論文 / 著書情報
Article / Book Information

題目(和文)	M凸関数最小化および関連問題に対するアルゴリズムと解析
Title(English)	Algorithms and Analysis of M-Convex Function Minimization and Related Problems
著者(和文)	南川智都
Author(English)	Norito Minamikawa
出典(和文)	学位:博士（工学）, 学位授与機関:東京工業大学, 報告番号:甲第11944号, 授与年月日:2021年3月26日, 学位の種別:課程博士, 審査員:塩浦 昭義,水野 眞治,松井 知己,大和 毅彦,中田 和秀
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第11944号, Conferred date:2021/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Algorithms and Analysis of M-Convex Function Minimization and Related Problems

Norito MINAMIKAWA

Supervisor: Professor Akiyoshi Shioura

Submitted in partial fulfillment of
the requirement for the degree of
DOCTOR OF ENGINEERING

Department of Industrial Engineering and Economics
Tokyo Institute of Technology
Oh-okayama 2-12-1, Meguro-ku
Tokyo 152-8550, Japan

February, 2021

Abstract

We discuss M-convex function minimization and related problems. The concept of M-convex function is introduced in Murota (1996) as a class of discrete convex functions, which is defined by extracting the well-solvability of discrete optimization problems. It is shown that M-convex functions have various nice properties as discrete convexity, which makes it possible to develop various efficient minimization algorithms. In this thesis, we aim to contribute to advance the research on M-convex function minimization and related problems theoretically and practically.

We first analyze the basic steepest descent algorithm for M-convex function minimization. It is known that a minimizer of an M-convex function can be found by a steepest descent algorithm in a finite number of iterations. Recently, the exact number of iterations required by a basic steepest descent algorithm was obtained. Furthermore, it was shown that the trajectory of the solutions generated by the basic steepest descent algorithm is a geodesic between the initial solution and the nearest minimizer. In this thesis, we give a simpler and shorter proof of this result by refining the "minimizer cut property," which states that any non-minimizer of an M-convex function can be separated by a hyperplane from a minimizer. We also consider the minimization of a jump M-convex function, which is a generalization of M-convex function, and analyze the number of iterations required by the basic steepest descent algorithm. In particular, we show that the trajectory of the solutions generated by the algorithm is a geodesic between the initial solution and the nearest minimizer.

We then investigate a new class of the separable convex resource allocation problem, and show that it can be formulated as M-convex function minimization. Separable convex resource allocation problem aims at finding an allocation of a discrete resource to several activities that minimizes a separable convex function representing the total cost or the total loss. In this thesis, we consider the separable convex resource allocation problem with an additional constraint that the L_1-distance between a given vector and a feasible solution is bounded by a given positive constant. This problem is motivated by re-allocation problem of dock capacity in a bike sharing system. We prove that the simple separable convex resource allocation problem with the L_1-distance constraint can be reformulated as a submodular resource allocation problem, which is a special case of M-convex function minimization. Furthermore, we present specialized implementations of the existing algorithms and analyze their running time.

Acknowledgment

First of all, I would like to show my greatest appreciation to Prof. Akiyoshi Shioura, my supervisor. During my master's course and Ph.D. course, he supported me all the time when I needed. My research was greatly supported by his deep insight and wide perspective. I would not be able to write this thesis without his help and guidance.

I am also very grateful to the committee members, Prof. Shinji Mizuno, Prof. Takehiko Yamato, Prof. Tomomi Matsui, and Associate Prof. Kazuhide Nakata for reviewing and evaluating my thesis.

I am also thankful to all the present and past members of Shioura Laboratory and Matsui Laboratory for their helpful comments in numerous seminars.

Finally, I am very grateful to my family. With warm words, they always supported and encouraged me to pursue my research as I wanted.

Contents

Chapter 1

Introduction

M-convexity is a convexity concept in discrete optimization problems. In this thesis, we discuss M-convex function minimization and related problems.

An optimization problem aims at finding an optimal solution that minimizes (or maximizes) the value of an objective function under given constraints. In particular, an optimization problem is called a discrete optimization problem if the set of feasible solutions is given as a discrete set. Various real world problems can be formulated as a discrete optimization problem. Due to its high versatility, discrete optimization problems have a wide range of applications, such as operations research, computer science, system engineering, and industrial engineering.

An optimal solution of a discrete optimization can be found by exhaustive search of all feasible solutions. This approach, however, is not realistic since it requires quite a long running time. Therefore, it is important to develop algorithms for solving discrete optimization problems in reasonable running time. Indeed, there is a large amount of literature on exact algorithms that find an optimal solution of a given problem within a few hours by removing unnecessary computation in exhaustive search, and on approximation algorithms that find a high-quality sub-optimal solution in a short time instead of an optimal solution.

While it is hard in most of discrete optimization problems to obtain optimal solutions within a reasonable time, there exist some problems that are known to have efficient algorithms for finding solutions quickly. We present some examples of such "well-solved" discrete optimization problems.

The first example of a well-solved discrete optimization problem is the minimum spanning tree problem. Given an undirected graph with edge weight, the problem aims at finding a spanning tree of the graph with the minimum total edge weight. It is well known that this problem can be solved by greedy algorithms such as Kruskal's and Prim's algorithms [22, 42].

The second example is the simple resource allocation problem. In this problem, we are given a family of univariate convex functions f_i $(i = 1, 2, \ldots, n)$ and a positive integer r. The goal of this problem is to find a non-negative integral vector $x \geq 0$ that minimizes the value $\sum_{i=1}^{n} f_i(x(i))$ under the constraint $\sum_{i=1}^{n} x(i) = r$. This problem can be solved by a simple greedy algorithm, which iteratively increments some component of vector x so that the increase in the value $\sum_{i=1}^{n} f_i(x(i))$ is the minimum [13].

While these "well-solved" discrete optimization problems are useful in their own right, they can also be used to find optimal solutions and approximate solutions, as explained

below.

It is know that the branch-and-bound method, which is an efficient enumeration algorithm, is effective in finding an optimal solution of a difficult problem. The algorithm finds an optimal solution of a given problem by decomposing the problem into subproblems recursively and making use of the information obtained from the subproblems such as lower and upper bounds of the optimal values. Therefore, for the speed-up of a branch-and-bound algorithm, it is important to decompose and/or relax a given difficult problem into "well-solved" (sub)problems in an appropriate manner and to develop an algorithm to solve the "well-solved" subproblems quickly.

We can also utilize "well-solved" discrete optimization problems to find approximate solutions of difficult discrete optimization problems. For example, for the traveling salesman problem, which is a typical example of difficult discrete optimization problems, there has been proposed a 2-approximation algorithm that uses the minimum spanning tree problem, a typical example of "well-solved" discrete optimization problems explained above.

As explained above, it is important to better understand well-solved discrete optimization problems to provide efficient algorithms not only for well-solved problems but also for difficult problems. As a unified framework for well-solved discrete optimization problems, the theory of discrete convex analysis is developed in Murota [30, 31]. In discrete convex analysis, the concept of M-convexity for functions defined over the integer lattice points plays a central role. M-convexity is defined by extracting the well-solvability of discrete optimization problems. Various examples of M-convex functions can be found in discrete optimization problems; indeed, the two examples of well-solved discrete optimization problems mentioned above can be reduced to M-convex function minimization. M-convex functions enjoy various nice properties such as extensibility to ordinary convex functions, conjugacy, duality, etc. [34]. In this thesis, we consider the minimization of an M-convex function, which is one of the most fundamental optimization problems in discrete convex analysis. M-convex functions have various nice properties as discrete convexity, which makes it possible to develop various efficient minimization algorithms such as steepest descent algorithms and scaling algorithms. This fact shows that if a discrete optimization problem can be reduced to M-convex function minimization, it can be regarded as a well-solved problem.

The main aim of the thesis is to better understand the well-solved discrete optimization problems through (i) the analysis of the algorithms for well-solved problems and (ii) the extension of the class of well-solved problems. In particular, we present the following results:

- Refined analysis of the steepest descent algorithms for M-convex function minimization and a generalized problem.
- A new class of resource allocation problems that can be formulated as M-convex function minimization.

Details of the results in this thesis are described as follows.

In Chapter 3, we analyze the basic steepest descent algorithm for M-convex function minimization. It is known that a minimizer of an M-convex function can be found by a steepest descent algorithm in a finite number of iterations. Recently, the exact number of iterations required by a basic steepest descent algorithm was obtained [45]. Furthermore, it was shown that the trajectory of the solutions generated by the basic steepest descent algorithm is a geodesic between the initial solution and the nearest minimizer [45]. In this

chapter, we give a simpler and shorter proof of this result by refining the "minimizer cut property," which states that any non-minimizer of an M-convex function can be separated by a hyperplane from a minimizer. We also consider the minimization of a jump M-convex function, which is a generalization of M-convex function, and derive a tight bound for the number of iterations required by the basic steepest descent algorithm. It should be noted that this tight bound is not known so far for jump M-convex function minimization. In particular, we show that the trajectory of the solutions generated by the algorithm is a geodesic between the initial solution and the nearest minimizer. The results obtained in this chapter provide us a better understanding of the behavior of steepest descent algorithms, which may lead to a faster algorithm for M-convex function minimization. The contents of this chapter is presented in the paper [25].

In Chapter 4, we investigate a new class of the separable convex resource allocation problem, and show that it can be formulated as M-convex function minimization. Separable convex resource allocation problem aims at finding an allocation of a discrete resource to several activities that minimizes a separable convex function representing the total cost or the total loss. In this chapter, we consider the separable convex resource allocation problem with an additional constraint that the L_1-distance between a given vector and a feasible solution is bounded by a given positive constant. This problem is motivated by re-allocation problem of dock capacity in a bike sharing system [10]. We prove that the simple separable convex resource allocation problem with the L_1-distance constraint can be reformulated as a submodular resource allocation problem, which is a special case of M-convex function minimization. This means that the problem is a well-solved discrete optimization problem, i.e., it admits an efficient algorithm for finding an optimal solution. Furthermore, we present specialized implementations of the existing algorithms and analyze their running time. The contents of this chapter is presented in the paper [24].

This thesis is organized as follows. In Chapter 2, we introduce necessary concepts and notations, and show elementary key properties. In Chapter 3, we consider minimization problems of two types of discrete convex functions. In Chapter 4, we discuss the simple resource allocation problem with the L_1-distance constraint. Finally, Chapter 5 concludes this thesis.

Chapter 2

Preliminaries

2.1 Notation

Let n be a positive integer with $n \geq 2$, and put $N = \{1, 2, \ldots, n\}$. We denote by $\mathbb{R}$ the set of real numbers, and by $\mathbb{Z}$ (resp., by $\mathbb{Z}_+$) the sets of integers (resp., non-negative integers).

We denote by $x(i)$ the ith entry of a vector, i.e.,

$$x = (x(1), x(2), \ldots, x(n)).$$

We also denote by $\mathbf{0}$ the zero vector in $\mathbb{R}^n$.

For $S \subseteq N$, we denote by $\chi_S \in \{0, 1\}^n$ the characteristic vector of S, i.e.,

$$\chi_S(i) = \begin{cases} 1 & (\text{if } i \in S), \\ 0 & (\text{otherwise}). \end{cases}$$

In particular, we denote $\chi_i = \chi_{\{i\}}$ for every $i \in N$. We also denote $\chi_0 = \mathbf{0}$. For vectors $x, y \in \mathbb{R}^n$, we denote $x \leq y$ if $x(i) \leq y(i)$ for all $i \in N$, and also denote $x < y$ if $x(i) < y(i)$ for all $i \in N$. For $x \in \mathbb{R}^n$ and $S \subseteq N$, we define

$$x(S) = \sum_{i \in S} x(i),$$

where $x(\emptyset) = 0$ is assumed by convention.

For $x = (x(i) \mid i \in N) \in \mathbb{R}^n$, the l_1 norm and the l_∞ norm of x are defined as

$$\|x\|_1 = \sum_{i \in N} |x(i)|, \quad \|x\|_\infty = \max_{i \in N} |x(i)|.$$

The support of a vector x is denoted by $\mathrm{supp}(x)$, i.e.,

$$\mathrm{supp}(x) = \{i \in N \mid x(i) \neq 0\},$$

and the positive and negative supports of x are denoted by $\mathrm{supp}^+(x)$ and by $\mathrm{supp}^-(x)$, respectively, i.e.,

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

Let $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be a function. The effective domain of f is defined as

$$\mathrm{dom} f = \{x \in \mathbb{Z}^n \mid f(x) < +\infty\}.$$

A vector $x \in \mathrm{dom} f$ is called a minimizer (an optimal solution) if it satisfies

$$f(x) \leq f(y) \quad (\forall y \in \mathbb{Z}^n).$$

We denote the set of minimizers of f by $\arg\min f$, i.e.,

$$\arg\min f = \{x \in \mathbb{Z}^n \mid f(x) \leq f(y)\ (\forall y \in \mathbb{Z}^n)\}.$$

2.2 Definition of M-convex function

The concept of M-convex function is introduced in Murota [30, 31, 34] as a generalization of the concept of valuated matroid. M-convex functions enjoy various nice properties as "discrete convexity" such as extensibility to ordinary convex functions, conjugacy, duality, etc.

A function $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ is said to be M-convex if it satisfies (M-EXC):

(M-EXC) $\forall x, y \in \mathrm{dom} f, \forall i \in \mathrm{supp}^+(x - y), \exists j \in \mathrm{supp}^-(x - y)$:

$$f(x) + f(y) \geq f(x - \chi_i + \chi_j) + f(y + \chi_i - \chi_j). \tag{2.1}$$

Note that the effective domain of an M-convex function f lies on a hyperplane $\sum_{i=1}^n x(i) = r$ of a constant component sum, i.e., $x(N) = y(N)$ holds for every $x, y \in \mathrm{dom} f$.

Below we present examples of M-convex functions that arise in discrete optimization problems.

Example 2.1. (Minimum Spanning Tree Problem) [32]

Let $G = (V, E)$ be an undirected graph where each edge $e \in E$ has weight $w(e)$. The minimum spanning tree problem on the graph G can be formulated as the minimization of a function $f_{\mathrm{MST}} : \mathbb{Z}^E \to \mathbb{R} \cup \{+\infty\}$ given by

$$f_{\mathrm{MST}}(x) = \begin{cases} \sum_{e \in F} w(e) & (\text{if } x = \chi_F \text{ for } F \subseteq E \text{ such that } T = (V, F) \text{ is a spanning tree}), \\ +\infty & (\text{otherwise}). \end{cases}$$

Then, f_{MST} is an M-convex function.

