Department of Industrial Engineering and Economics

Working Paper

No. 2016-5

Algorithms for L-convex Function minimization: Connection Between Discrete Convex Analysis and Other Research Fields

Akiyoshi Shioura

September, 2016

Tokyo Institute of Technology

2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552, JAPAN
http://educ.titech.ac.jp/iee/

ALGORITHMS FOR L-CONVEX FUNCTION MINIMIZATION: CONNECTION BETWEEN DISCRETE CONVEX ANALYSIS AND OTHER RESEARCH FIELDS

Akiyoshi Shioura
Tokyo Institute of Technology

(September 5, 2016)

Abstract L-convexity is a concept of discrete convexity for functions defined on the integer lattice points, and plays a central role in the framework of discrete convex analysis. In this paper, we present algorithms for L-convex function minimization. Algorithms proposed independently in research fields such as discrete optimization, auction theory, and computer vision can be regarded as minimization algorithms applied to specific L-convex functions. This fact indicates the close connection between discrete convex analysis and these research fields. We then theoretically analyze the number of iterations required by some minimization algorithms, and show that the precise bounds can be given in terms of distance between the initial solution and the minimizer found by the algorithms. This fact implies that the algorithms output the "nearest" minimizer to the initial solution, and that the trajectory of solutions generated by the algorithms are "shortest paths" from the initial solution to the found minimizer. In this way, we can provide a unified viewpoint to algorithms appearing in various research fields by analyzing steepest descent algorithms for an L-convex function. Finally, we apply the analysis results to iterative auctions in auction theory. We show that the essence of the iterative auctions proposed by Ausubel (2006) lies in L-convexity, and using this fact we analyze the behavior of the iterative auctions. We also review the iterative auctions proposed by Murota–Shioura–Yang (2016), which are based on the understanding of discrete convex analysis.

Keywords: discrete convex function, discrete optimization, steepest descent method, analysis of algorithm, iterative auction

1. Introduction

L-convexity and L$^\natural$-convexity are concepts of discrete convexity for functions defined on the integer lattice points. In this paper, we consider the minimization of L-convex and L$^\natural$-convex functions. The concept of L-convexity is introduced by Murota [27]; as a variant of L-convexity, the concept of L$^\natural$-convexity (read "L-natural-convexity") is introduced by Fujishige–Murota [12]. The concepts of L/L$^\natural$-convexity are later extended to functions in continuous variables [35]. L/L$^\natural$-convexity plays a central role in discrete convex analysis initiated by Murota [27, 28], which is a theoretical framework for efficiently solvable discrete optimization problems.

Minimization of an L-convex function can be found in various research fields. For example, discrete optimization problems such as the shortest path problem and the dual of the minimum cost flow problem can be seen as special cases of L-convex function minimization [13, 39, 40]. Moreover, L-convex function minimization can also be found in other research fields such as compute vision (see Section 3.3.2), auction theory (see Section 6), inventory theory [21, 48], etc. (see also [28, 29, 33]). This fact shows that discrete convex analysis is closely connected with various research fields through L-convex function minimization.

It is known that a global minimizer of an L-convex function can be characterized by a local minimality (see Section 2.3). Therefore, minimization of an L-convex function can be solved by steepest descent algorithms [28, 29]. Such algorithms can be found in other research fields; indeed, optimization algorithms proposed independently in different research

fields such as discrete optimization, computer vision, and auction theory coincide with some variants of the steepest descent algorithms applied to special L-convex functions (see Section 3.3). Hence, we can provide a unified viewpoint to algorithms appearing in various research fields through the analysis of steepest descent algorithms for L-convex functions.

In this paper, we review the results in [13, 40] concerning the analysis of the behavior of steepest descent algorithms for L-convex function minimization. In particular, we present a theoretical results on the number of iterations required by the algorithms, where the exact number of iterations is given in terms of distance between the initial solution and the set of minimizers. This fact implies that the algorithms output the "nearest" minimizer to the initial solution, and that the trajectory of solutions generated by the algorithms are "shortest paths" from the initial solution to the found minimizer.

The results on the analysis of L-convex function minimization algorithms are interesting on their own and also useful in other research fields. For example, it is a common approach in computer vision to process some tasks for images by solving the minimization of a certain energy function, which can be seen as a special case of L-convex function minimization. It is known that such a minimization problem can be solved by simple algorithms similar to a steepest descent algorithm run fast in practice. Since the running time of the steepest descent algorithm is heavily dependent on the number of iterations of the algorithm, we can provide a theoretical guarantee for the running time of the algorithm by a theoretical bound for the number of iterations.

As an interesting application of the analysis results of the steepest descent algorithms for L-convex function minimization, we explain iterative auctions in auction theory [41, 42] (see Section 6). An iterative auction is an algorithm (mechanism, more precisely) to compute equilibrium prices of goods, where prices are iteratively changed gradually by using the bidders' reported information. We show that the iterative auctions proposed by Ausubel [2] can be regarded as special cases of L-convex function minimization algorithms. We also analyze the number of iterations required by the algorithms. In an iterative auction, it is important to obtain an estimate for the number of biddings in advance since the number of iterations in an iterative auction is equal to the number of biddings of each bidder. A deeper understanding of the iterative auctions of Ausubel [2] is provided in Murota–Shioura–Yang [41, 42] from the viewpoint of discrete convex analysis; in addition, based on the understanding, new iterative auctions are proposed in Murota–Shioura–Yang [41, 42]. We review the iterative auction algorithms by Ausubel [2] and by Murota–Shioura–Yang [41, 42], and present the bounds on the number of iterations required by the algorithms.

The organization of this paper is as follows. In Section 2 we describe the definition of L-convex function and present fundamental properties of minimizers. In Section 3 various minimization algorithms for L-convex functions are presented. In Section 4, we analyze the behavior of minimization algorithms for L-convex functions theoretically, and provide precise bounds for the number of iterations. In Section 5 we present minimization algorithms for L-convex functions in continuous variables, and show that some theoretical results for the case of discrete variables extend to the case of continuous variables. Finally in Section 6, we explain an application to iterative auctions in auction theory.

This paper is based on the manuscript [47] (in Japanese) that appeared in the proceedings of the 27th RAMP Symposium of Operations Research Society of Japan.

2. L-convex and L$^\natural$-convex Functions

In this section we present the definitions and examples of L-convex and L$^\natural$-convex functions.

