^*(\rho, l_0) = \emptyset. \tag{5.3}
$$

For a vector $x \in \mathrm{P}(\rho)$ and an element $w \in V$, we define $\widehat{c}(x, w) = \sup\{\beta \in \mathbf{Z} \mid x + \beta\chi_w \in \mathrm{P}(\rho)\}$.

Procedure SCALEDGREEDY1_RAP(α, l)
Step 0: Put $x := l$.
Step 1: If $x + \chi_w \notin \mathrm{P}(\rho)$ for all $w \in V$, then output the current l. Stop.
Step 2: Find $v \in V$ satisfying (5.2).
Step 3: [full iteration] If $x + \alpha\chi_v \in \mathrm{P}(\rho)$, then put $l(v) := x(v) + 1$ and $x := x + \alpha\chi_v$. Go to Step 1.
Step 4: [partial iteration] Put $l(v) := x(v) + 1$ and $x := x + \alpha'\chi_v$ with $\alpha' = \widehat{c}(x, v)$. Go to Step 1. □

Note that SCALEDGREEDY1_RAP(α, l) consists of only one Phase-u with $u = v_0$. We denote by F the running time of the membership test in the submodular polyhedron $\mathrm{P}(\rho)$, where it is noted that the membership test in $\mathrm{P}(\rho)$ can be done in strongly polynomial time by minimizing a certain submodular function (see [10, 13, 25]).

Lemma 5.2. *Let $\alpha \in \mathbf{Z}_{++}$ and $l \in \mathbf{Z}^V$ be a vector satisfying the condition* (5.3).
(i) *When* SCALEDGREEDY1(α, l) *terminates, the vectors $x, l \in \mathbf{Z}^V$ satisfy* (5.3), $x(w) - l(w) \le \alpha - 1$ ($\forall w \in V$)*, and* $x(V) = \rho(V)$.
(ii) *The running time of* SCALEDGREEDY1_RAP(α, l) *is* $\mathrm{O}((\log_2 n + F)\{\rho(V) - l(V)\}/\alpha + (\log_2 n + F\log_2\alpha)n)$.

Proof. (i): When the procedure terminates, we have $x \in \mathrm{P}(\rho)$ and $x + \chi_w \notin \mathrm{P}(\rho)$ for all $w \in V$, implying $x(V) = \rho(V)$. The other claims can be shown similarly to Lemma 3.1 (i).

(ii): The value $x(V)$ is initially equal to $l(V)$ and at most $\rho(V)$, from which follows that the number of full iterations is at most $\{\rho(V) - x(V)\}/\alpha$. We can show that for each $w \in V$ the value $x(w)$ increases in at most one partial iteration, i.e., the number of partial iterations is at most n. Step 2 can be done in $\mathrm{O}(\log_2 n)$ time by using a data structure such as priority queue, and the value $\widehat{c}(x, w)$ can be computated in $\mathrm{O}(F\log_2\alpha)$ time. Hence, full and partial iterations take $\mathrm{O}(\log_2 n + F)$ time and $\mathrm{O}(\log_2 n + F\log\alpha)$ time, respectively. This concludes the proof of (ii). □

In the algorithm SCALING1_RAP, we modify the initialization and the update of the scaling parameter α to reduce the running time. We put $B = \rho(V) - l_0(V)$.

Algorithm SCALING1_RAP
Step 0: Put $l := l_0$ and $\alpha := 2^{\lceil \log_2(B/2n) \rceil}$.
Step 1: If $\alpha < 1$ then output l and stop. [The current l is optimal]
Step 2: Use Procedure SCALEDGREEDY1_RAP(α, l) to obtain a vector $l' \in \mathbf{Z}^V$.
Step 3: Put $l := l'$ and $\alpha := \alpha/2$. Go to Step 1. □

Theorem 5.3. *Algorithm* SCALING1_RAP *finds an optimal solution of* (RAP) *in* $\mathrm{O}(n(\log_2 n + F\log_2(B/n))\log_2(B/n))$ *time.*