Example 2.2. (Simple Resource Allocation Problem) [18, 44]

Given a positive integer r and univariate convex functions f_i $(i = 1, 2, \ldots, n)$, the resource allocation problem can be formulated as the minimization of a function $f_{\mathrm{SRA}} : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ given by

$$f_{\mathrm{SRA}}(x) = \begin{cases} \sum_{i=1}^n f_i(x(i)) & (\text{if } x \geq 0 \text{ and } \sum_{i=1}^n x(i) = r), \\ +\infty & (\text{otherwise}). \end{cases}$$

The function f_{SRA} is an M-convex function.

With an M-convex function $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$, the minimization problem (MC) is formulated as follows:

(MC) Minimize $f(x)$ subject to $x \in \mathrm{dom} f$.

The two examples mentioned above show that the minimum spanning tree problem and the simple resource allocation problem can be formulated as (MC). The problem (MC) has been discussed in the literature and various algorithms have been proposed [26, 33, 34, 35, 43, 44, 48]. We review algorithms for (MC) in detail in Chapter 3.

2.3 Definition of Jump M-convex function

In this thesis, we deal with a minimization problem that is related to the concept of jump system. The concept of jump system, introduced by Bouchet and Cunningham [7], is defined as a set of integer lattice points with a certain exchange property, and is a generalization of matroid [23], delta-matroid [6], integral base polyhedron [11], and integral bisubmodular polyhedron [7]. A jump system J is called a constant-parity jump system if the parity of the component sum of each vector in J is constant.

The concept of jump M-convex function[1] is introduced by Murota [36] as a discrete convex function defined on a constant-parity jump system. A function $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ is said to be jump M-convex if it satisfies (JM-EXC):

(JM-EXC) $\forall x, y \in \mathrm{dom} f, \forall s \in \mathrm{inc}(x, y), \exists t \in \mathrm{inc}(x + s, y)$:

$$f(x) + f(y) \geq f(x + s + t) + f(y - s - t), \tag{2.2}$$

where

$$U = \{+\chi_i, -\chi_i \mid i \in N\}, \quad \mathrm{inc}(x, y) = \{s \in U \mid \|(x + s) - y\|_1 = \|x - y\|_1 - 1\}. \tag{2.3}$$

Jump M-convex functions contain M-convex functions and valuated delta-matroids as special cases. An M-convex function is precisely a jump M-convex function such that the effective domain lies on a hyperplane with a constant component sum. A valuated delta-matroid is precisely a jump M-concave function such that the effective domain is included in the set of $\{0, 1\}$-vectors.

Below we present a typical example of jump M-convex functions that arises in discrete optimization problems.

Example 2.3. (Minsquare graph factor problem) [3, 4, 36]

Suppose that we are given an undirected graph $G = (V, E)$ that may contain loops and parallel edges and a positive integer k. In the minsquare graph factor problem, we aim at finding a subgraph H with exactly k edges that minimizes $\sum_{v \in V} x_H(v)^2$, where $x_H(v)$ is the degree of the vertex v in the subgraph (i.e., the number of edges in H incident to v).

We may also consider a variant of the minsquare graph factor problem, where we are given integers $d(v)$ for $v \in V$, and find a subgraph H of G (without any constraint on the number

[1] This concept is introduced in [36] under the name of "M-convex function on a jump system." The alternative name "jump M-convex function" is coined in [37] (see also [38]).

of edges in H) that minimizes $\sum_{v\in V}\{x_H(v)-d(v)\}^2$. Such a variant can be formulated as the minimization of a function $f_{\mathrm{MGF}}:\mathbb{Z}^V\to\mathbb{R}\cup\{+\infty\}$ given by

$$f_{\mathrm{MGF}}(x)=\begin{cases}\sum_{v\in V}\{x_H(v)-d(v)\}^2 & (\text{if there exists a subgraph } H \text{ such that}\\ & \quad \text{the degree of } v \text{ equals to } x_H(v) \text{ for each } v\in V),\\ +\infty & (\text{otherwise}).\end{cases}$$

Then, f_{MGF} is a jump M-convex function.

With a jump M-convex function $f:\mathbb{Z}^n\to\mathbb{R}\cup\{+\infty\}$, the minimization problem (JMC) is formulated as follows:

(JMC) Minimize $f(x)$ subject to $x\in \mathrm{dom} f$.

Example 2.3 shows that the variant of the minsquare graph factor problem can be formulated as (JMC). In addition, it is known that the problem (JMC) has various examples [5, 20, 21, 47]. The problem (JMC) has been discussed in the literature and various algorithms have been proposed [36, 41, 46]. We review algorithms for (JMC) in detail in Chapter 3.

2.4 Definition of Resource Allocation Problem

Resource allocation problem is a problem of finding an optimal allocation of some discrete resources to several activities, in the setting where cost (or loss) occurs according to resource allocation and total cost should be minimized. Investigation of resource allocation problem is initiated by Koopman [19] in 1953, and then various research has been done on the topic for more than fifty years. Resource allocation problem has many variations, which leads to a wide range of applications such as investment planning, manpower planning, production planning and optimal armaments planning.

The simplest resource allocation problem is the following one with a separable convex objective function and a simple constraint on the total resource to be allocated:

$$\begin{aligned}\text{Minimize}\quad & \sum_{i=1}^{n} f_i(x(i))\\ \text{subject to}\quad & \sum_{i=1}^{n} x(i)=r,\\ & x\in\mathbb{Z}_+^n,\end{aligned}$$

where r is a positive integer and $f_i:\mathbb{Z}\to\mathbb{R}\cup\{+\infty\}$ is a convex function ($i=1,\ldots,n$). Here, we say that a function $f_i:\mathbb{Z}\to\mathbb{R}\cup\{+\infty\}$ is *convex* if it satisfies $f_i(x)<+\infty$ for some $x\in\mathbb{Z}$ and

$$f_i(x-1)+f_i(x+1)\geq 2f_i(x)\quad(\forall x\in\mathbb{Z}).$$

This problem is referred to as the *simple resource allocation problem* (see, e.g., [16, 18]). We assume that each function f_i is defined explicitly, or given by a function evaluation oracle that, given $x(i)$, returns the value $f_i(x(i))$ in constant time.

It is well known that the simple resource allocation problem can be solved by a greedy algorithm [13], which runs in $\mathrm{O}(r\log n)$ time. Note that the running time of the greedy

algorithm is exponential in the input size of the problem, which is given as the sum of the dimension of this problem n and the bit size of the number of resource $\log r$. This means that the running time becomes huge for a problem with a large input size. Galil and Megiddo [12] and Katoh et al. [17] independently developed polynomial-time algorithms for the simple resource allocation problem, both of which run in $O(n(\log r)^2)$ time. The fastest algorithm so far for the simple resource allocation problem is due to Frederickson and Johnson [9], which runs in $O(n \log(r/n))$ time. It is known that this time complexity $O(n \log(r/n))$ is the best possible under a certain standard computation model [15].

While the simple resource allocation problem is the most basic resource allocation problem, more general resource allocation problems with generalized upper bound constraint, nested constraint, tree constraint, and network constraint have been discussed in the literature [16, 18]. Polymatroid constraint is a common generalization of the constraints mentioned above, and the resource allocation problem with the polymatroid constraint is called the submodular resource allocation problem. A greedy algorithm is also applicable to the submodular resource allocation problem [8], and a polynomial-time scaling algorithm is proposed by Hochbaum [15] (see also Moriguchi and Shioura [27]).

Chapter 3

Time Bounds of Basic Steepest Descent Algorithms for M-convex function Minimization and Related Problems

3.1 Introduction

In this chapter, we consider the minimization of a function $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ defined on integer lattice points. For such a problem, the steepest descent algorithm is known as a fundamental approach [33, 35, 40]. A basic form of the steepest descent algorithm, which is referred to as the *basic steepest descent algorithm* in this chapter, is described as follows, where $D \subseteq \mathbb{Z}^n$ is a set of possible directions:

Step 0: Choose an initial solution $x_0 \in \mathbb{Z}^n$ with $f(x_0) < +\infty$ arbitrarily and set $x := x_0$.
Step 1: If the current solution x is a local minimum in the sense that

$$f(x) \leq f(x+d) \qquad (\forall d \in D),$$

then output the current solution x and stop.
Step 2: Find a direction $d \in D$ with the smallest value of $f(x+d)$.
Step 3: Set $x := x + d$, and go to Step 1.

Note that the output of the algorithm is a minimizer of function f if the local minimality guarantees the global minimality for f.

In this chapter, we focus on the two classes of discrete convex functions, called M-convex functions and jump M-convex functions, for which the minimization problem can be solved by the basic steepest descent algorithm with an appropriately chosen set D of directions. The main aim of this chapter is to better understand the behavior of the basic steepest descent algorithms for the two classes of discrete convex functions.

3.1.1 Minimization of M-convex function

We first consider the problem (MC) of minimizing an M-convex function [26, 33, 34, 35, 43, 44, 48]. It is known [31] that a global minimizer of an M-convex function can be characterized by the local minimality with respect to the direction set $D = D_{\mathrm{MC}}$ given by

$$D_{\mathrm{MC}} = \{d \in \{0, \pm 1\}^n \mid \|d\|_1 \in \{0, 2\},\ \sum_{i=1}^{n} d(i) = 0\} = \{-\chi_i + \chi_j \mid i, j \in \{1, 2, \ldots, n\}\} \quad (3.1)$$

(see Theorem 3.1), where $\chi_i \in \{0, 1\}^n$ denotes the i-th characteristic vector for $i = 1, 2, \ldots, n$. Based on this property, the problem (MC) can be solved by the basic steepest descent algorithm with $D = D_{\mathrm{MC}}$ (see, e.g., [31, 43]).

A number of modified variants of the basic steepest descent algorithm have been proposed to ensure finite termination with guaranteed time bounds for M-convex functions under additional assumptions. These variants, to be described in Section 3.2.1, are all based on a key property, which is referred to as the "minimizer cut property" in [43]; the minimizer cut property states that any non-minimizer of an M-convex function can be separated by a hyperplane from a minimizer (see Theorem 3.2).

While refinements and improvements have been proposed for the steepest descent algorithm, no guaranteed bound has been obtained for the basic steepest descent algorithm before a tight bound is given in Shioura [45]. It is shown in [45] that the number of iterations in the basic steepest descent algorithm is exactly equal to the half of the L1-distance between the initial vector x_0 and a minimizer nearest to x_0. Furthermore, it is also shown that the trajectory of the solutions generated by the basic steepest descent algorithm is a geodesic (i.e., a "shortest" path) between the initial solution and the nearest minimizer. This result is obtained in [45] as a corollary of some result on a constrained optimization problem with an M-convex objective function, and accordingly, the proof in [45] is rather long.

In this chapter, we give an alternative, simple and short proof of the tight bound for the number of iterations in the basic steepest descent algorithm. Our proof is based on a stronger variant of the minimizer cut property (Theorem 3.5), which states that any non-minimizer and a *nearest* minimizer can be separated by a hyperplane. An outline of the alternative proof is presented in Section 3.2.2, while a detailed proof is given in Section 3.4.1.

3.1.2 Minimization of jump M-convex function

As a generalization of M-convex function minimization (MC), we discuss the problem (JMC) of minimizing a jump M-convex function. It is known [36] that a global minimizer of a jump M-convex function can be characterized by the local minimality with respect to the direction set $D = D_{\mathrm{JMC}}$ given by

$$D_{\mathrm{JMC}} = \{d \in \mathbb{Z}^n \mid \|d\|_1 \in \{0, 2\}\} = \{\pm\chi_i \pm \chi_j \mid i, j \in \{1, 2, \ldots, n\}\} \quad (3.2)$$

(see Theorem 3.6). Based on this property, the problem (JMC) can be solved by the basic steepest descent algorithm with $D = D_{\mathrm{JMC}}$.

Similarly to the case of M-convex functions, a variant of the minimizer cut property holds for jump M-convex functions [41, 46]. Based on this property, Murota and Tanaka

[41] proposed a modified version of the steepest descent algorithm (see Section 3.3.1), and analyzed the number of iterations required by the algorithm. On the other hand, no non-trivial time bound is known for the basic steepest descent algorithm.

In this chapter, we provide the exact bound on the number of iterations required by the basic steepest descent algorithm (Theorem 3.9), and show that the trajectory of the solutions generated by the algorithm is a geodesic between the initial solution and the nearest minimizer. Similarly to the case of the problem (MC), our proof is based on a stronger variant of the minimizer cut property (Theorem 3.10). An outline of the analysis for the time bound is presented in Section 3.3.2, and a proof of Theorem 3.10 is given in Section 3.4.2. The proof of Theorem 3.10 for the problem (JMC) is based on the case-by-case analysis and can be done in a similar way as that of Theorem 3.5 for the problem (MC), while the proof of Theorem 3.10 needs to deal with more cases since the set of possible directions D_{JMC} for the problem (JMC) is larger than that of the problem (MC).

3.2 M-convex function minimization

We first consider the problem (MC) of minimizing an M-convex function.

3.2.1 Review of Time Bounds for Steepest Descent Algorithms

We review the previous results on time bounds of the basic steepest descent algorithm and its variants. A global minimizer of an M-convex function can be characterized by a local minimality:

Theorem 3.1 ([31]). *For an M-convex function $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$, a vector $x^* \in \mathrm{dom} f$ is a minimizer of f if and only if $f(x^*) \leq f(x^* - \chi_i + \chi_j)$ for all $i, j \in N$.*

With the direction set D_{MC} defined by (3.1), this property can be rewritten as follows:

$$x^* \text{ is a minimizer} \iff f(x^*) \leq f(x^* + d) \qquad (\forall d \in D_{\mathrm{MC}}).$$

Due to this property, the problem (MC) can be solved by the following algorithm [31, 43], which is nothing but the basic steepest descent algorithm in Section 3.1 with $D = D_{\mathrm{MC}}$. We assume that the initial solution $x_0 \in \mathrm{dom} f$ is given in advance.

Algorithm Steepest_Descent_MC
Step 0: Set $x := x_0$.
Step 1: If $f(x) \leq f(x - \chi_i + \chi_j)$ for every $i, j \in N$, then output x and stop.
Step 2: Find $i, j \in N$ that minimize $f(x - \chi_i + \chi_j)$.
Step 3: Set $x := x + \chi_i - \chi_j$ and go to Step 1.

It is easy to see that the algorithm terminates in a finite number of iterations if the effective domain $\mathrm{dom} f$ is bounded or f is integer-valued.

The following property of an M-convex function shows that information obtained from a steepest descent direction can be used to restrict the region containing a global minimizer.

Theorem 3.2 (minimizer cut property (cf. [43, Corollary 2.3])). *Let $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be an M-convex function with $\arg\min f \neq \emptyset$, and $x \in \mathrm{dom} f$ be a vector that is not a minimizer of f, i.e., $x \notin \arg\min f$. Then, for every $i, j \in N$ minimizing the value $f(x - \chi_i + \chi_j)$, there exists some $x^* \in \arg\min f$ such that $x^*(i) \leq x(i) - 1$ and $x^*(j) \geq x(j) + 1$.*

On the basis of Theorem 3.2, the exact bound on the number of iterations required by the algorithm can be obtained in the special case where f has a unique minimizer. Recall that x_0 is the initial vector in the algorithm Steepest_Descent_MC.