2.1. Definitions

Let n be a positive integer and denote $N = \{1, 2, \ldots, n\}$. In the following, $\mathbb{Z}_+$ and $\mathbb{R}_+$ denote the sets of non-negative integers and non-negative reals, respectively.

A function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ is called an *L-convex function* [27] if it satisfies the following two properties:

$$\textbf{[Submodularity]} \quad g(p) + g(q) \geq g(p \vee q) + g(p \wedge q) \qquad (\forall p, q \in \operatorname{dom} g), \tag{2.1}$$

[Linearity in Direction of 1]

$$\exists r \in \mathbb{R} : g(p + \alpha \mathbf{1}) = g(p) + \alpha r \qquad (\forall p \in \mathbb{Z}^n,\ \forall \alpha \in \mathbb{Z}), \tag{2.2}$$

where $\operatorname{dom} g = \{p \in \mathbb{Z}^n \mid g(p) < +\infty\}$, $\mathbf{1} = (1, 1, \ldots, 1)$, and $p \vee q$, $p \wedge q$ denote the vectors obtained by the component-wise maximum and minimum of p and q, i.e.,

$$(p \vee q)(i) = \max(p(i), q(i)), \quad (p \wedge q)(i) = \min(p(i), q(i)) \qquad (i \in N).$$

Since L-convex function g has the linearity in the direction of $\mathbf{1}$ (i.e., (2.2)), the essential information of the function g is not lost by the restriction of g to an arbitrarily chosen coordinate plane. That is, for an L-convex function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ with n variables, the function $\check{g} : \mathbb{Z}^{n-1} \to \mathbb{R} \cup \{+\infty\}$ with $n - 1$ variables given by

$$\check{g}(q(1), \ldots, q(n-1)) = g(q(1), \ldots, q(n-1), 0)$$

has essentially the same information as g, which we call an *$L^\natural$-convex function* [12]. Hence, a function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ is $\mathrm{L}^\natural$-convex function if the function $\tilde{g} : \mathbb{Z}^{n+1} \to \mathbb{R} \cup \{+\infty\}$ with $n + 1$ variables given by

$$\tilde{g}(q(1), \ldots, q(n), q(n+1)) = g(q(1) - q(n+1), \ldots, q(n) - q(n+1)) \quad (q \in \mathbb{Z}^{n+1}) \tag{2.3}$$

is an L-convex function. Note that the function $\tilde{g}$ given by (2.3) always satisfies the property (2.2). Hence, $\tilde{g}$ is an L-convex function if and only if $\tilde{g}$ satisfies the submodularity (2.1). That is, $\mathrm{L}^\natural$-convexity of a function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ is equivalent to the submodularity of $\tilde{g}$.

It is known that $\mathrm{L}^\natural$-convexity is equivalent to the following property (see Figure 2.1):

[Translation Submodularity]

$$g(p) + g(q) \geq g((p - \alpha \mathbf{1}) \vee q) + g(p \wedge (q + \alpha \mathbf{1})) \quad (\forall p, q \in \operatorname{dom} g,\ \forall \alpha \in \mathbb{Z}_+). \tag{2.4}$$

$\mathrm{L}^\natural$-convexity is also equivalent to a discrete version of mid-point convexity:

[Discrete Mid-point Convexity]

$$g(p) + g(q) \geq g\left(\left\lceil \frac{p+q}{2} \right\rceil\right) + g\left(\left\lfloor \frac{p+q}{2} \right\rfloor\right) \qquad (\forall p, q \in \operatorname{dom} g), \tag{2.5}$$

where $\left\lceil \frac{p+q}{2} \right\rceil$ and $\left\lfloor \frac{p+q}{2} \right\rfloor$ denote the integral vectors obtained by component-wise rounding up and rounding down of the vector $\frac{p+q}{2}$.

From the discussion above, we obtain the following equivalence.

Theorem 2.1 ([12]). *For a function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ the following three conditions are equivalent:*

(a) *$L^\natural$-convexity, i.e., L-convexity of $\tilde{g}$ in* (2.3),
(b) *translation submodularity* (2.4),
(c) *discrete mid-point convexity* (2.5). □

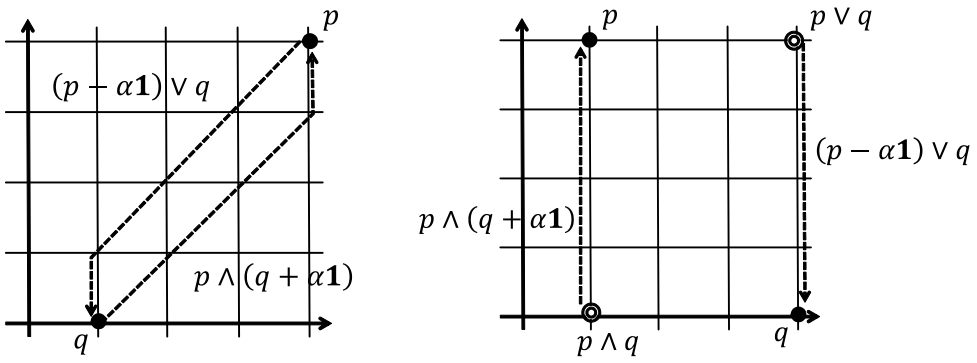

Figure 1: Vectors $(p - \alpha\mathbf{1}) \vee q$ and $p \wedge (q + \alpha\mathbf{1})$ in the translation submodularity (2.4): (Left) Case of $p \geq q$, (Right) Case of $p(1) > q(1)$ and $p(2) < q(2)$. Trajectory of the two vectors when α is increased from 0 is drawn by dashed arrows. Note that if $p \leq q$, then $(p - \alpha\mathbf{1}) \vee q = q$ and $p \wedge (q + \alpha\mathbf{1}) = p$ for every $\alpha \geq 0$.