Proof. Lemma 5.2 (i) implies that $\rho(V) - l(V) \le (2\alpha - 1)n$ holds at the beginning of Step 2 in SCALING1_RAP. From this and Lemma 5.2 (ii) follows that each iteration takes $\mathrm{O}(n(\log_2 n + F\log_2(B/n)))$ time. Since the number of iterations of SCALING1_RAP is $\mathrm{O}(\log_2(L/n))$, we have the claim. □

5.2 Specializing the Second Scaling Algorithm

We specialize our second scaling algorithm to the problem (RAP). The minimizer cut property with scaling (Theorem 4.1) for the M-convex function $\widetilde{f}$ defined by (5.1) and $u = v_0$ turns into the following property for (RAP). For $w \in V$, $\alpha \in \mathbf{Z}_{++}$, and $\beta \in \mathbf{Z}$, we define $\Delta^{(\alpha)} f_w(\beta) = f_w(\beta + \alpha) - f_w(\beta)$.

Theorem 5.4. *Let $\alpha \in \mathbf{Z}_{++}$, and $x \in \mathrm{P}(\rho)$ be a vector with $x \geq l_0$. Suppose that $v \in V$ satisfies*

$$x + \alpha\chi_v \in \mathrm{P}(\rho), \quad \Delta^{(\alpha)} f_v(x(v)) = \min\{\Delta^{(\alpha)} f_w(x(w)) \mid w \in V,\ x + \alpha\chi_w \in \mathrm{P}(\rho)\}. \tag{5.4}$$

Then, there exists an optimal solution x^* of (RAP) *satisfying $x^*(v) \geq x(v) + \alpha - (n-1)(\alpha - 1)$.*

Using this property, we specialize Procedure ScaledGreedy2$(\alpha,\, l)$. The input of the specialized version ScaledGreedy2_RAP$(\alpha,\, l)$ is a scaling parameter $\alpha \in \mathbf{Z}_{++}$ and a vector $l \in \mathbf{Z}^V$ with (5.3).

Procedure ScaledGreedy2_RAP$(\alpha,\, l)$
Step 0: Put $x := l$.
Step 1: If $x + \alpha\chi_w \notin \mathrm{P}(\rho)$ for all $w \in V$, then output the current l. Stop.
Step 2: Find $v \in V$ satisfying (5.4).
Step 3: Put $l(v) := \max\{l(v), x(v) + \alpha - (n-1)(\alpha - 1)\}$ and $x := x + \alpha\chi_v$. Go to Step 1. □

Lemma 5.5. *Let $\alpha \in \mathbf{Z}_{++}$ and $l \in \mathbf{Z}^V$ be a vector satisfying the condition* (5.3).
(i) *When* ScaledGreedy2_RAP$(\alpha,\, l)$ *terminates, the vectors $x, l \in \mathbf{Z}^V$ satisfy* (5.3)*, $x(w) - l(w) \leq (n-1)(\alpha - 1)$ $(w \in V)$, and $\rho(V) - x(V) \leq n(\alpha - 1)$.*
(ii) *The running time of* ScaledGreedy2_RAP$(\alpha,\, l)$ *is* $\mathrm{O}((\log_2 n + F)\{\rho(V) - l(V)\}/\alpha)$.

Proof. We show the inequality $\rho(V) - x(V) \leq n(\alpha - 1)$ only. The proof of the other claims is similar to that for Lemma 5.2. Since $\mathrm{P}(\rho)$ is a submodular polyhedron, there exists a vector $y \in \mathrm{P}(\rho)$ with $y \geq x$ and $y(V) = \rho(V)$ [7, Theorem 2.3]. Since $x + \{y(w) - x(w)\}\chi_w \in \mathrm{P}(\rho)$ and $x + \alpha\chi_w \notin \mathrm{P}(\rho)$ for all $w \in V$, we have $y(w) - x(w) < \alpha$, from which follows $\rho(V) - x(V) = y(V) - x(V) \leq n(\alpha - 1)$. □

The following algorithm is a specialized version of Scaling2. Recall that $B = \rho(V) - l(V)$.