Proposition 3.3 (cf. [35, Lemma 2.2]). *Suppose that the algorithm* Steepest_Descent_MC *is applied to an M-convex function $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ with a unique minimizer x^*. Then, the algorithm terminates in $(1/2)\|x^* - x_0\|_1$ iterations.*

For a general M-convex function f, which may have multiple minimizers, a variant of the steepest descent algorithm is proposed in Murota [35] with the use of a perturbation of f that makes Proposition 3.3 applicable. It is shown [35] that the algorithm Steepest_Descent_MC applied to the perturbed function amounts to a variant of the steepest descent algorithm for the original function f using a certain tie-breaking rule in the choice of $d \in D_{\mathrm{MC}}$ in Step 2, and the resulting algorithm runs in at most

$$(1/2) \max\{\|x^* - x_0\|_1 \mid x^* \in \arg\min f\}$$

iterations if $\mathrm{dom} f$ is bounded.

Another approach for obtaining a guaranteed time bound is to maintain an interval $[l, u]$ with $l, u \in \mathbb{Z}^n$ containing a minimizer, as in [26]. In each iteration, this modified steepest descent algorithm selects a direction $d = -\chi_i + \chi_j \in D_{\mathrm{MC}}$ minimizing the function value $f(x + d)$ under the constraint $x + d \in [l, u]$. Theorem 3.2 implies that there exists some minimizer x^* of f satisfying $x^*(i) \leq x(i) - 1$ and $x^*(j) \geq x(j) + 1$, which makes it possible to update the interval $[l, u]$ appropriately. Note that if $l = u$ then u must be a minimizer, and the algorithm stops. Based on this approach, the bound

$$(1/2)n \max\{\|x - y\|_\infty \mid x, y \in \mathrm{dom} f\}$$

can be obtained for the number of iterations in the modified steepest descent algorithm.

On the other hand, no guaranteed bound has been obtained for the basic steepest descent algorithm before the following result is shown in [45]. For $x \in \mathbb{Z}^n$, we define

$$\mu(x) = \min\{\|x^* - x\|_1 \mid x^* \in \arg\min f\}; \tag{3.3}$$

$\mu(x) \in \mathbb{Z}_+$ is the L_1-distance between the vector x and a minimizer nearest to x.

Theorem 3.4 ([45, Corollary A.5]). *For an M-convex function $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ with $\arg\min f \neq \emptyset$, the algorithm* Steepest_Descent_MC *terminates in a finite number of iterations, and the number of iterations is exactly equal to $\mu(x_0)/2$.*

It follows from Theorem 3.4 that the trajectory of the solutions generated by the algorithm Steepest_Descent_MC is a geodesic between the initial solution and the nearest minimizer.

3.2.2 Alternative Proof for Exact Time Bound

We will provide an alternative proof of Theorem 3.4 that is simpler and shorter than the existing one in [45]. Our proof is based on the following property of a steepest descent direction, stating that after moving one step in a steepest descent direction, the L_1-distance to a nearest minimizer reduces by two. For $x \in \mathbb{Z}^n$, we define

$$M^*(x) = \{x^* \in \mathbb{Z}^n \mid x^* \in \arg\min f, \|x^* - x\|_1 = \mu(x)\}, \tag{3.4}$$

which is the set of the minimizers of f nearest to x.

Theorem 3.5. *Let $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be an M-convex function with $\arg\min f \neq \emptyset$, and let $x \in \mathrm{dom} f$ be a vector that is not a minimizer of f, i.e., $x \notin \arg\min f$. Also, let i, j be a pair of elements in N minimizing the value $f(x - \chi_i + \chi_j)$.*
(i) *There exists some $x^* \in M^*(x)$ such that $x^*(i) \leq x(i) - 1$ and $x^*(j) \geq x(j) + 1$.*
(ii) *$\mu(x - \chi_i + \chi_j) = \mu(x) - 2$ holds.*
(iii) *It holds that $M^*(x - \chi_i + \chi_j) = \{x \in M^*(x) \mid x^*(i) \leq x(i) - 1, x^*(j) \geq x(j) + 1\}$. In particular, the vector x^* in* (i) *is contained in $M^*(x - \chi_i + \chi_j)$.*

It should be noted that the elements i and j in Theorem 3.5 are distinct by Theorem 3.1 since x is not a minimizer of f.

The claim (i) in Theorem 3.5 is a stronger variant of Theorem 3.2 (minimizer cut property); the main difference is that x^* is a minimizer with the minimum L_1-distance from the vector x in the claim (i), while such a restriction is not imposed in Theorem 3.2. A proof of the claim (i) can be obtained by a modification of the proof for Theorem 3.2; we give a detailed proof for completeness in Section 3.4.1. Proofs of the claims (ii) and (iii) are given later in this section.

It is easy to see that Theorem 3.4 follows immediately from Theorem 3.5 (ii) since the L_1-distance $\mu(x)$ between the current vector x and a nearest minimizer reduces by two in each iteration of the algorithm Steepest_Descent_MC. Theorem 3.5 (ii) also implies that the trajectory of the solutions generated by the algorithm is a geodesic between the initial solution and the nearest minimizer.

We now prove the claims (ii) and (iii) assuming (i).

Proof of the claims (ii) *and* (iii). We denote

$$\widetilde{M} = \{x^* \in M^*(x) \mid x^*(i) \leq x(i) - 1,\ x^*(j) \geq x(j) + 1\}.$$

We see that the vector x^* in the claim (i) is contained in $\widetilde{M}$, i.e., $\widetilde{M} \neq \emptyset$.

It holds that

$$\begin{aligned} \forall y^* \in M^*(x - \chi_i + \chi_j): \quad \mu(x - \chi_i + \chi_j) &= \|y^* - (x - \chi_i + \chi_j)\|_1 \\ &\geq \|y^* - x\|_1 - \|\chi_i - \chi_j\|_1 \\ &= \|y^* - x\|_1 - 2 \geq \mu(x) - 2, \end{aligned} \tag{3.5}$$

where the first inequality is by the triangle inequality. We also have

$$\forall x^* \in \widetilde{M}: \quad \mu(x - \chi_i + \chi_j) \leq \|x^* - (x - \chi_i + \chi_j)\|_1 = \|x^* - x\|_1 - 2 = \mu(x) - 2, \tag{3.6}$$

where the first equality is by the inequalities $x^*(i) \leq x(i) - 1,\ x^*(j) \geq x(j) + 1$ and the second equality is by $x^* \in M^*(x)$. It follows from (3.5) and (3.6) that all the inequalities in (3.5) and (3.6) hold with equality. In particular, we have

$$\forall y^* \in M^*(x - \chi_i + \chi_j): \quad \|y^* - (x - \chi_i + \chi_j)\|_1 = \|y^* - x\|_1 - 2 = \mu(x) - 2, \tag{3.7}$$

$$\forall x^* \in \widetilde{M}: \quad \mu(x - \chi_i + \chi_j) = \|x^* - (x - \chi_i + \chi_j)\|_1 = \mu(x) - 2. \tag{3.8}$$

The claim (ii) follows immediately from (3.8).

The equation (3.7) implies the inclusion $M^*(x - \chi_i + \chi_j) \subseteq \widetilde{M}$. Indeed, for every $y^* \in M^*(x - \chi_i + \chi_j)$, it holds that $y^* \in M^*(x)$ by $\|y^* - x\|_1 = \mu(x)$, and also holds that $y^*(i) \leq x(i) - 1$ and $y^*(j) \geq x(j) + 1$ by $\|y^* - (x - \chi_i + \chi_j)\|_1 = \|y^* - x\|_1 - 2$. Hence, $M^*(x - \chi_i + \chi_j) \subseteq \widetilde{M}$ holds.

The equation (3.8) implies that for every $x^* \in \widetilde{M}$, we have $\|x^* - (x - \chi_i + \chi_j)\|_1 = \mu(x - \chi_i + \chi_j)$, i.e., $x^* \in M^*(x - \chi_i + \chi_j)$ holds. Thus, we have $M^*(x - \chi_i + \chi_j) \supseteq \widetilde{M}$, and the claim (iii) follows. □

3.3 Jump M-convex function minimization

In this section, we consider the problem (JMC) of minimizing a jump M-convex function.

3.3.1 Review of Time Bounds for Steepest Descent Algorithms

We first review the previous results on time bounds of the basic steepest descent algorithm and its variants. For a jump M-convex function, a global minimality is guaranteed by a local minimality in the neighborhood of L1-distance two.

Theorem 3.6 ([36, Theorem 3.3]). *For a jump M-convex function $f: \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$, a vector $x^* \in \mathrm{dom} f$ is a minimizer of f if and only if $f(x^*) \leq f(x + s + t)$ for all $s, t \in U$.*

With the direction set D_{JMC} defined by (3.2), this property can be rewritten as follows:

$$x^* \text{ is a minimizer} \iff f(x^*) \leq f(x^* + d) \qquad (\forall d \in D_{\mathrm{JMC}}).$$

This property naturally suggests the following algorithm [41], which is nothing but the basic steepest descent algorithm in Section 3.1 with $D = D_{\mathrm{JMC}}$. We assume that the initial solution $x_0 \in \mathrm{dom} f$ is given in advance.

Algorithm Steepest_Descent_JMC

Step 0: Set $x := x_0$.

Step 1: If $f(x) \leq f(x + s + t)$ for every $s, t \in U$, then output x and stop.

Step 2: Find $s, t \in U$ that minimize $f(x + s + t)$.

Step 3: Set $x := x + s + t$ and go to Step 1.

It is easy to see that the algorithm terminates in a finite number of iterations if the effective domain $\mathrm{dom} f$ is bounded or f is integer-valued.

The following property, which is a generalization of Theorem 3.2, is a key fact for analysis of the algorithm Steepest_Descent_JMC.

Theorem 3.7 (minimizer cut property ([41, Theorem 4.1])). *Let $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be a jump M-convex function with $\arg\min f \neq \emptyset$, and $x \in \mathrm{dom} f$ be a vector that is not a minimizer of f, i.e., $x \notin \arg\min f$. Also, let $s^*, t^* \in U$ satisfy*

$$f(x + s^* + t^*) = \min\{f(x + s + t) \mid s, t \in U\}.$$

Put $\{i\} = \mathrm{supp}(s^*)$ and $\{j\} = \mathrm{supp}(t^*)$. Then, there exists $x^* \in \arg\min f$ such that

$$\begin{cases} x^*(i) \leq x(i) - 2 & (\text{if } i = j,\ s^* = t^* = -\chi_i), \\ x^*(i) \geq x(i) + 2 & (\text{if } i = j,\ s^* = t^* = +\chi_i), \\ x^*(i) \leq x(i) - 1,\ x^*(j) \leq x(j) - 1 & (\text{if } i \neq j,\ s^* = -\chi_i,\ t^* = -\chi_j), \\ x^*(i) \leq x(i) - 1,\ x^*(j) \geq x(j) + 1 & (\text{if } i \neq j,\ s^* = -\chi_i,\ t^* = +\chi_j), \\ x^*(i) \geq x(i) + 1,\ x^*(j) \leq x(j) - 1 & (\text{if } i \neq j,\ s^* = +\chi_i,\ t^* = -\chi_j), \\ x^*(i) \geq x(i) + 1,\ x^*(j) \geq x(j) + 1 & (\text{if } i \neq j,\ s^* = +\chi_i,\ t^* = +\chi_j). \end{cases}$$

On the basis of Theorem 3.7, the exact bound on the number of iterations required by the algorithm can be obtained in the special case where f has a unique minimizer. Recall that x_0 is the initial vector in the algorithm Steepest_Descent_JMC.

Theorem 3.8 ([41, Theorem 4.2]). *Suppose that the algorithm* Steepest_Descent_JMC *is applied to a jump M-convex function $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ with a unique minimizer x^*. Then, the algorithm terminates in $(1/2)\|x^* - x_0\|_1$ iterations.*

For a general jump M-convex function f, which may have multiple minimizers, a variant of the steepest descent algorithm is proposed in Murota and Tanaka [41] with the use of a perturbation of f that makes Theorem 3.8 applicable. Murota and Tanaka [41] show that the algorithm Steepest_Descent_JMC applied to the perturbed function amounts to a variant of the algorithm for the original function f using a certain tie-breaking rule in the choice of $d \in D_{\mathrm{JMC}}$ in Step 2, and the resulting algorithm runs in at most

$$(1/2) \max\{\|x - y\|_1 \mid x, y \in \mathrm{dom} f\}$$

iterations if $\mathrm{dom} f$ is bounded.

3.3.2 Exact Time Bound for Basic Steepest Descent Algorithm

We analyze the exact number of iterations required by the algorithm Steepest_Des-cent_JMC for a general jump M-convex function f, which may have multiple minimizers. Recall that the definition of the set of the minimizers $M^*(x)$ in (3.4), and the L_1-distance between the vector x and a nearest minimizer $\mu(x)$ in (3.3).

Theorem 3.9. *For a jump M-convex function $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ with $\arg\min f \neq \emptyset$, the algorithm* Steepest_Descent_JMC *terminates in a finite number of iterations, and the number of iterations is exactly equal to $\mu(x_0)/2$.*

Similarly to the case of the problem (MC), our proof is based on the following property of a steepest descent direction, stating that after moving one step in a steepest descent direction, the L_1-distance to a nearest minimizer reduces by two.

Theorem 3.10. *Let $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be a jump M-convex function with $\arg\min f \neq \emptyset$, and $x \in \mathrm{dom} f$ be a vector that is not a minimizer of f, i.e., $x \notin \arg\min f$. Also, let $s^*, t^* \in U$ satisfy*

$$f(x + s^* + t^*) = \min\{f(x + s + t) \mid s, t \in U\}.$$

With the elements $i, j \in N$ satisfying $\{i\} = \mathrm{supp}(s^*)$ and $\{j\} = \mathrm{supp}(t^*)$, define the set $S \subseteq \mathbb{Z}^n$ as

$$S = \begin{cases} \{y \in \mathbb{Z}^n \mid y(i) \leq x(i) - 2\} & (\text{if } i = j,\ s^* = t^* = -\chi_i), \\ \{y \in \mathbb{Z}^n \mid y(i) \geq x(i) + 2\} & (\text{if } i = j,\ s^* = t^* = +\chi_i), \\ \{y \in \mathbb{Z}^n \mid y(i) \leq x(i) - 1,\ y(j) \leq x(j) - 1\} & (\text{if } i \neq j,\ s^* = -\chi_i,\ t^* = -\chi_j), \\ \{y \in \mathbb{Z}^n \mid y(i) \leq x(i) - 1,\ y(j) \geq x(j) + 1\} & (\text{if } i \neq j,\ s^* = -\chi_i,\ t^* = +\chi_j), \\ \{y \in \mathbb{Z}^n \mid y(i) \geq x(i) + 1,\ y(j) \leq x(j) - 1\} & (\text{if } i \neq j,\ s^* = +\chi_i,\ t^* = -\chi_j), \\ \{y \in \mathbb{Z}^n \mid y(i) \geq x(i) + 1,\ y(j) \geq x(j) + 1\} & (\text{if } i \neq j,\ s^* = +\chi_i,\ t^* = +\chi_j). \end{cases} \tag{3.9}$$

Then, the following properties hold.
(i) $M^*(x) \cap S \neq \emptyset$.
(ii) $\mu(x + s^* + t^*) = \mu(x) - 2$.
(iii) $M^*(x + s^* + t^*) = M^*(x) \cap S$.