The concept of $\mathrm{L}^\natural$-convex function is essentially equivalent to the concept of L-convex function by its definition. On the other hand, an L-convex function can be seen as a special case of an $\mathrm{L}^\natural$-convex function. Indeed, translation submodularity of an L-convex function can be shown as follows, where $p, q \in \operatorname{dom} g$ and $\alpha \in \mathbb{Z}_+$:

$$\begin{aligned} g(p) + g(q) &= g(p - \alpha\mathbf{1}) + \alpha r + g(q) \quad \text{(by (2.2))} \\ &\geq g((p - \alpha\mathbf{1}) \vee q) + g((p - \alpha\mathbf{1}) \wedge q) + \alpha r \quad \text{(by (2.1))} \\ &= g((p - \alpha\mathbf{1}) \vee q) + g(p \wedge (q + \alpha\mathbf{1})) \quad \text{(by (2.2))}. \end{aligned}$$

2.2. Examples

We present several examples of L-convex and $\mathrm{L}^\natural$-convex functions. See Section 3.3 for more examples.

Example 2.2 (linear functions)**.** For a real vector $c \in \mathbb{R}^n$, the linear function given by $g(p) = \sum_{i=1}^n c(i)p(i)$ $(p \in \mathbb{Z}^n)$ is L-convex as well as $\mathrm{L}^\natural$-convex. Moreover, given $a(i,j) \in \mathbb{Z} \cup \{-\infty\}$ $(i, j \in N,\ i \leq j)$ and $b(i,j) \in \mathbb{Z} \cup \{+\infty\}$ $(i, j \in N,\ i \leq j)$, the function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ given by

$$\begin{aligned} \operatorname{dom} g = \{p \in \mathbb{Z}^n \mid\ & a(i,i) \leq p(i) \leq b(i,i)\ (i \in N), \\ & a(i,j) \leq p(i) - p(j) \leq b(i,j)\ (i, j \in N,\ i < j)\}, \end{aligned}$$

$$g(p) = \sum_{i=1}^n c(i)p(i) \qquad (p \in \operatorname{dom} g)$$

is $\mathrm{L}^\natural$-convex, provided that $\operatorname{dom} g$ is nonempty; the function g is L-convex if $a(i,i) = -\infty$ $(i \in N)$ and $b(i,i) = +\infty$ $(i \in N)$, in addition. □

Example 2.3 (quadratic functions)**.** A quadratic function $g(p) = p^\top Ap$ $(p \in \mathbb{Z}^n)$ given by an $n \times n$ real symmetric matrix A is $\mathrm{L}^\natural$-convex if and only if each off-diagonal component $a(i,j)$ $(i, j \in N,\ i \neq j)$ is non-positive and the sum of components in each row (or column) $\sum_{j=1}^n a(i,j)$ $(i \in N)$ is non-negative [37]. □

Example 2.4 (maximum-value functions)**.** Define

$$\begin{aligned} g(p) &= \max\{p(1), p(2), \ldots, p(n)\} \qquad (p \in \mathbb{Z}^n), \\ g_0(p) &= \max\{0, p(1), p(2), \ldots, p(n)\} \qquad (p \in \mathbb{Z}^n). \end{aligned}$$

Then, g is an L-convex function and g_0 is an L$^\natural$-convex function. We also consider the following functions associated with real numbers $a(0), a(1), \ldots, a(n), b(0), b(1), \ldots, b(n) \in \mathbb{R}$:

$$\tilde{g}(p) = \max_{1\le i\le n}\{p(i)+a(i)\} - \min_{1\le i\le n}\{p(i)+b(i)\} \qquad (p \in \mathbb{Z}^n),$$
$$\tilde{g}_0(p) = \max_{0\le i\le n}\{p(i)+a(i)\} - \min_{0\le i\le n}\{p(i)+b(i)\} \qquad (p \in \mathbb{Z}^n),$$

where $p(0) = 0$. Function $\tilde{g}$ is L-convex and $\tilde{g}_0$ is L$^\natural$-convex [33]. □

2.3. Properties of minimizers

We show fundamental properties of minimizers of L-convex and L$^\natural$-convex functions.

For an L$^\natural$-convex function, the global minimality can be characterized by the local minimality with respect to a certain neighborhood. For a function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$, $p \in \operatorname{dom} g$, and $d \in \mathbb{Z}^n$, we denote by $g'(p; d)$ the *slope* of g at p in the direction d, i.e., $g'(p; d) = g(p + d) - g(p)$. We denote by $\arg\min g$ the set of minimizers of g, i.e.,

$$\arg\min g = \{p \in \mathbb{Z}^n \mid g(p) \le g(q)\ (\forall q \in \mathbb{Z}^n)\}.$$

Theorem 2.5 ([27, 28, 29]). *Let $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be an L$^\natural$-convex function and $p \in \operatorname{dom} g$. Then, we have $p \in \arg\min g$ if and only if $g'(p; d) \ge 0$ holds for every $d \in \{0, +1\}^n \cup \{0, -1\}^n$.* □

For $p \in \operatorname{dom} g$, a vector $d = d_* \in \{0, +1\}^n \cup \{0, -1\}^n$ minimizing $g'(p; d)$ is called a *steepest descent direction* of g at p. Since $g'(p; \mathbf{0}) = 0$, every steepest descent direction d_* satisfies $g'(p; d_*) \ge g'(p; \mathbf{0}) = 0$. Therefore, we have the following equivalence:

$$\begin{aligned} & g'(p; d) \ge 0 \quad (\forall d \in \{0, +1\}^n \cup \{0, -1\}^n) \\ \iff\ & g'(p; d_*) = 0 \quad (\forall d_* : \text{steepest descent direction at } p) \\ \iff\ & d_* = \mathbf{0} \text{ is a steepest descent direction at } p. \end{aligned}$$

If we know in advance that a vector $p \in \operatorname{dom} g$ is a lower bound (or an upper bound) of some minimizer, then we can characterize the global minimality of p by using the local minimality with respect to a smaller neighborhood.

Theorem 2.6 (cf. [40]). *Let $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be an L$^\natural$-convex function, and $p \in \operatorname{dom} g$ be a vector that is a lower bound of some minimizer p_* of g (i.e., $p \le p_*$ holds).*
(i) *There exists a steepest descent direction d of g at p such that $d \ge 0$ (i.e., $d \in \{0, +1\}^n$).*
(ii) *$p \in \arg\min g$ holds if and only if $g'(p; d) \ge 0$ holds for every $d \in \{0, +1\}^n$.* □

Theorem 2.7 (cf. [40]). *Let $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be an L$^\natural$-convex function, and $p \in \operatorname{dom} g$ be a vector that is an upper bound of some minimizer p_* of g (i.e., $p \ge p_*$ holds).*
(i) *There exists a steepest descent direction d of g at p such that $d \le 0$ (i.e., $d \in \{0, -1\}^n$).*
(ii) *$p \in \arg\min g$ holds if and only if $g'(p; d) \ge 0$ holds for every $d \in \{0, -1\}^n$.* □

For an L-convex function g that has a minimizer p_*, the real number r in the condition (2.2) must be equal to zero, from which follows that $p_* + \alpha\mathbf{1}$ is also a minimizer of g for every $\alpha \in \mathbb{Z}$. Therefore, every $p \in \operatorname{dom} g$ is a lower bound of some minimizer as well as an upper bound of another minimizer, which, together with Theorems 2.6 and 2.7, implies the following property.