Algorithm Scaling2_RAP
Step 0: Put $l := l_0$ and $\alpha := 2^{\lceil \log_2(B/2n) \rceil}$.
Step 1: If $\alpha < 1$ then output l and stop. [The current l is optimal]
Step 2: Use Procedure ScaledGreedy2_RAP$(\alpha,\, l)$ to obtain a vector $l' \in \mathbf{Z}^V$.
Step 3: Put $l := l'$ and $\alpha := \alpha/2$. Go to Step 1. □

We can show the following theorem in the same way as Theorem 5.3 by using Lemma 5.5.

Theorem 5.6. *Algorithm* Scaling2_RAP *finds an optimal solution of* (RAP) *in* $\mathrm{O}(n^2(\log_2 n + F)\log_2(B/n))$ *time.*

6 Concluding Remarks

In this paper, we proposed two fast scaling algorithms for the minimization of an M-convex function. In fact, these algorithms can be applied to a wider class of functions called *semistrictly quasi M-convex functions* introduced by Murota–Shioura [23].

A function $f : \mathbf{Z}^V \to \mathbf{R} \cup \{+\infty\}$ is said to be semistrictly quasi M-convex if $\mathrm{dom}\, f$ is nonempty and f satisfies the following property:

(SSQM) $\forall x, y \in \mathrm{dom}\, f$, $\forall u \in \mathrm{supp}^+(x-y)$, $\exists v \in \mathrm{supp}^-(x-y)$:
(i) $\Delta f(x; v, u) \geq 0 \Longrightarrow \Delta f(y; u, v) \leq 0$, and (ii) $\Delta f(y; u, v) \geq 0 \Longrightarrow \Delta f(x; v, u) \leq 0$,

where $\Delta f(x; v, u) = f(x - \chi_u + \chi_v) - f(x)$ for $x \in \mathrm{dom}\, f$ and $u, v \in V$. It is easy to see that (M-EXC) implies (SSQM). We also consider a slightly weaker version of (SSQM):

(SSQM$^{\neq}$) $\forall x, y \in \mathrm{dom}\, f$ with $f(x) \neq f(y)$, $\forall u \in \mathrm{supp}^+(x-y)$, $\exists v \in \mathrm{supp}^-(x-y)$:
(i) $\Delta f(x; v, u) \geq 0 \Longrightarrow \Delta f(y; u, v) \leq 0$, and (ii) $\Delta f(y; u, v) \geq 0 \Longrightarrow \Delta f(x; v, u) \leq 0$.

The following theorems show that the minimizer cut property (Theorem 2.1) and the minimizer cut property with scaling (Theorem 4.1), which are the key properties to prove the correctness of our scaling algorithms, still hold for semistrictly quasi M-convex functions.

Theorem 6.1 ([23, Theorem 4.3]). *Let* $f : \mathbf{Z}^V \to \mathbf{R} \cup \{+\infty\}$ *be a function with* (SSQM$^{\neq}$) *and* $\arg\min f \neq \emptyset$. *Also, let* $x \in \mathrm{dom}\, f$ *and* $u \in V$. *Suppose that* $v \in V$ *satisfies* (2.1). *Then, there exists* $x^* \in \arg\min f$ *satisfying* $x^*(v) \geq x(v) + 1 - \chi_u(v)$.

Theorem 6.2 ([27, Theorem 4.4]). *Let* $f : \mathbf{Z}^V \to \mathbf{R} \cup \{+\infty\}$ *be a function with* (SSQM$^{\neq}$) *and* $\arg\min f \neq \emptyset$. *Also, let* $x \in \mathrm{dom}\, f$, $u \in V$, *and* $\alpha \in \mathbf{Z}_{++}$. *Suppose that* $v \in V$ *satisfies* (4.1). *Then, there exists* $x^* \in \arg\min f$ *satisfying* $x^*(v) \geq x(v) + \alpha(1 - \chi_u(v)) - (n-1)(\alpha - 1)$.

Hence, both of our scaling algorithms Scaling1 and Scaling2 also work for the minimization of a semistrictly quasi M-convex function. Note that the effective domain $\mathrm{dom}\, f$ of a function f with (SSQM$^{\neq}$) does not necessarily satisfy (B-EXC).