It should be noted that s^* and t^* in Theorem 3.6 satisfy $s^* + t^* \neq \mathbf{0}$ by Theorem 3.6 since x is not a minimizer of f.

Theorem 3.10 is a generalization of Theorem 3.5 for M-convex functions. Indeed, for a jump M-convex function f such that the effective domain lies on a hyperplane with a constant component sum, Theorem 3.10 coincides with Theorem 3.5.

Note that the claim (i) in Theorem 3.10 is a stronger variant of Theorem 3.7 (minimizer cut property); the main difference is that x^* is a minimizer with the minimum L_1-distance from the vector x in the claim (i), while such a restriction is not imposed in Theorem 3.7. While a proof of the claim (i) can be obtained by a modification of the proof for Theorem 3.7, we give a detailed proof for completeness in Section 3.4.2. Proofs of the claims (ii) and (iii) are given later in this section.

Theorem 3.9 follows immediately from Theorem 3.10 (ii) since the L_1-distance $\mu(x)$ between the current vector x and a nearest minimizer reduces by two in each iteration of the algorithm Steepest_Descent_JMC. Theorem 3.10 (ii) also implies that the trajectory of the solutions generated by the algorithm is a geodesic between the initial solution and the nearest minimizer.

We now give a proof of the claims (ii) and (iii) assuming (i). The proof below is quite similar to that for Theorem 3.5 (ii) and (iii).

Proof of the claims (ii) *and* (iii). We denote

$$\widetilde{M} = M^*(x) \cap S.$$

By the claim (i), we have $\widetilde{M} \neq \emptyset$. It holds that

$$\begin{aligned} \forall y^* \in M^*(x+s^*+t^*): \quad \mu(x+s^*+t^*) &= \|y^* - (x+s^*+t^*)\|_1 \\ &\geq \|y^* - x\|_1 - \|s^*+t^*\|_1 \\ &= \|y^* - x\|_1 - 2 \geq \mu(x) - 2, \end{aligned} \tag{3.10}$$

where the first inequality is by the triangle inequality. We also have

$$\forall x^* \in \widetilde{M}: \quad \mu(x+s^*+t^*) \leq \|x^* - (x+s^*+t^*)\|_1 = \|x^* - x\|_1 - 2 = \mu(x) - 2, \tag{3.11}$$

where the first equality is by $x^* \in S$ and the second equality is by $x^* \in M^*(x)$. It follows from (3.10) and (3.11) that all the inequalities in (3.10) and (3.11) hold with equality. In particular, we have

$$\forall y^* \in M^*(x+s^*+t^*): \quad \|y^* - (x+s^*+t^*)\|_1 = \|y^* - x\|_1 - 2 = \mu(x) - 2, \tag{3.12}$$

$$\forall x^* \in \widetilde{M}: \quad \mu(x+s^*+t^*) = \|x^* - (x+s^*+t^*)\|_1 = \mu(x) - 2. \tag{3.13}$$

The claim (ii) follows immediately from (3.13).

The equation (3.12) implies the inclusion $M^*(x+s^*+t^*) \subseteq \widetilde{M}$. Indeed, for every $y^* \in M^*(x+s^*+t^*)$, it holds that $y^* \in M^*(x)$ by $\|y^* - x\|_1 = \mu(x)$, and also holds that $y^* \in S$ by $\|y^* - (x+s^*+t^*)\|_1 = \|y^* - x\|_1 - 2$. Hence, $M^*(x+s^*+t^*) \subseteq M^*(x) \cap S = \widetilde{M}$ holds.

The equation (3.13) implies that for every $x^* \in \widetilde{M}$, we have $\|x^* - (x+s^*+t^*)\|_1 = \mu(x+s^*+t^*)$, i.e., $x^* \in M^*(x+s^*+t^*)$ holds. Thus, we have $M^*(x+s^*+t^*) \supseteq \widetilde{M}$, and the claim (iii) follows. □

3.4 Proofs

3.4.1 Proof of Theorem 3.5 (i)

Instead of Theorem 3.5 (i), we prove the same statement under a weaker assumption.

Theorem 3.11. *Let $f: \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be an M-convex function with $\arg\min f \neq \emptyset$. Also, let $x \in \mathrm{dom} f$ be a vector, and $i, j \in N$ be elements satisfying the following conditions:*

$$f(x - \chi_i + \chi_j) < f(x), \tag{3.14}$$

$$f(x - \chi_i + \chi_j) \leq f(x - \chi_h + \chi_j) \qquad (\forall h \in N), \tag{3.15}$$

$$f(x - \chi_i + \chi_j) \leq f(x - \chi_i + \chi_k) \qquad (\forall k \in N). \tag{3.16}$$

Then, there exists some $x^* \in M^*(x)$ such that $x^*(i) \leq x(i) - 1$ and $x^*(j) \geq x(j) + 1$.

It is easy to see that the claim (i) in Theorem 3.5 follows immediately from Theorem 3.11 since the elements i, j in Theorem 3.5 satisfy the conditions (3.14)–(3.16).

In the proof we use the following lemmas.

Lemma 3.12. *Let $x, x^* \in \mathbb{Z}^n$ and $i, j \in N$. It holds that $\|(x^* - \chi_i + \chi_j) - x\|_1 \leq \|x^* - x\|_1$ if $x^*(i) > x(i)$ or $x^*(j) < x(j)$ holds.*

Proof. We consider the case $x^*(i) > x(i)$ only the case $x^*(j) < x(j)$ can be proven similarly. Since $x^*(i) > x(i)$, we have $\|(x^* - \chi_i) - x\|_1 - \|x^* - x\|_1 = -1$, from which follows that

$$\begin{aligned}
&\|(x^* - \chi_i + \chi_j) - x\|_1 - \|x^* - x\|_1 \\
&= (\|(x^* - \chi_i + \chi_j) - x\|_1 - \|(x^* - \chi_i) - x\|_1) + (\|(x^* - \chi_i) - x\|_1 - \|x^* - x\|_1) \\
&\leq \|\chi_j\|_1 - 1 = 0,
\end{aligned}$$

where we use the triangle inequality for L_1 norm. □

Lemma 3.13. *Let $x \in \mathrm{dom} f$ and $i, j \in N$ be elements satisfying the conditions* (3.14)*,* (3.15)*, and* (3.16)*. Put $y = x - \chi_i + \chi_j$.*
(i) *For every $x^* \in \arg\min f$ with $i \in \mathrm{supp}^+(x^* - y)$, there exists some $h \in \mathrm{supp}^-(x^* - y) \setminus \{i, j\}$ such that $x^* - \chi_i + x_h \in \arg\min f$.*
(ii) *For every $x^* \in \arg\min f$ with $j \in \mathrm{supp}^-(x^* - y)$, there exists some $k \in \mathrm{supp}^+(x^* - y) \setminus \{i, j\}$ such that $x^* + \chi_j - x_k \in \arg\min f$.*

Proof. We prove (i) only; (ii) can be proven similarly. The condition (M-EXC) applied to x^*, y and $i \in \mathrm{supp}^+(x^* - y)$ implies that

$$f(x^*) + f(y) \geq f(x^* - \chi_i + \chi_h) + f(y + \chi_i - \chi_h) \tag{3.17}$$

for some $h \in \mathrm{supp}^-(x^* - y)$. Note that $h \neq i$ holds since $\mathrm{supp}^+(x^* - y) \cap \mathrm{supp}^-(x^* - y) = \emptyset$. We have

$$f(x^*) \leq f(x^* - \chi_i + \chi_h) \tag{3.18}$$

since $x^* \in \arg\min f$. By the condition (3.15), we have

$$f(y) \leq f(y + \chi_i - \chi_h) \tag{3.19}$$

since $y + \chi_i - \chi_h = x - \chi_h + \chi_j$. It follows from (3.17), (3.18), and (3.19) that the inequalities in (3.18) and (3.19) hold with equality, i.e.,

$$f(x^*) = f(x^* - \chi_i + \chi_h), \qquad f(y) = f(y + \chi_i - \chi_h).$$

The equation $f(y) = f(y + \chi_i - \chi_h) = f(x - \chi_h + \chi_j)$ and the inequality $f(y) < f(x)$ imply that $h \neq j$, i.e., $h \notin \{i, j\}$. The equation $f(x^*) = f(x^* - \chi_i + \chi_h)$ implies that $x^* - \chi_i + \chi_h$ is also a minimizer of f. □

Proof of Theorem 3.11. Let x, i, j be as in the statement of Theorem 3.11, and put $y = x - \chi_i + \chi_j$. Then, we have $f(y) < f(x)$ by (3.14), which implies that $i \neq j$.

We first show that there exists some $y^* \in M^*(x)$ such that $y^*(i) \leq x(i) - 1$. Assume, to the contrary, that $y^*(i) > x(i) - 1 = y(i)$ holds for every $y^* \in M^*(x)$. Let $y^* \in M^*(x)$ be a vector minimizing the value $y^*(i)$ among all such vectors. By Lemma 3.13 (i), there exists some $h \in \mathrm{supp}^-(y^* - y) \setminus \{i, j\}$ such that $y^* - \chi_i + \chi_h \in \arg\min f$. Since $y^*(h) < y(h) = x(h)$, we have $\|(y^* - \chi_i + \chi_h) - x\|_1 \leq \|y^* - x\|_1$ by Lemma 3.12. Since $y^* \in M^*(x)$ and $y^* - \chi_i + \chi_h \in \arg\min f$, it follows that $y^* - \chi_i + \chi_h \in M^*(x)$. This, however, is a contradiction to the choice of y^* since $(y^* - \chi_i + \chi_h)(i) = y^*(i) - 1 < y^*(i)$.

We then will show that there exists some $x^* \in M^*(x)$ satisfying both of $x^*(i) \leq x(i) - 1$ and $x^*(j) \geq x(j) + 1$. Let $x^* \in M^*(x)$ be a vector with $x^*(i) \leq x(i) - 1$. If there exists such x^* with $x^*(j) \geq x(j) + 1$, then we are done. Hence, we assume, to the contrary, that $x^*(j) < x(j) + 1 = y(j)$ for every $x^* \in M^*(x)$ with $x^*(i) \leq x(i) - 1$, and suppose that x^* maximizes the value $x^*(j)$ among all such x^*. By Lemma 3.13, there exists some $k \in \mathrm{supp}^+(x^* - y) \setminus \{i, j\}$ such that $x^* + \chi_j - \chi_k \in \arg\min f$.

We show that $x^* + \chi_j - \chi_k \in M^*(x)$ holds. Since $x^*(k) > y(k) = x(k)$, we have $\|(x^* + \chi_j - \chi_k) - x\|_1 \leq \|x^* - x\|_1$ by Lemma 3.12 (ii). We also have $(x^* + \chi_j - \chi_k)(i) = x^*(i) \leq x(i) - 1$ since $i \notin \{j, k\}$. This, however, is a contradiction to the choice of x^* since $(x^* + \chi_j - \chi_k)(j) = x^*(j) + 1 > x^*(j)$. Hence, we have $x^*(j) \geq x(j) + 1$.

This concludes the proof of Theorem 3.11. □

3.4.2 Proof of Theorem 3.10 (i)

Instead of Theorem 3.10 (i), we prove the same statement under a weaker assumption.

Theorem 3.14. *Let $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be a jump M-convex function with $\arg\min f \neq \emptyset$. Also, let $x \in \mathrm{dom} f$ be a vector, and $s^*, t^* \in U$ be vectors satisfying the following conditions:*

$$f(x + s^* + t^*) < f(x), \tag{3.20}$$
$$f(x + s^* + t^*) \leq f(x + p + t^*) \qquad (\forall p \in U), \tag{3.21}$$
$$f(x + s^* + t^*) \leq f(x + s^* + q) \qquad (\forall q \in U). \tag{3.22}$$

Put $\{i\} = \mathrm{supp}(s^*)$ and $\{j\} = \mathrm{supp}(t^*)$, and define the set $S \subseteq \mathbb{Z}^n$ by (3.9). Then, $M^*(x) \cap S \neq \emptyset$ holds.

It is easy to see that the claim (i) in Theorem 3.10 follows immediately from Theorem 3.14 since the two vectors s^*, t^* in Theorem 3.10 satisfy the conditions (3.20)–(3.22).

In the proof we repeatedly use the following lemmas.