Theorem 2.8. *For an L-convex function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ and a vector $p \in \operatorname{dom} g$, the following three conditions are equivalent:*

(a) $p \in \arg\min g$, (b) $g'(p; d) \ge 0\ (\forall d \in \{0, +1\}^n)$, (c) $g'(p; d) \ge 0\ (\forall d \in \{0, -1\}^n)$. □

We then consider minimizers of L$^\natural$-convex functions. Since every L$^\natural$-convex function satisfies submodularity, the following properties hold.

Theorem 2.9. *Let $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be an $L^\natural$-convex function.*
(i) *For $p, q \in \arg\min g$, we have $p \vee q, p \wedge q \in \arg\min g$.*
(ii) *Minimal and maximal minimizers of g are uniquely determined if they exist. In particular, if $\arg\min g$ is bounded, then minimal and maximal minimizers of g are uniquely determined.* □

It should be noted that there may exist many minimal and maximal minimizers for a general function.

3. Minimization Algorithms

In this section we explain minimization algorithms for L-convex and $L^\natural$-convex functions. Since the minimization of an L-convex function is a special case of the minimization of an $L^\natural$-convex function, we mainly deal with the latter in this section. Throughout this section, we assume that $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ is an $L^\natural$-convex function with bounded $\operatorname{dom} g$.

3.1. Fundamental algorithms

We present three minimization algorithms. The first algorithm finds a minimizer by iteratively moving a solution along steepest descent directions.

Algorithm 1 (GREEDY)**.**
Step 0: Select an initial solution $p_0 \in \operatorname{dom} g$ arbitrarily, and set $p := p_0$.
Step 1: If $g'(p; d) \geq 0$ holds for every $d \in \{0, +1\}^n \cup \{0, -1\}^n$, then output p and stop.
Step 2: Find $d = d_* \in \{0, +1\}^n \cup \{0, -1\}^n$ minimizing $g'(p; d)$. Set $p := p + d_*$ and go to Step 1. □

By Theorem 2.5, the algorithm GREEDY outputs a minimizer when applied to $L^\natural$-convex function g. Since the function value $g(p)$ decreases strictly in each iteration and $\operatorname{dom} g$ is bounded, the algorithm GREEDY terminates in a finite number of iterations. Note that the computation of a steepest descent direction in each iteration can be done in polynomial time (in n) by reduction to the minimization of submodular set functions ρ_p^+, ρ_p^- given by

$$\rho_p^+(X) = g'(p; +e_X), \qquad \rho_p^-(X) = g'(p; -e_X) \qquad (X \subseteq N),$$

where $e_X \in \{0, +1\}^n$ denotes the characteristic vector of $X \subseteq N$; see [22, 45] for the strongly-polynomial time algorithms for submodular set function minimization.

If a lower bound of some minimizer of g is available as the initial solution, then an increasing-type steepest descent algorithm can be applied by Theorem 2.6. Here, an "increasing-type" algorithm means that in the algorithm each component of vector p is monotonically increasing (non-decreasing).

Algorithm 2 (GREEDYUP)**.**
Step 0: Select an initial solution $p_0 \in \operatorname{dom} g$ that is a lower bound of some minimizer of g, and set $p := p_0$.
Step 1: If $g'(p; d) \geq 0$ holds for every $d \in \{0, +1\}^n$, then output p and stop.
Step 2: Find $d = d_* \in \{0, +1\}^n$ minimizing $g'(p; d)$. Set $p := p + d_*$ and go to Step 1. □

Similarly, if an upper bound of some minimizer of g is available as the initial solution, then a decreasing-type steepest descent algorithm can be applied by Theorem 2.7.

Algorithm 3 (GREEDYDOWN)**.**
Step 0: Select an initial solution $p_0 \in \operatorname{dom} g$ that is an upper bound of some minimizer of g, and set $p := p_0$.
Step 1: If $g'(p; d) \geq 0$ holds for every $d \in \{0, -1\}^n$, then output p and stop.
Step 2: Find $d = d_* \in \{0, -1\}^n$ minimizing $g'(p; d)$. Set $p := p + d_*$ and go to Step 1. □

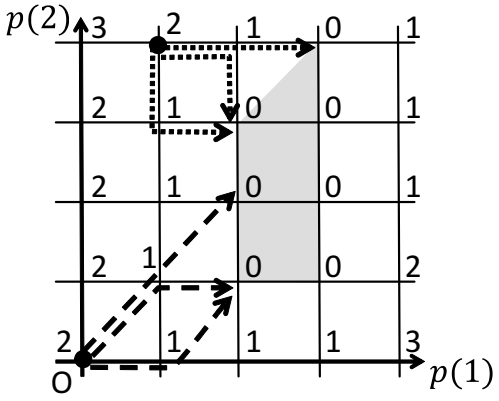

Figure 2: The behavior of the steepest descent algorithms. Each numbers associated with each integer lattice point denotes the function value of g at the point. The shaded region shows the set of minimizers of g.