Theorem 6.3. *Let* $f : \mathbf{Z}^V \to \mathbf{R} \cup \{+\infty\}$ *be a function with* (SSQM$^{\neq}$).
(i) *Suppose that* $\mathrm{dom}\, f$ *satisfies the property* (B-EXC) *and that a vector* $x_0 \in \mathrm{dom}\, f$ *is given. Then, Algorithm* Scaling1 *finds a minimizer of* f *in* $\mathrm{O}((n^3 + n^2 \log_2(L/n))\{\log_2(L/n)/\log_2 n\})$ *time.*
(ii) *Suppose that the value* L *defined by (1.1) and a vector* $x_0 \in \mathrm{dom}\, f$ *are given. Then, Algorithm* Scaling2 *finds a minimizer of* f *in* $\mathrm{O}(n^3 \log_2(L/n))$ *time.*

Acknowledgement

The author thanks Kazuo Murota for valuable comments on the manuscript, and Akihisa Tamura for stimulating discussions which lead to the second scaling algorithm. This work is supported by Grant-in-Aid of the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, Network Flows: Theory, Algorithms, and Applications (Prentice Hall, Englewood Cliffs, 1993).

[2] A. W. M. Dress and W. Wenzel, Valuated matroid: A new look at the greedy algorithm, Appl. Math. Lett. 3 (1990) 33–35.

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

[4] P. Favati and F. Tardella, Convexity in nonlinear integer programming, Ricerca Operativa 53 (1990) 3–44.

[5] A. Federgruen and H. Groenevelt, The greedy procedure for resource allocation problems: necessary and sufficient conditions for optimality, Oper. Res. 34 (1986) 909–918.

[6] A. Frank, Generalized Polymatroids, in: A. Hajnal et al., eds., Finite and Infinite Sets (North-Holland, Amsterdam, 1984) 285–294.

[7] S. Fujishige, Submodular Functions and Optimization, Annals of Discrete Mathematics 47 (North-Holland, Amsterdam, 1991).

[8] E. Girlich, M. Kovalev, and A. Zaporozhets, A polynomial algorithm for resource allocation problems with polymatroid constraints, Optimization 37 (1996) 73–86.

[9] H. Groenevelt, Two algorithms for maximizing a separable concave function over a polymatroid feasible region, European J. Oper. Res. 54 (1991) 227–236.

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

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

[12] T. Ibaraki and N. Katoh, Resource Allocation Problems: Algorithmic Approaches (MIT Press, Boston, MA, 1988).

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

[14] N. Katoh and T. Ibaraki, Resource Allocation Problems, in: D. -Z. Du and P. M. Pardalos, eds., Handbook of Combinatorial Optimization II (Kluwer Academic Publishers, Boston, MA, 1998) 159–260.

[15] B. L. Miller, On minimizing nonseparable functions defined on the integers with an inventory application, SIAM J. Appl. Math. 21 (1971) 166–185.

[16] S. Moriguchi, K. Murota, and A. Shioura, Scaling algorithms for M-convex function minimization, IEICE Trans. Fundamentals (2002), to appear.

[17] S. Moriguchi, and A. Shioura, On Hochbaum's scaling algorithm for the general resource allocation problem, Research Report No. B-377, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, 2002.

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

[19] K. Murota, Discrete convex analysis, Math. Prog. 83 (1998) 313–371.

[20] K. Murota, Discrete Convex Analysis: an Introduction (Kyoritsu Publ. Co., Tokyo, 2001). [In Japanese]

[21] K. Murota, Discrete Convex Analysis, to be published.

[22] K. Murota and A. Shioura, M-convex function on generalized polymatroid, Math. Oper. Res. 24 (1999) 95–105.

[23] K. Murota and A. Shioura, Quasi M-convex and L-convex functions: quasi-convexity in discrete optimization, Discrete Appl. Math., to appear.

[24] R. T. Rockafellar, Network Flows and Monotropic Optimization (Wiley, New York, 1984).

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

[26] A. Shioura, Minimization of an M-convex function, Discrete Appl. Math. 84 (1998) 215–220.

[27] A. Tamura, Coordinatewise domain scaling algorithm for M-convex function minimization, in: Proceedings of the Ninth Conference on Integer Programming and Combinatorial Optimization, Lecture Notes in Computer Science (Springer-Verlag, Berlin, 2002), to appear.