Lemma 3.15. *We have $\|(x^* - \chi_i - p) - x\|_1 \leq \|x^* - x\|_1$ for $x, x^* \in \mathbb{Z}^n$, $i \in N$, and $p \in \mathrm{inc}(x, x^*)$.*

Proof. Since $p \in \mathrm{inc}(x, x^*)$, we have $\|(x^* - p) - x\|_1 - \|x^* - x\|_1 = -1$, from which follows that

$$\begin{aligned}
&\|(x^* - \chi_i - p) - x\|_1 - \|x^* - x\|_1 \\
&= (\|(x^* - \chi_i - p) - x\|_1 - \|(x^* - p) - x\|_1) + (\|(x^* - p) - x\|_1 - \|x^* - x\|_1) \\
&\leq \|-\chi_i\|_1 - 1 = 0,
\end{aligned}$$

where we use the triangle inequality of L_1 norm. □

Lemma 3.16. *Let $x \in \mathrm{dom} f$ and $s^*, t^* \in U$ be vectors satisfying the conditions (3.20), (3.21), and (3.22). Suppose that $s^* = -\chi_i$ and $t^* = -\chi_j$ for some $i, j \in N$, and put $y = x - \chi_i - \chi_j$. Then, for every $x^* \in \arg\min f$ with $x^*(i) > y(i)$, there exists some $p \in \mathrm{inc}(x - \chi_j, x^*) \setminus \{-\chi_i, +\chi_j\}$ such that $x^* - \chi_i - p \in \arg\min f$.*

Proof. The condition (JM-EXC) applied to y, x^* and $+\chi_i \in \mathrm{inc}(y, x^*)$ implies that

$$f(y) + f(x^*) \geq f(y + \chi_i + p) + f(x^* - \chi_i - p) \tag{3.23}$$

for some $p \in \mathrm{inc}(y + \chi_i, x^*) = \mathrm{inc}(x - \chi_j, x^*)$; we have $p \neq -\chi_i$ since $x^*(i) > y(i)$ and $p \in \mathrm{inc}(y + \chi_i, x^*)$. It holds that

$$f(x^*) \leq f(x^* - \chi_i - p) \tag{3.24}$$

since $x^* \in \arg\min f$. By the condition (3.21), we have

$$f(y) \leq f(y + \chi_i + p) \tag{3.25}$$

since $y + \chi_i + p = x - \chi_j + p$. It follows from (3.23), (3.24), and (3.25) that the inequalities in (3.24) and in (3.25) hold with equality, i.e.,

$$f(x^*) = f(x^* - \chi_i - p), \qquad f(y) = f(y + \chi_i + p) = f(x + p - \chi_j).$$

The first equation implies that $x^* - \chi_i - p \in \arg\min f$ since $x^* \in \arg\min f$. The equation $f(y) = f(x + p - \chi_j)$ implies that $p \neq +\chi_j$ since $f(y) < f(x)$ holds by (3.20). □

Proof of Theorem 3.14. Let x, s^*, t^* be as in the statement of Theorem 3.14, and put $y = x + s^* + t^*$. Then, we have $f(y) < f(x)$ by (3.20), which implies that $s^* \neq -t^*$. Hence, it suffices to consider the following six cases:

Case 1 : $i = j,\ s^* = t^* = -\chi_i$,
Case 2 : $i = j,\ s^* = t^* = +\chi_i$,
Case 3 : $i \neq j,\ s^* = -\chi_i,\ t^* = -\chi_j$,
Case 4 : $i \neq j,\ s^* = -\chi_i,\ t^* = +\chi_j$,
Case 5 : $i \neq j,\ s^* = +\chi_i,\ t^* = -\chi_j$,
Case 6 : $i \neq j,\ s^* = +\chi_i,\ t^* = +\chi_j$.

In the following, we give proofs only for Case 1 and Case 3; proofs for the remaining cases can be done similarly since Case 2 is symmetric to Case 1 and Cases 4, 5, and 6 are symmetric to Case 3.

[**Proof of Case 1:** $s^* = t^* = -\chi_i$] We prove that there exists some $x^* \in M^*(x)$ such that $x^*(i) \leq x(i) - 2$. Assume, to the contrary, that $x^*(i) > x(i) - 2 = y(i)$ holds for every $x^* \in M^*(x)$. Let $x^* \in M^*(x)$ be a vector minimizing the value $x^*(i)$ among all such vectors. By Lemma 3.16, there exists some $p' \in \mathrm{inc}(x - \chi_i, x^*) \setminus \{-\chi_i, +\chi_i\}$ such that $x^* - \chi_i - p' \in \arg\min f$. Since $p' \in \mathrm{inc}(x - \chi_i, x^*) \setminus \{-\chi_i, +\chi_i\} \subseteq \mathrm{inc}(x, x^*)$, we have $\|(x^* - \chi_i - p') - x\|_1 \leq \|x^* - x\|_1$ by Lemma 3.15, while we have $\|(x^* - \chi_i - p') - x\|_1 \geq \|x^* - x\|_1$ since $x^* \in M^*(x)$ and $x^* - \chi_i - p' \in \arg\min f$. Hence, it follows that $x^* - \chi_i - p' \in M^*(x)$. This, however, is a contradiction to the choice of x^* since $(x^* - \chi_i - p')(i) = x^*(i) - 1 < x^*(i)$. Hence, we have $x^*(i) \leq x(i) - 2$.

[**Proof of Case 3:** $i \neq j,\ s^* = -\chi_i,\ t^* = -\chi_j$] We first show that there exists a vector $y^* \in M^*(x)$ with $y^*(i) \leq x(i) - 1$. Assume, to the contrary, that $y^*(i) > x(i) - 1 = y(i)$ holds for every $y^* \in M^*(x)$. Let $y^* \in M^*(x)$ be a vector minimizing the value $y^*(i)$ among all such vectors. By Lemma 3.16, there exists some $p' \in \mathrm{inc}(x - \chi_j, y^*) \setminus \{-\chi_i, +\chi_j\}$ such

that $y^* - \chi_i - p' \in \arg\min f$. Since $p' \in \mathrm{inc}(x - \chi_j, y^*) \setminus \{+\chi_j\} \subseteq \mathrm{inc}(x, y^*)$, it follows from Lemma 3.15 that $\|(y^* - \chi_i - p') - x\|_1 \leq \|y^* - x\|_1$, which, together with $y^* \in M^*(x)$, implies that $y^* - \chi_i - p' \in M^*(x)$. This, however, is a contradiction to the choice of y^* since

$$(y^* - \chi_i - p')(i) = y^*(i) - 1 - p'(i) \leq y^*(i) - 1 < y^*(i),$$

where the first inequality is by $p' \neq -\chi_i$.

We then show that there exists some $x^* \in M^*(x)$ satisfying both of $x^*(i) \leq x(i) - 1$ and $x^*(j) \leq x(j) - 1$. Let $x^* \in M^*(x)$ be a vector with $x^*(i) \leq x(i) - 1$. If there exists such x^* with $x^*(j) \leq x(j) - 1$, then we are done. Hence, we assume, to the contrary, that $x^*(j) > x(j) - 1 = y(j)$ for every such x^*, and suppose that x^* minimizes the value $x^*(j)$ among all such x^*. By Lemma 3.16, there exists some $q' \in \mathrm{inc}(x - \chi_i, x^*) \setminus \{-\chi_j, +\chi_i\}$ such that $x^* - \chi_j - q' \in \arg\min f$. Since $q' \in \mathrm{inc}(x - \chi_i, x^*) \setminus \{+\chi_i\} \subseteq \mathrm{inc}(x, x^*)$, it follows from Lemma 3.15 that $\|(x^* - \chi_j - q') - x\|_1 \leq \|x^* - x\|_1$, which, together with $x^* \in M^*(x)$, implies that $x^* - \chi_j - q' \in M^*(x)$.

Moreover, we have $(x^* - \chi_j - q')(i) \leq x(i) - 1$, which can be shown as follows. Since $x^*(i) \leq x(i) - 1$, we have $(x^* - \chi_j - q')(i) \leq x^*(i) \leq x(i) - 1$ if $q'(i) \neq -1$, i.e., $q' \neq -\chi_i$. If $q' = -\chi_i$, then it holds that $x^*(i) < x(i) - 1$ since $q' \in \mathrm{inc}(x - \chi_i, x^*)$. Hence, we have $(x^* - \chi_j - q')(i) = x^*(i) + 1 \leq x(i) - 1$.

We have shown that $x^* - \chi_j - q' \in M^*(x)$ and $(x^* - \chi_j - q')(i) \leq x(i) - 1$. This, however, is a contradiction to the choice of x^* since

$$(x^* - \chi_j - q')(j) = x^*(j) - 1 - q'(j) \leq x^*(j) - 1 < x^*(j),$$

where the first inequality is by $q' \neq -\chi_j$. Hence, we have $x^*(j) \leq x(j) - 1$. This concludes the proof of Theorem 3.14. □

3.5 Remarks on Minimizer Cut Properties

The following variants of the minimizer cut properties (Theorems 3.2 and 3.7) for the problems (MC) and (JMC) are known:

Theorem 3.17 ([43, Theorem 2.2]). *Let $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be an M-convex function with $\arg\min f \neq \emptyset$, and $x \in \mathrm{dom} f$.*
(i) *For $j \in N$, let $i \in N$ be an element satisfying*

$$f(x - \chi_i + \chi_j) \leq f(x - \chi_h + \chi_j) \qquad (\forall h \in N). \tag{3.26}$$

Then, there exists $x^* \in \arg\min f$ such that $x^*(i) \leq x(i) - 1$ if $i \neq j$ and $x^*(i) \leq x(i)$ if $i = j$.
(ii) *For $i \in N$, let $j \in N$ be an element satisfying*

$$f(x - \chi_i + \chi_j) \leq f(x - \chi_i + \chi_k) \qquad (\forall k \in N). \tag{3.27}$$

Then, there exists $x^* \in \arg\min f$ such that $x^*(j) \geq x(j) + 1$ if $j \neq i$ and $x^*(j) \geq x(j)$ if $j = i$.

Theorem 3.18 ([41, Theorem 4.1])**.** *Let $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be a jump M-convex function with $\arg\min f \neq \emptyset$, and $x \in \mathrm{dom} f$. For $t \in U$, let $s^* \in U$ be a vector satisfying*

$$f(x + s^* + t) \leq f(x + p + t) \qquad (\forall p \in U). \tag{3.28}$$

Put $\{i\} = \mathrm{supp}(s^*)$ and $\{j\} = \mathrm{supp}(t)$, and define

$$\widehat{S} = \begin{cases} \{y \in \mathbb{Z}^n \mid y(i) \leq x(i) - 2\} & (\text{if } i = j,\ s^* = t = -\chi_i), \\ \{y \in \mathbb{Z}^n \mid y(i) \geq x(i) + 2\} & (\text{if } i = j,\ s^* = t = +\chi_i), \\ \{y \in \mathbb{Z}^n \mid y(i) \leq x(i)\} & (\text{if } i = j,\ s^* = -\chi_i,\ t = +\chi_i), \\ \{y \in \mathbb{Z}^n \mid y(i) \geq x(i)\} & (\text{if } i = j,\ s^* = +\chi_i,\ t = -\chi_i), \\ \{y \in \mathbb{Z}^n \mid y(i) \leq x(i) - 1\} & (\text{if } i \neq j,\ s^* = -\chi_i), \\ \{y \in \mathbb{Z}^n \mid y(i) \geq x(i) + 1\} & (\text{if } i \neq j,\ s^* = +\chi_i). \end{cases} \tag{3.29}$$

Then, $\arg\min f \cap \widehat{S} \neq \emptyset$ holds.

As in Sections 3.2 and 3.3, we may also consider the following stronger versions of Theorems 3.17 and 3.18 by replacing the set $\arg\min f$ in the statements of the theorems with $M^*(x)$; recall that $M^*(x)$ is the set of the minimizers of f nearest to x (see (3.4)).

Theorem 3.19. *Let $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be an M-convex function with $\arg\min f \neq \emptyset$, and $x \in \mathrm{dom} f$.*
(i) *For $j \in N$, let $i \in N$ be an element satisfying* (3.26)*. Then, there exists $x^* \in M^*(x)$ such that $x^*(i) \leq x(i) - 1$ if $i \neq j$ and $x^*(i) \leq x(i)$ if $i = j$.*
(ii) *For $i \in N$, let $j \in N$ be an element satisfying* (3.27)*. Then, there exists $x^* \in M^*(x)$ such that $x^*(j) \geq x(j) + 1$ if $j \neq i$ and $x^*(j) \geq x(j)$ if $j = i$.*

Theorem 3.20. *Let $f : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be a jump M-convex function with $\arg\min f \neq \emptyset$, and $x \in \mathrm{dom} f$. For $t \in U$, let $s^* \in U$ be a vector satisfying* (3.28)*. Put $\{i\} = \mathrm{supp}(s^*)$ and $\{j\} = \mathrm{supp}(t)$, and define the set $\widehat{S} \subseteq \mathbb{Z}^n$ by* (3.29)*. Then, $M^*(x) \cap \widehat{S} \neq \emptyset$ holds.*

Proofs of Theorems 3.19 and 3.20 are similar to and simpler than those of Theorems 3.11 and 3.14 and therefore omitted.

Chapter 4

Algorithms for Separable Convex Resource Allocation Problem with L_1-distance Constraint

4.1 Introduction

In this chapter, we consider the simple resource allocation problem with an additional constraint called the L_1-distance constraint. The L_1-distance constraint is a constraint that the L_1-distance from a given vector to a solution vector is bounded by a given constant. Thus, the problem is formulated as follows:

$$\begin{array}{lll} (\mathrm{L_1SRA}) & \text{Minimize} & \displaystyle\sum_{i=1}^{n} f_i(x(i)) \\ & \text{subject to} & \displaystyle\sum_{i=1}^{n} x(i) = r, \\ & & \|x-y\|_1 \leq K, \\ & & x \in \mathbb{Z}_+^n, \end{array}$$

where K is a non-negative integer and y is an n-dimensional non-negative integral vector with $y(N) = r$.

The resource allocation problem with the L_1-distance constraint arises naturally when re-allocation of a given resource is required. Let us consider a situation where the original allocation of a resource is given by a vector y and we are required to re-allocate the resource. In such a situation it is often the case that we have a constraint that a new allocation x is close to the original allocation y, which can be represented by the L_1-distance constraint $\|x-y\|_1 \leq K$. Indeed, such a constraint is used by Freund et al. [10] in the formulation of the bike allocation problem in a bike sharing system.

The aim of this chapter is to reveal the combinatorial structure of the problem (L_1SRA) and to show its polynomial-time solvability. For this, we show that (L_1SRA) can be formulated as a submodular resource allocation problem, which is a special case of M-convex function minimization. This result immediately implies the polynomial-time solvability of (L_1SRA) since the submodular resource allocation problem can be solved in polynomial

time. Furthermore, we apply to (L_1SRA) the existing algorithms for the submodular resource allocation problem such as the greedy algorithm [8] and the scaling algorithm [15] and analyze the running time of the specialized implementations. In particular, we show that the greedy algorithm and the scaling algorithm for (L_1SRA) run in $O(\min(r, K) \log n)$ time and $O(n \log n \min(\log(r/n), \log(K/n)))$ time, respectively.

4.2 Review of Submodular Resource Allocation Problem

In this section, we explain the submodular resource allocation problem and present its algorithms. A set function $\rho : 2^N \to \mathbb{Z}$ is said to be *submodular* if it satisfies the submodular inequality:

$$\rho(S) + \rho(T) \geq \rho(S \cup T) + \rho(S \cap T) \quad (\forall S, T \in 2^N). \tag{4.1}$$

We also say that a set function ρ is *monotone nondecreasing* if the inequality $\rho(S) \leq \rho(T)$ holds for every $S, T \in 2^N$ with $S \subseteq T$. A *polymatroid constraint* is a constraint given as

$$x(S) \leq \rho(S) \quad (\forall S \in 2^N)$$

with a monotone nondecreasing submodular function $\rho : 2^N \to \mathbb{Z}$ with $\rho(\emptyset) = 0$. The simple resource allocation problem with a polymatroid constraint is called the *submodular resource allocation problem*. In the submodular resource allocation problem, we assume $\rho(N) = r$ without loss of generality. This problem is formulated as follows:

$$\begin{array}{lll} \text{Minimize} & \displaystyle\sum_{i=1}^{n} f_i(x(i)) & \\ \text{subject to} & x(N) = r, & \\ & x(S) \leq \rho(S) & (\forall S \subseteq N), \\ & x \in \mathbb{Z}_+^n. & \end{array}$$

It is known that this problem is a special case of M-convex function minimization [44].