Remark 3.1. The algorithm GreedyUp is originally proposed as a minimization algorithm of L-convex functions (see, e.g., [29, Section 10.3.1]). If an L-convex function g has a minimizer, then for every initial vector $p_0 \in \operatorname{dom} g$ there exists some minimizer p_* of g such that $p_* \geq p_0$ (see the discussion just before Theorem 2.8). Hence, a minimizer of L-convex function g can be obtained by GreedyUp with an arbitrarily chosen initial vector p_0. Similarly, GreedyDown can be also applied to a minimizer of an L-convex function. □

Remark 3.2. The behavior of vector $p \in \mathbb{R}^n$ in the algorithm Greedy applied to an L$^\natural$-convex function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ coincides with the behavior of the original variable vector $p \in \mathbb{R}^n$ in GreedyUp applied to L-convex function $\tilde{g} : \mathbb{Z}^{n+1} \to \mathbb{R} \cup \{+\infty\}$ with $n+1$ variables given by (2.3). □

3.2. A numerical example

To illustrate the behavior of the steepest descent algorithms explained above, we consider a numerical example of L$^\natural$-convex function $g : \mathbb{Z}^2 \to \mathbb{R} \cup \{+\infty\}$ given as follows (see Figure 2):

$$\begin{aligned} &\operatorname{dom} g = \{(p(1), p(2)) \in \mathbb{Z}^2 \mid 0 \leq p(i) \leq 4 \ (i = 1, 2)\}, \\ &g(p(1), p(2)) = \max(0,\ -p(1) + 2,\ -p(2) + 1,\ p(1) - 3, \\ &\qquad\qquad -p(1) + p(2) - 1,\ 2p(1) - p(2) - 5) \qquad ((p(1), p(2)) \in \operatorname{dom} g). \end{aligned}$$

L$^\natural$-convexity of this function can be checked by using the translation submodularity (2.4) or discrete-midpoint convexity (2.5) (see Theorem 2.8).

Suppose that the algorithm Greedy is applied to the function g with the initial solution $p_0 = (1, 4)$. Then, the trajectory of vector p is given by one of the three dotted arrows in Figure 2 starting from $(1, 4)$. In particular, the minimizer found by the algorithm is either of $(3, 4)$ and $(2, 3)$.

We then suppose that the algorithm Greedy or GreedyUp is applied to g with another initial solution $p_0 = (0, 0)$. Then, the trajectory of vector p is given by one of the three dashed arrows in Figure 2 starting from $(0, 0)$. In particular, the minimizer found by the algorithm is either of $(2, 1)$ and $(2, 2)$.

3.3. Steepest descent algorithms in other research fields

We show that various optimization algorithms proposed independently in other research fields can be seen as steepest descent algorithms in Section 3.1 applied to special L$^\natural$-convex (or L-convex) functions.

3.3.1. Hassin's algorithm for the minimum cost flow problem

Given a directed graph $G = (V, E)$, non-negative edge capacity $c(e) \in \mathbb{R}_+$ $(e \in E)$, and edge cost $\gamma(e) \in \mathbb{R}$ $(e \in E)$, the minimum cost flow problem (minimum cost circulation problem) is formulated as follows:

$$\begin{aligned} &\text{Minimize} && \sum_{(u,v)\in E} \gamma(u,v)x(u,v) \\ &\text{subject to} && \sum_{v:(u,v)\in E} x(u,v) - \sum_{v:(v,u)\in E} x(v,u) = 0 \quad (u \in V), \\ & && 0 \le x(u,v) \le c(u,v) \quad ((u,v) \in E). \end{aligned}$$

This is a linear programming problem, and its dual is given as

$$\begin{aligned} &\text{Maximize} && g_{\mathrm{H}}(p) \equiv \sum_{(u,v)\in E} c(u,v)\min\{0, p(u) - p(v) + \gamma(u,v)\} \\ &\text{subject to} && p(v) \in \mathbb{R} \quad (v \in V). \end{aligned}$$

Hassin's algorithm [20] solves the minimum cost flow problem by finding an optimal solution of the dual problem, instead of solving the primal problem directly.

Assume that edge cost $\gamma(u, v)$ is integer for each $(u, v) \in E$. Then, it can be shown that the dual problem has an integral optimal solution. Hence, we may restrict the values of dual variables $p(v)$ $(v \in V)$ to integers, and the objective function g_{H} can be regarded as a function in discrete variables. It is known that this function g_{H} is an L-concave function in discrete variables (see [28, 29, 32, 38]). Moreover, Hassin's algorithm coincides with the algorithm GreedyUp (more precisely, the algorithm GreedyUpMinimal in Section 3.4) applied to L-convex function $-g_{\mathrm{H}}$ (see [39] for details).

For the minimum cost submodular flow problem, which is a generalization of the minimum cost flow problem, Chung–Tcha [6] proposed an algorithm by generalizing Hassin's algorithm under the integrality assumption for the problem input. This algorithm can be also seen as a (variant of) the algorithm GreedyUp applied to an L-convex function arising from the dual of the minimum cost submodular flow problem (see [39] for details).

3.3.2. Discrete optimization approach in computer vision

Given an undirected graph $G = (V, E)$ and univariate convex functions $\varphi_v : \mathbb{Z} \to \mathbb{R} \cup \{+\infty\}$ $(v \in V)$, $\psi_{uv} : \mathbb{Z} \to \mathbb{R} \cup \{+\infty\}$ $((u, v) \in E)$, consider the following optimization problem:

$$\begin{aligned} \text{(P):}\quad &\text{Minimize} && g_{\mathrm{CV}}(p) \equiv \sum_{v\in V} \varphi_v(p(v)) + \sum_{(u,v)\in E} \psi_{uv}(p(v) - p(u)) \\ &\text{subject to} && p(v) \in \mathbb{Z} \quad (v \in V). \end{aligned}$$

Here, we say that a univariate function $\varphi : \mathbb{Z} \to \mathbb{R} \cup \{+\infty\}$ is convex if $2\varphi(\alpha) \le \varphi(\alpha - 1) + \varphi(\alpha + 1)$ holds for every $\alpha \in \mathbb{Z}$ with $\varphi(\alpha) < +\infty$. The objective function g_{CV} of the problem (P) is an $\mathrm{L}^\natural$-convex function. Moreover, if the first term $\sum_{v\in V} \varphi_v(p(v))$ is removed from the function g_{CV}, then the resulting function is an L-convex function (see [28, 29, 32, 38]).

The problem (P) is used to process various tasks in computer vision such as panoramic image stitching [49], image restoration [5], minimization of total variation [8], phase unwrapping in SAR images [3], etc. In these problems, the vertex set V of the graph G corresponds to the set of pixels in a given image, and edges of the graph represent the neighborhood relation of pixels. Variable $p(v)$ represents the label of the pixel v such as disparity and intensity. The function φ_v for each $v \in V$ is called the data term and used to evaluate the

label of the pixel v. The function ψ_{uv} for $(u, v) \in E$ is called the smoothness term and used to evaluate the difference of the labels for a pair of adjacent pixels u and v. The objective function g_{CV} of the problem (P) is called an energy function in the context of Markov random field [17], and its minimizer corresponds to a set of labels maximizing the a-posteriori probability.