In the following, we explain a greedy algorithm [8] and a scaling algorithm [15] as two fundamental algorithms for the submodular resource allocation problem. These algorithms are used to solve (L_1SRA) in Section 4.4.

We first explain a greedy algorithm. This algorithm is described as follows:

Algorithm GREEDY_SRA

Step 0: Set $x := \mathbf{0}$ and $N' = N$.

Step 1: If $x(N) = r$, output x and stop.

Step 2: Find $j \in N'$ with the smallest value of $f(x + \chi_j)$.

Step 3: If $x + \chi_j$ does not satisfy the polymatroid constraint, then set $N' = N' \setminus \{j\}$ and $j = 0$.

Step 4: Set $x = x + \chi_j$, and go to Step 1.

The greedy algorithm starts with an initial solution given by $x = (0, 0, \ldots, 0)^\top$. In each iteration, some variable $x(i)$ with the minimum value of the increment $f_i(x(i) + 1) - f_i(x(i))$

is increased by one. This step is repeated until the sum of all components of vector x is equal to r. The time complexity of this algorithm is $O(r(\log n + F))$, where F denotes the time required to check whether a given vector satisfies a polymatroid constraint.

We then describe a scaling algorithm for the submodular resource allocation problem [15]. In the scaling algorithm, we increase variables by multiple units in each iteration, while in the greedy algorithm variables are increased by single unit.

This scaling algorithm consists of the main routine named Scaling_SRA and the subroutine named Submodular_Greedy_SRA(s, l). This algorithm is described as follows:

Algorithm Submodular_Greedy_SRA(s, l)

Step 0: Set $x = l$ and $N' = N$.
Step 1: If $N' = \emptyset$ then output $x^{(s)} = x$.
Step 2: Find $j \in N'$ with the smallest value of $f(x + \chi_j)$, and set $d = s \cdot \chi_j$.
Step 3: If $x + d$ does not satisfy the polymatroid constraint, then set $d = \alpha \cdot \chi_j$ and $N' = N' \backslash \{j\}$ where $\alpha = \max\{\alpha' \mid x + \alpha' \cdot \chi_j$ satisfies the polymatroid constraint$\}$.
Step 4: Set $x = x + d$, and go to Step 1.

Algorithm Scaling_SRA

Step 0: Set $s = \lceil r/2n \rceil$, $l = \mathbf{0}$.
Step 1: If $s \leq 1$ then call Submodular_Greedy_SRA$(1, l)$, and return its output x^*.
Step 2: Call Submodular_Greedy_SRA(s, l), and let $x^{(s)}$ be its output.
Step 3: Let l be defined by $l(i) = \max(x^{(s)}(i) - s, 0)$ for each $i \in N$.
Step 4: $s = \lceil s/2 \rceil$, and go to Step 1.

In Submodular_Greedy_SRA(s, l), where $s \in \mathbb{Z}_+$ and $l \in \mathbb{Z}_+^n$, we start with the initial solution l and iteratively increment some variable by s units; if it is infeasible, then we find maximum feasible increment α and increase the variable by α units. Scaling_SRA repeatedly executes Submodular_Greedy_SRA(s, l), where s and l are initially set to $s = \lceil r/2n \rceil$ and $l = \mathbf{0}$, and in each iteration the step size s is gradually decreased and the vector l is updated. The time complexity of Scaling_SRA is $O(n(\log n + \tilde{F}) \log(r/n))$, where $\tilde{F}$ denotes the time required to compute α in Step 3. Note that $\tilde{F} = O(F \log n)$ since α can be computed by binary search.

We can use any lower bound of an optimal solution as an initial solution of these algorithms instead of the zero vector (see, e.g., [11]). As a candidate of such an initial vector, we can use any vector b that satisfies $x \geq b$ for every feasible solution x of the problem. For example, this condition is satisfied by the vector b given by

$$b_i = \rho(N) - \rho(N \setminus \{i\}) \qquad (i \in N).$$

The time complexity of the modified greedy algorithm and the modified scaling algorithm depend on $\tilde{r} = \rho(N) - b(N)$ instead of the parameter r and given as $O(\tilde{r}(\log n + F))$ and $O(n(\log n + \tilde{F}) \log(\tilde{r}/n))$, respectively.

4.3 Structure and Algorithms of L_1SRA

To show that (L_1SRA) can be formulated as a submodular resource allocation problem, we use a set function $\rho : 2^N \to \mathbb{Z}$ defined by

$$\rho(S) = \begin{cases} 0 & (\text{if } S = \emptyset), \\ r & (\text{if } S = N), \\ \min(r, y(S) + k) & (\text{otherwise}), \end{cases} \tag{4.2}$$

where $k = \lfloor (K/2) \rfloor$.

Lemma 4.1. (i) *The function ρ is monotone nondecreasing and submodular.*
(ii) *The feasible region $R \subseteq \mathbb{Z}^n$ of* (L_1SRA) *is equal to the set $R_\rho \subseteq \mathbb{Z}^n$ given by*

$$R_\rho = \{x \in \mathbb{Z}^n_+ \mid x(S) \leq \rho(S) \ (\forall S \in 2^N), \ x(N) = r\}. \tag{4.3}$$

Proof of the claim (i). The function ρ is monotone nondecreasing by definition. We prove that ρ satisfies the submodular inequality (4.1) for every $S, T \in 2^N$. Recall that y is a non-negative vector with $y(N) = r$. The submodular inequality holds obviously if one of S and T is in $\{\emptyset, N\}$. In the following, we discuss the case where S and T are nonempty proper subsets of N.

Suppose that $\rho(S) = r$. Then we obtain $\rho(S \cup T) = r$, and therefore it holds that

$$\rho(S) + \rho(T) = r + \min(r, y(T) + k) \geq r + \min(r, y(S \cap T) + k) \geq \rho(S \cup T) + \rho(S \cap T).$$

The proof for the case $\rho(T) = r$ is similar. We then assume $\rho(S) = y(S) + k$ and $\rho(T) = y(T) + k$. Then it holds that

$$\begin{aligned} \rho(S) + \rho(T) &= (y(S) + k) + (y(T) + k) \\ &= (y(S \cup T) + k) + (y(S \cap T) + k) \\ &\geq \min(r, y(S \cup T) + k) + \min(r, y(S \cap T) + k) \\ &\geq \rho(S \cup T) + \rho(S \cap T). \end{aligned}$$

Therefore, the function ρ satisfies the submodular inequality. □

Proof of the claim (ii). We first show that $x \in R_\rho$ holds for all $x \in R$. For $x \in R$, the vector x is non-negative and satisfies $x(E) = r$ since R is the feasible region of (L_1SRA). Therefore, it suffices to show that x satisfies the polymatroid constraint $x(S) \leq \rho(S)$ $(S \in 2^N)$. The polymatroid constraint holds obviously if $S = \emptyset$ and $S = N$. Therefore, in the following, we discuss the case with $\emptyset \subset S \subset N$ and show the inequality $x(S) \leq \rho(S) = \min(r, y(S) + k)$.

This inequality is equivalent to the two inequalities $x(S) \leq r$ and $x(S) \leq y(S) + k$. The former inequality follows from $x(N) = r$ and $x \geq 0$. We use the L_1-distance constraint $\|x - y\|_1 \leq K$ in order to show the latter inequality $x(S) \leq y(S) + k$. The L_1-distance $\|x - y\|_1$ is an even number since $x(N) = y(N)$. Therefore, it holds that $\|x - y\|_1 \leq 2\lfloor (K/2) \rfloor = 2k$. Since the sum of all components of the vector $x - y$ is zero, it holds that

$$\sum_{i \in N} \max(0, x(i) - y(i)) = \sum_{i \in N} \max(0, -x(i) + y(i)).$$

Consequently, we obtain

$$\begin{aligned} 2k &\geq \|x - y\|_1 \\ &= \sum_{i \in N} \max(0, x(i) - y(i)) + \sum_{i \in N} \max(0, -x(i) + y(i)) \\ &= 2 \sum_{i \in N} \max(0, x(i) - y(i)). \end{aligned}$$

By the inequality above, it holds that

$$x(S) - y(S) \leq \sum_{i \in S} \max(0, x(i) - y(i)) \leq \sum_{i \in N} \max(0, x(i) - y(i)) \leq k.$$

This inequality implies $x \in R_\rho$.

We then show that $x \in R$ holds for all $x \in R_\rho$. For $x \in R_\rho$, it is easy to see that x satisfies the constraints of (L_1SRA), except for the L_1-distance constraint. In the following, we show that the vector x satisfies the L_1-distance constraint $\|x - y\|_1 \leq K$.

We define subsets $S_+, S_- \subseteq N$ by

$$S_+ = \{i \in N \mid x(i) \geq y(i)\}, \quad S_- = \{i \in N \mid x(i) < y(i)\}.$$

It is noted that $S_+ \cap S_- = \emptyset$ and $S_+ \cup S_- = N$. Also, the set S_+ is nonempty since $x(N) = y(N)$. If $S_- = \emptyset$, then we have $x = y$ since $x(N) = y(N) = r$, implying that $\|x - y\|_1 = 0 \leq K$. In the following, we assume $S_- \neq \emptyset$ and prove the inequality $\|x - y\|_1 \leq K$.

It holds that

$$\begin{aligned} \|x - y\|_1 &= \sum_{i \in S_+} (x(i) - y(i)) - \sum_{j \in S_-} (x(j) - y(j)) \\ &= (x(S_+) - y(S_+)) - (x(S_-) - y(S_-)) \\ &= x(S_+) - x(S_-) - y(S_+) + y(S_-). \end{aligned}$$

Since $x(S_+) + x(S_-) = y(S_+) + y(S_-) = r$, it follows that

$$\|x - y\|_1 = 2x(S_+) - 2y(S_+). \tag{4.4}$$

Since the vector x satisfies the polymatroid constraint $x(S) \leq \rho(S)$ for $S \subseteq N$, we obtain

$$x(S_+) \leq \rho(S_+) = \min(r, y(S_+) + k) \leq y(S_+) + k,$$

which, combined with (4.4), implies

$$\|x - y\|_1 = 2(x(S_+) - y(S_+)) \leq 2k \leq K.$$

This concludes the proof of $x \in R_\rho$. □

From Lemma 4.1 the next theorem follows immediately.

Theorem 4.2. (L_1SRA) *can be reformulated as a submodular resource allocation problem with a polymatroid constraint associated with the submodular function* $\rho : 2^N \to \mathbb{Z}$ *in* (4.2).

Remark 4.3. Theorem 4.2 shows that if we add the L_1-distance constraint to the simple resource allocation problem, then the resulting problem can be reformulated as a submodular resource allocation problem. On the other hand, the example below shows that if the L_1-distance constraint is added to a slightly more submodular resource allocation problem, then the resulting problem cannot be reformulated as a submodular resource allocation problem.

As a more submodular resource allocation problem, we consider the resource allocation problem with the *generalized upper bound constraint*; the generalized upper bound constraint is a constraint of the form $x(S_j) \leq b_j$ $(j = 1, 2, \ldots, m)$, where $\{S_1, S_2, \ldots, S_m\}$ is a partition of the set N and $b_1, b_2, \ldots, b_m$ are non-negative integers. If the L_1-distance constraint is added to the problem, then the feasible region is given as the set of non-negative integral vectors $x \in \mathbb{Z}_+^n$ satisfying

$$\begin{aligned} &x(N) = r, \\ &\|x - y\|_1 \leq K, \\ &x(S_j) \leq b_j \quad (j = 1, 2, \ldots, m). \end{aligned}$$

As a concrete example, we consider the following special case with $N = \{1, 2, 3, 4\}$:

$$\begin{aligned} &x(1) + x(2) + x(3) + x(4) = 4, \\ &|x(1) - 1| + |x(2) - 1| + |x(3) - 1| + |x(4) - 1| \leq 2, \\ &x(1) + x(2) \leq 2,\ x(3) + x(4) \leq 4. \end{aligned}$$

Assume, to the contrary, that this feasible region can be represented by a polymatroid constraint. Then, the set function $\rho : 2^N \to \mathbb{Z}$ defined by

$$\rho(S) = \max\{x(S) \mid x \in \mathbb{Z}_+^4 \text{ is a feasible solution}\} \quad (S \in 2^N)$$

must be a submodular function (see, e.g., [11]). By the constraint $x(1) + x(2) \leq 2$, it holds that

$$\rho(\{1, 2\}) \leq 2.$$

Since the vectors $(0, 2, 1, 1)$ and $(1, 1, 2, 0)$ are feasible solutions, we obtain

$$\rho(\{2\}) = 2,\ \rho(\{2, 3\}) = 3,\ \rho(\{1, 2, 3\}) = 4.$$

Therefore, for $S = \{1, 2\}, T = \{2, 3\}$, it holds that

$$\rho(S) + \rho(T) \leq 5 < 6 = \rho(S \cup T) + \rho(S \cap T).$$

Consequently, the set function ρ does not satisfy the submodular inequality (4.1), a contradiction. This implies that the feasible region given above cannot be represented by a polymatroid constraint. □

Remark 4.4. It can be shown that the set of vectors satisfying the L_1-distance constraint $\|x - y\|_1 \leq K$ has a nice structure called a bisubmodular polyhedron. This fact, however, does not imply the statement of Theorem 4.2. That is, the intersection of a bisubmodular polyhedron and a hyperplane of the form $\sum_{i \in N} x(i) = r$ cannot be represented by a polymatroid constraint, as shown below.

We denote $3^N = \{(X, Y) \mid X, Y \subseteq N,\ X \cap Y = \emptyset\}$. A function $\mu : 3^N \to \mathbb{R}$ is called a *bisubmodular function* if it satisfies the following inequality:

$$\mu(X_1, Y_1) + \mu(X_2, Y_2) \geq \mu((X_1 \cup X_2) \setminus (Y_1 \cup Y_2), (Y_1 \cup Y_2) \setminus (X_1 \cup X_2)) + \mu(X_1 \cap X_2, Y_1 \cap Y_2) \qquad (\forall (X_1, Y_1), (X_2, Y_2) \in 3^N).$$

A *bisubmodular polyhedron* is a polyhedron given as follows by using a bisumobular function μ:

$$P_*(\mu) = \{x \in \mathbb{R}^n \mid x(X) - x(Y) \leq \mu(X, Y)\ (\forall (X, Y) \in 3^N)\}.$$

It is known that $P_*(\mu)$ is an integral polyhedron for an integer-valued bisubmodular function μ [11]. Optimization of a linear function over a bisubmodular polyhedron can be solved by a certain greedy algorithm, and minimization of a separable convex function over an integral bisubmodular polyhedron can be solved in polynomial time (see, e.g., [11]).

Note that the set of vectors satisfying the L_1-distance constraint $\|x - y\|_1 \leq K$ is represented by a bisubmodular polyhedron $P_*(\mu)$ associated with the bisubmodular function $\mu : 3^N \to \mathbb{R}$ given by $\mu(X, Y) = K + y(X) - y(Y)$ $(\forall (X, Y) \in 3^N)$; bisubmodularity of this μ is easy to see from the definition.

In the following, we show by a concrete example that the intersection of an integral bisubmodular polyhedron and a hyperplane of the form $\sum_{i \in N} x(i) = r$ cannot be represented by a polymatroid constraint in general.

We consider the convex hull $\overline{S} \subseteq \mathbb{R}^4$ of the set

$$S = \{(0, 0, 0, 0), (1, 1, 0, 0), (0, 0, 1, 1), (1, 1, 1, 1)\}.$$

Each vector in the set S corresponds to a *matchable* set of an undirected graph with vertex set $\{1, 2, 3, 4\}$ and edge set $\{(1, 2), (3, 4)\}$; a vertex set of a graph is said to be matchable if there exists a matching that exactly covers the vertex set. It is known that given an undirected graph, the convex hull of the characteristic vectors of all matchable sets is an integral bisubmodular polyhedron [7]. Intersection of the convex hull $\overline{S}$ and the hyperplane $x(1)+x(2)+x(3)+x(4) = 2$ is given as the convex hull of the sets $\{(1, 1, 0, 0), (0, 0, 1, 1)\}$, and it is not difficult to see that this set cannot be represented by a polymatroid constraint. □

4.4 Algorithms for L_1SRA

We present algorithms for solving (L_1SRA) and analyze the running time. Theorem 4.2 implies that we can apply to (L_1SRA) the greedy algorithm and the scaling algorithm in Section 4.2. We analyze the running time of the algorithms.

Theorem 4.5. (L_1SRA) *can be solved by the greedy algorithm and the scaling algorithm in* $O(r \log n)$ *time and in* $O(n \log n \log(r/n))$ *time, respectively.*

Proof. To obtain the bound $O(r \log n)$ for the greedy algorithm, it suffices to show that $F = O(1)$, i.e., we can check in constant time whether a given vector x in each iteration of the algorithm satisfies the polymatroid constraint associated with the submodular function ρ in (4.2).

By keeping the value $x(N)$ during the run of the algorithm, we can check the constraint $x(N) \leq \rho(N)$ in constant time. In the case where the set S is nonempty proper subsets of N, we can check the polymatroid constraint $x(S) \leq \rho(S) = \min(r, y(S) + k)$ $(\emptyset \subset \forall S \subset N)$, by checking the two inequalities $x(S) \leq r$ and $x(S) \leq y(S) + k$ for each S. Since the vector x is non-negative, $x(S) \leq r$ holds automatically if the constraint $x(N) \leq \rho(N) = r$ holds. On the other hand, we can rewrite the inequality $x(S) \leq y(S) + k$ as

$$\sum_{i \in S} (x(i) - y(i)) \leq k.$$

Since the maximum value of the left-hand side is $\sum_{i:x(i)>y(i)}(x(i) - y(i))$, it suffices to check the inequality

$$\sum_{i:x(i)>y(i)} (x(i) - y(i)) \leq k.$$

We can check the inequality by keeping the value $\sum_{i:x(i)>y(i)}(x(i) - y(i))$ in each iteration. It is noted that the set $S^* = \{i \in N \mid x(i) > y(i)\}$ is not equal to the set N in each iteration since $x(N) \leq r = y(N)$. It is easy to see that we can update the value $\sum_{i:x(i)>y(i)}(x(i) - y(i))$ in constant time whenever the vector x is updated. Therefore, we can decide in $F = \mathrm{O}(1)$ time whether a vector x satisfies the polymatroid constraint.

To obtain the bound $\mathrm{O}(n \log n \log(r/n))$ for the scaling algorithm, we also need to show that $\tilde{F} = \mathrm{O}(1)$, i.e., we can compute in constant time the maximum feasible increment α of the variable $x(j)$. Since the maximum feasible increment α is computed by

$$\alpha = \min(r - x(N), k - \sum_{i:x(i)>y(i)} (x(i) - y(i)) - \min(0, x(j) - y(j))),$$

the discussion above shows that the value α can be computed in constant time, i.e., $\tilde{F} = \mathrm{O}(1)$. □

In the following, we show that the running time of the greedy algorithm and the scaling algorithm can be made faster in the case where K is smaller than r. Our idea is to replace the initial solution of the algorithms, which is originally set to the zero vector (see Section 4.2).

Consider a typical situation where K is much smaller than r. Since an optimal solution x^* of (L_1SRA) satisfies $\|x^* - y\|_1 \leq K$ by the L_1-distance constraint, the vector x^* is much closer to y than to the zero vector. We see from this observation that it makes sense that the initial solution is set to some vector close to y instead of the zero vector. The initial solution of the algorithms, on the other hand, must be a lower bound of some optimal solution of (L_1SRA) (see the discussion in Section 4.2). Since the vector y is not a lower bound of any optimal solution for (L_1SRA) in general, we cannot use y as an initial solution. Hence, as a good initial solution we need a vector that is close to y and is a lower bound of some optimal solution for (L_1SRA).

The next lemma shows that such a vector can be obtained by solving a certain optimization problem. Note that any vector $x \in \mathbb{Z}_+^n$ with $\sum_{i=1}^n x(i) = r - k$ and $x \leq y$ satisfies the L_1-distance constraint $\|x - y\|_1 \leq K$.

Lemma 4.6. *Let x' be an optimal solution of the following optimization problem:*

$$\begin{aligned} (\mathrm{SRA}^-) \quad \text{Minimize} \quad & \sum_{i=1}^{n} f_i(x(i)) \\ \text{subject to} \quad & \sum_{i=1}^{n} x(i) = r - k, \\ & x \le y, \\ & x \in \mathbb{Z}_+^n. \end{aligned}$$

Then, there exists an optimal solution x^* of (L_1SRA) *with $x^* \ge x'$.*

Proof. Let x^* be an optimal solution of (L_1SRA), and assume that the value $\sum_{i=1}^{n} \max(0, x'(i) - x^*(i))$ is the minimum among all optimal solutions. Since $\sum_{i=1}^{n} \max(0, x'(i) - x^*(i)) \le 0$ implies $x^* \ge x'$, we assume $\sum_{i=1}^{n} \max(0, x'(i) - x^*(i)) > 0$ and derive a contradiction.

By the assumption, there exists $h \in N$ with $x'(h) > x^*(h)$. We define

$$S'_+ = \{i \in N \mid x'(i) \ge x^*(i)\}, \quad S'_- = \{i \in N \mid x'(i) < x^*(i)\}.$$

Then, $h \in S'_+$ holds. Since

$$\sum_{i=1}^{n} x'(i) = r - k < r = \sum_{i=1}^{n} x^*(i),$$

we have $S'_- \neq \emptyset$. In the following, we show the existence of $j \in S'_-$ with $x'(j) < y(j)$.

We have

$$\sum_{i \in S'_+} (y(i) - x'(i)) < \sum_{i \in S'_+} (y(i) - x^*(i)) \le \sum_{i \in N} \max(0, y(i) - x^*(i)),$$

where the first strict inequality is by the definition of S'_+ and $h \in S'_+$. Since the vector x^* satisfies the L_1-distance constraint $\|x^* - y\|_1 \le K$, we can obtain the inequality $\sum_{i=1}^{n} \max(0, y(i) - x^*(i)) \le k$, as in the proof of Theorem 4.2, from which follows that

$$\sum_{i \in S'_+} (y(i) - x'(i)) < k. \tag{4.5}$$

On the other hand, it holds that

$$\begin{aligned} \sum_{i \in S'_+} (y(i) - x'(i)) + \sum_{i \in S'_-} (y(i) - x'(i)) &= \sum_{i=1}^{n} (y(i) - x'(i)) \\ &= \sum_{i=1}^{n} y(i) - \sum_{i=1}^{n} x'(i) \\ &= r - (r - k) = k, \end{aligned}$$

which, together with (4.5), implies that $\sum_{i \in S'_-} (y(i) - x'(i)) > 0$. This inequality shows that $x'(j) < y(j)$ holds for some $j \in S'_-$.

By the convexity of functions f_h and f_j and the inequalities $x'(h) > x^*(h)$ and $x'(j) < x^*(j)$, it holds that

$$f_h(x^*(h)+1) - f_h(x^*(h)) \leq f_h(x'(h)) - f_h(x'(h)-1), \tag{4.6}$$
$$f_j(x'(j)+1) - f_j(x'(j)) \leq f_j(x^*(j)) - f_j(x^*(j)-1). \tag{4.7}$$

Denoting $f(x) = \sum_{i=1}^{n} f_i(x(i))$ $(x \in \mathbb{Z}^n)$, we have the following inequality by (4.6) and (4.7):

$$f(x^*) + f(x') \geq f(x^* + \chi_h - \chi_j) + f(x' - \chi_h + \chi_j). \tag{4.8}$$

The vector $x' - \chi_h + \chi_j$ is a feasible solution of (SRA^-) since $x'(j) < y(j)$, and therefore it holds that $f(x') \leq f(x' - \chi_h + \chi_j)$. This inequality and (4.8) imply $f(x^*) \geq f(x^* + \chi_h - \chi_j)$. On the other hand, the vector $x^* + \chi_h - \chi_j$ satisfies the L_1-distance constraint since the inequality $y(h) \geq x'(h) > x^*(h)$ holds, and therefore $x^* + \chi_h - \chi_j$ is a feasible solution of (L_1SRA), from which follows the inequality $f(x^*) \leq f(x^* + \chi_h - \chi_j)$. Hence, it holds that $f(x^* + \chi_h - \chi_j) = f(x^*)$, implying that the vector $x^* + \chi_h - \chi_j$ is also an optimal solution of (L_1SRA), a contradiction to the choice of x^* since $x'(h) > x^*(h)$ and $x'(j) < x^*(j)$. Therefore, there exists an optimal solution x^* of (L_1SRA) with $x^* \geq x'$. □

By Lemma 4.6, (L_1SRA) can be solved by the greedy algorithm and the scaling algorithm by using an optimal solution of (SRA^-) as an initial solution. We estimate the time complexity of the resulting algorithms.

An optimal solution of (SRA^-) can be found in $\mathrm{O}(\min(K \log n, n \log(K/n)))$ time since (SRA^-) is essentially equivalent to the simple resource allocation problem. If we use an optimal solution of (SRA^-) as an initial solution b of the greedy algorithm and the scaling algorithm, the value of the parameter $\tilde{r}$ (see Section 4.2) satisfies

$$\tilde{r} = \rho(N) - b(N) = r - (r - k) = k = \mathrm{O}(K).$$

Therefore, the time complexity of the greedy algorithm is

$$\mathrm{O}(K \log n) + \mathrm{O}(K \log n) = \mathrm{O}(K \log n),$$

and the time complexity of the scaling algorithm is

$$\mathrm{O}(n \log(K/n)) + \mathrm{O}(n \log n \log(K/n)) = \mathrm{O}(n \log n \log(K/n)).$$

From the discussion above and Theorem 4.5, we obtain the following result:

Theorem 4.7. (L_1SRA) *can be solved by the greedy algorithm and the scaling algorithm in* $\mathrm{O}(\min(r, K) \log n)$ *time and* $\mathrm{O}(n \log n \min(\log(r/n), \log(K/n)))$ *time, respectively, by using the zero vector or an optimal solution of* (SRA^-) *as an initial solution of the algorithm.*

4.5 L_1SRA with Lower and Upper Bounds

As a more general problem, we consider the problem (L_1SRA) with lower and upper bounds for variables $x(j)$; such a problem is formulated as follows:

$$\begin{array}{lll} (\mathrm{L_1BRA}) & \text{Minimize} & \displaystyle\sum_{i=1}^{n} f_i(x(i)) \\ & \text{subject to} & x(N) = r, \\ & & \|x - y\|_1 \le K, \\ & & l(j) \le x(j) \le u(j) \quad (\forall j \in N), \\ & & x \in \mathbb{Z}_+^n. \end{array}$$

We may assume, without loss of generality, that each $l(j)$ is a non-negative integer. We will show that the problem with lower and upper bounds for variables can be solved in the same running time as the original problem (L_1SRA).

It is known that the combination of the submodular constraint and the lower and upper bounds for variables can be represented by another submodular constraint by using an appropriate submodular function (see, e.g., [18, Section 3.2]). Since the L_1-distance constraint is a special case of the submodular constraint, we can apply to (L_1BRA) the greedy algorithm and the scaling algorithm for the submodular resource allocation problem in Section 4.2.

Theorem 4.8. (L_1BRA) *can be solved by the greedy algorithm and the scaling algorithm in* $O(r \log n)$ *time and in* $O(n \log n \log(r/n))$ *time, respectively.*

Proof. To obtain the bound $O(r \log n)$ for the greedy algorithm, it suffices to show that $F = O(1)$. As in the proof of Theorem 4.5, we can check in constant time whether a given vector x satisfies the polymatroid constraint associated with the submodular function $\rho : 2^N \to \mathbb{Z}$ in (4.2). We can also check the lower and upper bounds constraint $l(j) \le x(j) \le u(j)$ in constant time by checking the inequality $x(j) \le u(j)$ for the incremented element $j \in N$ in each iteration. Therefore, we can decide in constant time whether a vector x satisfies the constraints of (L_1BRA) in each iteration. That is, $F = O(1)$ holds.

To obtain the bound $O(n \log n \log(r/n))$ for the scaling algorithm, we need to show that the maximum feasible increment α of a variable $x(j)$ can be computed in constant time. Since the maximum feasible increment α is given by

$$\alpha = \min[r - x(N), k - \sum_{i:x(i)>y(i)} (x(i) - y(i)) - \min(0, x(j) - y(j)), u(j) - x(j)],$$

as in the proof of Theorem 4.5, the value α can be computed in constant time. □

In the following, we show that the running time of the algorithms in Theorem 4.8 can be improved in the case where K is smaller than r. The idea is to use a non-zero initial vector in the algorithms, as we have done in Section 4.4 for (L_1SRA). We use the lower bound vector l as an initial vector if $l(N) \ge r - k$. In this case, the value of the parameter $\tilde{r}$ (see Section 4.2 for the definition of $\tilde{r}$) satisfies

$$\tilde{r} = \rho(N) - l(N) \le r - (r - k) = k = O(K).$$

Therefore, the greedy algorithm and the scaling algorithm for (L_1BRA) run in $O(\min(r, K) \log n)$ time and $O(n \log n \min(\log(r/n), \log(K/n)))$ time, respectively.