In computer vision, many algorithms for the problem (P) have been proposed (see, e.g., [3, 24]). In particular, Kolmogorov [24] proposed a general framework of minimization algorithms that includes the algorithms Greedy, GreedyUp, and GreedyDown as special cases. An algorithm by Bioucas-Dias–Valadão [3] coincides with the algorithm GreedyUp applied to the problem (P) without the first term $\sum_{v \in V} \varphi_v(p(v))$.

3.3.3. Iterative auction in auction theory

Consider an auction with multiple indivisible items. In such an auction, the auctioneer needs to find "appropriate" prices of items as well as "appropriate" allocation of items to bidders. An ascending auction due to Ausubel [2], which is an algorithm for computing such "appropriate" prices by using the Lyapunov function, can be regarded as an application of the algorithm GreedyUp to a special $\mathrm{L}^\natural$-convex function. We discuss this topic in more details in Section 6.

3.4. Computation of minimal and maximal minimizers

For an $\mathrm{L}^\natural$-convex function with bounded effective domain, minimal and maximal minimizers are uniquely determined by Theorem 2.9. We show that unique minimal and maximal minimizers can be computed by using variants of the algorithms in Section 3.1.

We first consider the minimal minimizer. The minimal minimizer of function g is the same as the unique minimizer of the function

$$g_\varepsilon(p) = g(p) + \varepsilon \sum_{i=1}^{n} p(i),$$

where ε is a sufficiently small positive number. Since the function g_ε is also an $\mathrm{L}^\natural$-convex function, its minimizer can be computed by the algorithm GreedyUp, although we need to select the value of ε appropriately so that a minimizer of g_ε is also a minimizer of g. In fact, we do not need to compute such ε explicitly. The behavior of the algorithm GreedyUp applied to the function g_ε with a sufficiently small $\varepsilon > 0$ coincides with the behavior of GreedyUpMinimal to be explained below applied to the original function g. The difference between GreedyUpMinimal and GreedyUp is in the choices of the initial solution and a steepest descent direction in each iteration.

Algorithm 4 (GreedyUpMinimal)**.**

Step 0: Select an initial solution $p_0 \in \operatorname{dom} g$ that is a lower bound of the minimal minimizer of g, and set $p := p_0$.

Step 1: If $g'(p; d) \geq 0$ holds for every $d \in \{0, +1\}^n$, then output p and stop.

Step 2: Find a (unique) minimal minimizer $d = d_* \in \{0, +1\}^n$ of $g'(p; d)$. Set $p := p + d_*$ and go to Step 1. □

Note that a minimal minimizer $d = d_* \in \{0, +1\}^n$ of $g'(p; d)$ found in Step 2 is uniquely determined due to the submodularity of g.

Proposition 3.3. *For an $L^\natural$-convex function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ with bounded* $\operatorname{dom} g$*, the algorithm* GreedyUpMinimal *outputs the unique minimal minimizer.* □

Based on a similar idea, the algorithm Greedy can also be used to find the minimal minimizer by modifying the condition of the termination and the choice of a steepest descent direction.

Algorithm 5 (GreedyMinimal)**.**
Step 0: Select an initial solution $p_0 \in \operatorname{dom} g$ arbitrarily and set $p := p_0$.
Step 1: Find a (unique) minimal minimizer $d = d_* \in \{0, +1\}^n \cup \{0, -1\}^n$ of $g'(p; d)$.
Step 2: If $d_* = \mathbf{0}$, then output p and stop. Otherwise, set $p := p + d_*$ and go to Step 1. □

Proposition 3.4. *For an $L^\natural$-convex function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ with bounded* $\operatorname{dom} g$*, the algorithm* GreedyMinimal *outputs the unique minimal minimizer.* □

By modifying the algorithm GreedyDown in a similar way, we can obtain the algorithm GreedyDownMinimal that finds the minimal minimizer.

Remark 3.5. Iteration of the algorithm GreedyMinimal does not necessarily stop even if the current vector p is a minimizer of g. For example, let us consider the $L^\natural$-convex function g in Section 3.2. If we apply GreedyMinimal with the initial solution $p_0 = (1, 4)$, then the trajectory of p is given as $(1, 4) \to (1, 3) \to (2, 3) \to (2, 2) \to (2, 1)$, which reaches the minimal minimizer $(2, 1)$ after passing through the minimizers $(2, 3)$ and $(2, 2)$. □

We then consider the computation of the maximal minimizer. For a general function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$, a vector p is a maximal minimizer of g if and only if $-p$ is a minimal minimizer of the function $h : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ defined by $h(q) = g(-q)$ $(q \in \mathbb{Z}^n)$. Moreover, function g is $L^\natural$-convex if and only if function h is $L^\natural$-convex. Hence, an algorithm for computing the maximal minimizer of an $L^\natural$-convex function can be easily obtained by modification of an algorithm for the minimal minimizer of an $L^\natural$-convex function. For example, we can obtain the following algorithm for the unique maximal minimizer of $L^\natural$-convex function g by rewriting the algorithm GreedyUpMinimal applied to $L^\natural$-convex function h in terms of the function g; the resulting algorithm can be seen as a variant of the algorithm GreedyDown.

Algorithm 6 (GreedyDownMaximal)**.**
Step 0: Select an initial solution $p_0 \in \operatorname{dom} g$ that is an upper bound of the maximal minimizer of g, and set $p := p_0$.
Step 1: If $g'(p; d) \geq 0$ holds for every $d \in \{0, -1\}^n$, then output p and stop.
Step 2: Find a (unique) maximal minimizer $d = d_* \in \{0, -1\}^n$ of $g'(p; d)$.
Set $p := p + d_*$ and go to Step 1. □

Proposition 3.6. *For an $L^\natural$-convex function $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ with bounded* $\operatorname{dom} g$*, the algorithm* GreedyDownMaximal *outputs the unique maximal minimizer.* □

By using similar ideas, we can obtain the algorithms GreedyMaximal and GreedyUpMaximal for the maximal minimizer as variants of Greedy and GreedyUp.

3.5. Use of long step length

In all the algorithms explained in Sections 3.1 and 3.4, the current vector p repeats moving along a certain direction by unit step length. We consider a modification of the algorithms where the vector p moves along the same steepest descent direction d_* as far as the value of the slope $g'(p; d_*)$ in the steepest descent direction does not change.

For example, if this modification is applied to the algorithm GreedyUp, the following algorithm is obtained. For a vector $p \in \operatorname{dom} g$ and a direction $d \in \{0, +1\}^n \cup \{0, -1\}^n$, we define the value $\bar{c}(p; d) \in \mathbb{Z} \cup \{+\infty\}$ by

$$\bar{c}(p; d) = \sup\{\lambda \in \mathbb{Z}_+ \mid g(p + \lambda d) - g(p) = \lambda\, g'(p; d)\}.$$

Algorithm 7 (GreedyUp-LS)**.**
Step 0: Select an initial solution $p_0 \in \operatorname{dom} g$ that is a lower bound of some minimizer of g, and set $p := p_0$.
Step 1: If $g'(p; d) \geq 0$ holds for every $d \in \{0, +1\}^n$, then output p and stop.

Step 2: Find $d = d_* \in \{0, +1\}^n$ minimizing $g'(p; d)$. Set $\lambda := \bar{c}(p; d_*)$, $p := p + \lambda d_*$, and go to Step 1. □

As shown in Section 4.2, this algorithm can be seen as a special implementation of GreedyUp. In particular, the output of GreedyUp-LS is a minimizer of g.

Computation of a steepest descent direction is essentially equivalent to the minimization of a submodular set function (see Section 3.1), for which we require quite large running time to obtain a minimizer (see, e.g., [22, 45]). Therefore, the use of long step length helps to reduce the total running time. Note that the computation of the step length $\bar{c}(p; d_*)$ can be done by binary search; for the special cases of $\mathrm{L}^\natural$-convex functions shown in Sections 2.2 and 3.3, step length can be computed more easily.

Long step length can be used to other steepest descent algorithms in a similar way; we omit the details in this paper.

4. Analysis of Number of Iterations in Steepest Descent Algorithms

We analyze the number of iterations required by the algorithms in Section 3. In Section 4.1, the exact bounds for the steepest descent algorithms and its variants are provided in terms of the distance to minimizers. In Section 4.2, we consider an increasing-type steepest descent algorithm using long step length, and obtain an upper bound of the number of iterations by using the slope of steepest descent directions. While other minimization algorithms can be analyzed in a similar way, we omit the details in this paper.

In the following, let $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ be an $\mathrm{L}^\natural$-convex function with bounded $\operatorname{dom} g$.

4.1. Analysis by using the distance to minimizers

We analyze the number of iterations of algorithms by using the distance from the current vector $p \in \mathbb{Z}^n$ to minimizers. In the analysis we use two kinds of distance. We define a function $\hat{\mu} : \mathbb{Z}^n \to \mathbb{Z}_+ \cup \{+\infty\}$ by

$$\hat{\mu}(p) = \begin{cases} \min\{\|p_* - p\|_\infty \mid p_* \in \arg\min g,\ p_* \geq p\} & (\text{if } \{p_* \in \arg\min g \mid p_* \geq p\} \neq \emptyset), \\ +\infty & (\text{otherwise}). \end{cases} \tag{4.1}$$

That is, $\hat{\mu}(p)$ is the ℓ_∞-distance between a vector p and a nearest minimizer p_* under the condition $p_* \geq p$. We also define a function $\mu : \mathbb{Z}^n \to \mathbb{Z}_+$ by

$$\mu(p) = \min\{\|p_* - p\|^+_\infty + \|p_* - p\|^-_\infty \mid p_* \in \arg\min g\} \qquad (p \in \mathbb{Z}^n), \tag{4.2}$$

where

$$\begin{aligned} \|p_* - p\|^+_\infty &= \max_{i \in N} \max(0, p_*(i) - p(i)), \\ \|p_* - p\|^-_\infty &= \max_{i \in N} \max(0, -p_*(i) + p(i)). \end{aligned}$$

Function $\mu(p)$ gives a distance between p and a nearest minimizer p_*.

Due to the $\mathrm{L}^\natural$-convexity of g, the functions $\mu(p)$ and $\hat{\mu}(p)$ have the following relationship; the proof is given in Appendix.

Proposition 4.1 (K. Murota)**.** *$\hat{\mu}(p) = \mu(p)$ holds for a vector $p \in \mathbb{Z}^n$ such that $\{q \in \arg\min g \mid q \geq p\} \neq \emptyset$.* □

Example 4.2. Consider the example of an $\mathrm{L}^\natural$-convex function in Section 3.2 (see Figure 2). We have $\hat{\mu}(p) = \mu(p) = 2$ for $p = (1, 4)$. The value $\hat{\mu}(p) = 2$ is attained by the minimizer $(3, 4)$, while $\mu(p) = 2$ is attained by both of the two minimizers $(3, 4)$ and $(2, 3)$. □

We first analyze the two increasing-type steepest descent algorithms, GreedyUp and GreedyUpMinimal. Suppose that a vector $p \in \operatorname{dom} g$ is a lower bound of some minimizer of function g. We consider the unique minimal minimizer p_* of g under the condition $p_* \geq p$, which we denote by $\hat{p}$. Recall that in an increasing-type algorithm each component of vector p increases monotonically. The following conditions are desirable properties of increasing-type algorithms (for an unknown $\hat{p}$):

(a) for each $i \in \arg\max_{j \in N}\{\hat{p}(j) - p(j)\}$, the value of $p(i)$ is increased if $\max_{j \in N}\{\hat{p}(j) - p(j)\} > 0$.

(b) for each $i \in N$ with $\hat{p}(i) - p(i) = 0$, the value of $p(i)$ is unchanged.

The next proposition shows that if vector p is updated by using a steepest descent direction $d \in \{0, +1\}^n$, then the condition (a) is satisfied; moreover, the condition (b) is also satisfied if d is a minimal steepest descent direction, in particular. We denote $\operatorname{supp}^+(d) = \{i \in N \mid d(i) > 0\}$ for a vector $d \in \mathbb{Z}^n$.