We next consider the case with $l(N) < r - k$. In this case, we obtain an initial vector of the algorithms by solving the following optimization problem:

$$\begin{aligned} (\mathrm{BRA}^-) \quad \text{Minimize} \quad & \sum_{i=1}^{n} f_i(x(i)) \\ \text{subject to} \quad & x(N) = r - k, \\ & l(j) \le x(j) \le \max(l(j), \min(y(j), u(j))) \quad (\forall j \in N), \\ & x \in \mathbb{Z}_+^n. \end{aligned}$$

As will be proved later, any optimal solution of (BRA^-) is a lower bound of some optimal solution of (L_1BRA). An optimal solution of (BRA^-) can be found in $O(\min(K \log n, n \log(K/n)))$ time since (BRA^-) is essentially equivalent to the simple resource allocation problem with the lower and upper bounds constraint. If we use an optimal solution of (BRA^-) as an initial solution b, the value of the parameter $\tilde{r}$ satisfies

$$\tilde{r} = \rho(N) - b(N) = r - (r - k) = k = O(K).$$

Therefore, the time complexity of the greedy algorithm is

$$O(K \log n) + O(K \log n) = O(K \log n),$$

and the time complexity of the scaling algorithm is

$$O(n \log(K/n)) + O(n \log n \log(K/n)) = O(n \log n \log(K/n)).$$

From the discussion above and Theorem 4.8, we obtain the following result:

Theorem 4.9. (L_1BRA) *can be solved by the greedy algorithm and the scaling algorithm in* $O(\min(r, K) \log n)$ *time and* $O(n \log n \min(\log(r/n), \log(K/n)))$ *time, respectively, by using* $l \in \mathbb{Z}^n$ *or an optimal solution of* (BRA^-) *as an initial solution of the algorithm.*

To conclude the proof of Theorem 4.9, it suffices to show that an optimal solution of (BRA^-) can be used as an initial solution of the algorithms.

Lemma 4.10. *Let* x' *be an optimal solution of* (BRA^-). *Then, there exists an optimal solution* x^* *of* (L_1BRA) *with* $x^* \ge x'$.

Proof. The proof is quite similar to that for Lemma 4.6. Let x^* be an optimal solution of (L_1BRA), and assume that the value $\sum_{i=1}^{n} \max(0, x'(i) - x^*(i))$ is the minimum among all optimal solutions. Since $\sum_{i=1}^{n} \max(0, x'(i) - x^*(i)) \le 0$ implies $x^* \ge x'$, we assume $\sum_{i=1}^{n} \max(0, x'(i) - x^*(i)) > 0$ and derive a contradiction.

By the assumption, there exists $h \in N$ with $x'(h) > x^*(h)$. We define

$$S'_+ = \{i \in N \mid x'(i) \ge x^*(i)\}, \quad S'_- = \{i \in N \mid x'(i) < x^*(i)\}.$$

Then, $h \in S'_+$ holds. Since $x'(N) = r - k < r = x^*(N)$, we have $S'_- \neq \emptyset$. In a similar way as in the proof of Lemma 4.6, we can show that $x'(j) < y(j)$ holds for some $j \in S'_-$.

In the following, we show that the vector $x' - \chi_h + \chi_j$ is a feasible solution of (BRA$^-$). Since $x'(h) > x^*(h)$, it holds that

$$l(h) \leq x^*(h) \leq (x' - \chi_h + \chi_j)(h) < x'(h) \leq \max(l(h), \min(y(h), u(h))).$$

Since $x'(j) < x^*(j)$, it also holds that

$$l(j) \leq x'(j) < (x' - \chi_h + \chi_j)(j) \leq x^*(j) \leq u(j),$$

which, together with $x'(j) < y(j)$ implies that

$$l(j) \leq (x' - \chi_h + \chi_j)(j) \leq \min(y(j), u(j)) \leq \max(l(j), \min(y(j), u(j))).$$

Hence, the vector $x' - \chi_h + \chi_j$ is a feasible solution of (BRA$^-$).

By the convexity of functions f_h and f_j and the inequalities $x'(h) > x^*(h)$ and $x'(j) < x^*(j)$, it holds that

$$f_h(x^*(h) + 1) - f_h(x^*(h)) \leq f_h(x'(h)) - f_h(x'(h) - 1), \tag{4.9}$$
$$f_j(x'(j) + 1) - f_j(x'(j)) \leq f_j(x^*(j)) - f_j(x^*(j) - 1). \tag{4.10}$$

Denoting $f(x) = \sum_{i=1}^n f_i(x(i))$ $(x \in \mathbb{Z}^n)$, we have the following inequality by (4.9) and (4.10):

$$f(x^*) + f(x') \geq f(x^* + \chi_h - \chi_j) + f(x' - \chi_h + \chi_j). \tag{4.11}$$

Since the vector $x' - \chi_h + \chi_j$ is a feasible solution of (BRA$^-$), it holds that $f(x') \leq f(x' - \chi_h + \chi_j)$. This inequality and (4.11) imply $f(x^*) \geq f(x^* + \chi_h - \chi_j)$. On the other hand, the vector $x^* + \chi_h - \chi_j$ satisfies the L_1-distance constraint and the lower and upper bounds constraint since $x^*(h) < x'(h) \leq \max(l(h), \min(y(h), u(h)))$ and $l(j) \leq x'(j) < x^*(j)$ hold. Therefore $x^* + \chi_h - \chi_j$ is a feasible solution of (L_1BRA), from which follows the inequality $f(x^*) \leq f(x^* + \chi_h - \chi_j)$. Hence, it holds that $f(x^* + \chi_h - \chi_j) = f(x^*)$, implying that the vector $x^* + \chi_h - \chi_j$ is also an optimal solution of (L_1BRA), a contradiction to the choice of x^* since $x'(h) > x^*(h)$ and $x'(j) < x^*(j)$. Therefore, there exists an optimal solution x^* of (L_1BRA) with $x^* \geq x'$. □

Chapter 5

Conclusion

In this thesis, we studied M-convex function minimization and related problems. We conclude this thesis with a brief summary of our contribution.

In Chapter 3, we analyzed the number of iterations required by the basic steepest descent algorithm for M-convex function minimization. Our proof for the exact time bound is based on a stronger variant of the minimizer cut property. In particular, we showed that the trajectory of the solutions generated by the basic steepest descent algorithm is a geodesic between the initial solution and the nearest minimizer. Furthermore, we extended our results for M-convex function minimization to jump M-convex function minimization, i.e., we provided the exact bound on the number of iterations required by the basic steepest descent algorithm, and showed that the trajectory of the solutions generated by the algorithm is a geodesic.

In Chapter 4, we considered the separable convex resource allocation problem with L_1-distance constraint (L_1SRA). We showed that (L_1SRA) can be formulated as a submodular resource allocation problem and can be solved efficiently by existing algorithms for the submodular resource allocation problem. We also showed by a concrete example that (L_1SRA) with a generalized upper bound cannot be represented by a submodular resource allocation problem. Furthermore, we provided fast implementation of existing algorithms, which called a greedy algorithm and a scaling algorithm, and analyzed their running time. We showed that the greedy algorithm and the scaling algorithm for (L_1SRA) run in $O(\min(r, K) \log n)$ time and $O(n \log n \min(\log(r/n), \log(K/n)))$ time, respectively.

Bibliography

[1] K. Ando, S. Fujishige and T. Naitoh: A greedy algorithm for minimizing a separable convex function over an integral bisubmodular polyhedron. *Journal of the Operations Research Society of Japan*, **37(3)** (1994), 362–375.

[2] K. Ando, S. Fujishige and T. Naitoh: A greedy algorithm for minimizing a separable convex function over an finite jump system. *Journal of the Operations Research Society of Japan*, **38(3)** (1995), 362–375.

[3] N. Apollonio and A. Sebő: Minsquare factors and maxfix covers of graphs. In *Proceedings of the 10th Conference on Integer Programming and Combinatorial Optimization (IPCO'04)*, (2004), 388–400.

[4] N. Apollonio and A. Sebő: Minconvex factors of prescribed size in graphs. *SIAM Journal on Discrete Mathematics*, **23** (2009), 1297–1310.

[5] K. Bérczi and Y. Kobayashi: An algorithm for $(n-3)$-connectivity augmentation problem: jump system approach. *Journal of Combinatorial Theory*, **B102** (2012), 565–587.

[6] A. Bouchet: Greedy algorithm and symmetric matroids. *Mathematical Programming*, **38** (1987), 147–159.

[7] A. Bouchet and W.H. Cunningham: Delta-matroids, jump systems, and bisubmodular polyhedra. *SIAM Journal on Discrete Mathematics*, **8** (1995), 17–32.

[8] A. Federgruen and H. Groenevelt: The greedy procedure for resource allocation problems: Necessary and sufficient conditions for optimality. *Operations Research*, **34** (1986), 909–918.

[9] G.N. Frederickson and D.B. Johnson: The complexity of selection and ranking in $X+Y$ and matrices with sorted columns. *Journal of Computer and System Sciences*, **24** (1982), 197–208.

[10] D. Freund, S.G. Henderson, and D.B. Shmoys: Minimizing multimodular functions and allocating capacity in bike-sharing systems. In *Proceedings of the 19th Conference on Integer Programming and Combinatorial Optimization (IPCO'17)*, (2017), 186–198.

[11] S. Fujishige: *Submodular Functions and Optimization, 2nd Edition* (Elsevier, Amsterdam, 2005).

[12] Z. Galil and N. Megiddo: A fast selection algorithm and the problem of optimum distribution of effort. *Journal of ACM*, **26** (1979), 58–64.

[13] O. Gross: A class of discrete type minimization problems. Research Memorandum RM-1644, Rand Corporation (1956).

[14] D.S. Hochbaum, R. Shamir, and J.G. Shanthikumar: A polynomial algorithm for an integer quadratic non-separable transportation problem. *Mathematical Programming*, **55(3)** (1992), 359–371.

[15] D.S. Hochbaum: Lower and upper bounds for the allocation problem and other nonlinear optimization problems. *Mathematics of Operations Research*, **19** (1994), 390–409.

[16] T. Ibaraki and K. Katoh: *Resource Allocation Problems: Algorithmic Approaches* (MIT Press, Cambridge, 1988).

[17] N. Katoh, T. Ibaraki, and H. Mine: A polynomial time algorithm for the resource allocation problem with a convex objective function. *Journal of Operational Research Society*, **30** (1979), 449–455.

[18] N. Katoh, A. Shioura, and T. Ibaraki: Resource allocation problems. In P.M. Pardalos, D.Z. Du, and R.L. Graham (eds.): *Handbook of Combinatorial Optimization* (Springer, New York, NY, 2013), 2897–2988.

[19] O. Koopman: The optimum distribution of effort. *Journal of Operations Research Society of America*, **1** (1953), 52–63.

[20] Y. Kobayashi, Y. Szabó and K. Takazawa: A proof of Cunningham's conjecture on restricted subgraphs and jump systems. *Journal of Combinatorial Theory*, **B102** (2012), 948–966.

[21] Y. Kobayashi and K. Takazawa: Even factors, jump systems, and discrete convexity. *Journal of Combinatorial Theory*, **B99** (2009), 139–161.

[22] J.B. Kruskal: On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, **7(1)** (1956), 48–50.

[23] E.L. Lawler: *Combinatorial Optimization: Networks and Matroids* (Dover Publications, New York, NY, 1976).

[24] N. Minamikawa and A. Shioura: Separable convex resource allocation problem with L_1-distance constraint. *Journal of the Operations Research Society of Japan*, **62(3)** (2019), 109–120.

[25] N. Minamikawa and A. Shioura: Time bounds of basic steepest descent algorithms for M-convex function minimization and related Pproblems. *Journal of the Operations Research Society of Japan*, to appear.

[26] S. Moriguchi, K. Murota, and A. Shioura: Scaling algorithms for M-convex function minimization. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, **E85-A** (2002), 922–929.

[27] S. Moriguchi and A. Shioura: On Hochbaum's proximity-scaling algorithm for the submodular resource allocation problem. *Mathematics of Operations Research*, **29** (2004), 394–397.

[28] S. Moriguchi, K. Murota, A. Tamura, and F. Tardella: Scaling, proximity, and optimization of integrally convex functions. *Mathematical Programming*, **175(1-2)** (2019), 119–154.

[29] S. Moriguchi, K. Murota, A. Tamura, and F. Tardella: Discrete midpoint convexity. *Mathematics of Operations Research*, **45(1)** (2020), 99–128.

[30] K. Murota: Convexity and Steinitz's exchange property. *Advances in Mathematics*, **124** (1996), 272–311.

[31] K. Murota: Discrete convex analysis. *Mathematical Programming*, **83** (1998), 313–371.

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

[33] K. Murota: Algorithms in discrete convex analysis. *IEICE Transactions on Information and Systems*, **E83-D** (2000), 344–352.

[34] K. Murota: *Discrete Convex Analysis* (Society for Industrial and Applied Mathematics, Philadelphia, PA, 2003).

[35] K. Murota: On steepest descent algorithms for discrete convex functions. *SIAM Journal on Optimization*, **14** (2003), 699–707.

[36] K. Murota: M-convex functions on jump systems: a general framework for minsquare graph factor problem. *SIAM Journal on Discrete Mathematics*, **20** (2006), 213–226.

[37] K. Murota: A survey of fundamental operations on discrete convex functions of various kinds. *Optimization Methods and Software* (2019), available online. doi:10.1080/10556788.2019.1692345.

[38] K. Murota: A note on M-convex functions on jump systems. *Discrete Applied Mathematics*, **289** (2021), 492–502.

[39] K. Murota and A. Shioura: M-convex function on generalized polymatroid. *Mathematics of Operations Research*, **24** (1999), 95–105.

[40] K. Murota and A. Shioura: Exact bounds for steepest descent algorithms of L-convex function minimization. *Operations Research Letters*, **42** (2014), 361–366.

[41] K. Murota and K. Tanaka: A steepest descent algorithm for M-convex functions on jump systems. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, **E89-A** (2006), 1160–1165.

[42] R.C. Prim: Shortest connection networks and some generalizations. *The Bell System Technical Journal* **36(6)** (1957), 1389–1401.

[43] A. Shioura: Minimization of an M-convex function. *Discrete Applied Mathematics*, **84** (1998), 215–220.

[44] A. Shioura: Fast scaling algorithms for M-convex function minimization with application to the resource allocation problem. *Discrete Applied Mathematics*, **134** (2004), 303–316.

[45] A. Shioura: M-convex function minimization under L_1-distance constraint. `arXiv:1809.03126`, (2018).

[46] A. Shioura and K. Tanaka: Polynomial-time algorithms for linear and convex optimization on jump systems. *SIAM Journal on Discrete Mathematics*, **21** (2007), 504–522.

[47] K. Takazawa: Optimal matching forests and valuated delta-matroids. *SIAM Journal on Discrete Mathematics*, **28** (2014), 445–467.

[48] A. Tamura: Coordinatewise domain scaling algorithm for M-convex function minimization. *Mathematical Programming*, **102** (2005), 339–354.