Proposition 4.3 ([40]). *Let $p \in \operatorname{dom} g$ be a vector that is a lower bound of some minimizer of g, and $\hat{p} \in \arg\min g$ be a (unique) minimal minimizer of g under the condition $\hat{p} \geq p$.*
(i) *For every steepest descent direction $d \in \{0, +1\}^n$ at p, we have $\arg\max_{j \in N}\{\hat{p}(j) - p(j)\} \subseteq \operatorname{supp}^+(d)$. In addition, $p_* = \hat{p} \vee (p + d)$ is a (unique) minimal minimizer under the condition $p_* \geq p + d$ and satisfies $\|p_* - (p + d)\|_\infty = \|\hat{p} - p\|_\infty - 1$.*
(ii) *For a (unique) minimal steepest descent direction $d \in \{0, +1\}^n$ at p, we have*

$$\operatorname{supp}^+(d) \cap \{i \in N \mid \hat{p}(i) = p(i)\} = \emptyset, \quad \hat{p} \geq p + d, \quad \|\hat{p} - (p + d)\|_\infty = \|\hat{p} - p\|_\infty - 1.$$

□

From the proposition above, the exact number of iterations in GreedyUp and GreedyUpMinimal can be obtained. In general, if we update vector p by increasing each component at most one, then the value $\hat{\mu}(p)$ decreases by at most one. Hence, $\hat{\mu}(p_0)$ gives a lower bound for the number of iterations required by any increasing-type algorithm with the initial solution p_0. In fact, the number of iterations of GreedyUp is exactly equal to $\hat{\mu}(p_0)$. We define

$$\Phi_g = \max\{\|p - q\|_\infty \mid p, q \in \operatorname{dom} g\},$$

which gives the "size" of the effective domain of g.

Theorem 4.4 (cf. [40]).
(i) *If vector p is updated in an iteration of the algorithm GreedyUp, then the value $\hat{\mu}(p)$ decreases by one.*
(ii) *The number of update of vector p in GreedyUp is equal to $\hat{\mu}(p_0)$ and at most Φ_g.*
(iii) *The minimizer $p_* \in \arg\min g$ found by GreedyUp satisfies $\|p_* - p_0\|_\infty = \hat{\mu}(p_0)$.* □

This theorem implies that the trajectory of vector p generated by GreedyUp is a shortest path (with respect to ℓ_∞-distance) from the initial solution p_0 to a nearest minimizer p_* under the condition $p_* \geq p_0$.

Corollary 4.5 ([40, 42]). *The algorithm GreedyUpMinimal outputs the unique minimal minimizer p_* after vector p is updated $\|p_* - p_0\|_\infty$ times.* □

Example 4.6. We consider the example for the behavior of the algorithm in Section 3.2. If we apply the algorithm GreedyUp with the initial solution $p_0 = (0, 0)$, then the output of the algorithm is either $p_* = (2, 1)$ or $p_* = (2, 2)$, each of which is a nearest minimizer to p_0 (i.e., a minimizer with the minimum value of $\|p_* - p_0\|_\infty$) under the condition $p_* \geq p_0$. The number of iterations required by the algorithm is equal to $\hat{\mu}(p_0) = 3$, the distance from p_0 to a nearest minimizer. Three possible trajectories of p in the algorithm GreedyUp are drawn by dashed arrows in Figure 2, each of which is a shortest path from the initial

solution $p_0 = (0, 0)$ to a nearest minimizer. Also, if we apply GreedyUpMinimal, then the trajectory of p is given as $(0, 0) \to (1, 0) \to (2, 1)$, and its output $p_* = (2, 1)$ is the minimal minimizer. □

A similar result can be obtained for the decreasing-type steepest descent algorithms GreedyDown and GreedyDownMaximal. For a vector $p \in \operatorname{dom} g$ that is an upper bound of some minimizer of $\mathrm{L}^\natural$-convex function g, we define the value $\check{\mu}(p) \in \mathbb{Z}_+$ by

$$\check{\mu}(p) = \begin{cases} \min\{\|p_* - p\|_\infty \mid p_* \in \arg\min g,\ p_* \leq p\} & (\text{if } \{p_* \in \arg\min g \mid p_* \leq p\} \neq \emptyset), \\ +\infty & (\text{otherwise}). \end{cases}$$

Proposition 4.7 (K. Murota)**.** *$\check{\mu}(p) = \mu(p)$ holds for a vector $p \in \mathbb{Z}^n$ such that $\{q \in \arg\min g \mid q \leq p\} \neq \emptyset$.* □

Theorem 4.8 (cf. [40])**.**
(i) *If vector p is updated in an iteration of the algorithm* GreedyDown*, then the value $\check{\mu}(p)$ decreases by one.*
(ii) *The number of update of vector p in* GreedyDown *is equal to $\check{\mu}(p_0)$ and at most Φ_g.*
(iii) *The minimizer $p_* \in \arg\min g$ found by* GreedyDown *satisfies $\|p_* - p_0\|_\infty = \check{\mu}(p_0)$.* □

Corollary 4.9 ([40, 42])**.** *The algorithm* GreedyDownMaximal *outputs the unique maximal minimizer $\overline{p}_*$ after vector p is updated $\|\overline{p}_* - p_0\|_\infty$ times.* □

Remark 4.10. We consider the special case where $g : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ is an L-convex function. If g has a minimizer, then for every vector p_0, there exists a minimizer p_* satisfying $p_* \geq p_0$ and $\min_{i \in N}\{p_*(i) - p_0(i)\} = 0$ (see the discussion just before Theorem 2.8). Hence, $\hat{\mu}(p_0) \leq \Psi_g$ holds with

$$\Psi_g = \max\{\|p - q\|_\infty \mid p, q \in \operatorname{dom} g,\ \min_{i \in N}\{p(i) - q(i)\} = 0\}.$$

If the value Ψ_g is finite, then the increasing-type steepest descent algorithms GreedyUp and GreedyUpMinimal terminate in at most Ψ_g iterations. Similarly, the decreasing-type steepest descent algorithms GreedyDown and GreedyDownMaximal terminate in at most Ψ_g iterations. □

By generalizing the analysis used in Theorem 4.4, we can obtain the exact bound for the number of iterations required by the algorithms Greedy and GreedyMinimal.

Theorem 4.11 ([40])**.**
(i) *If vector p is updated in an iteration of the algorithm* Greedy*, then the value $\mu(p)$ decreases by one.*
(ii) *The number of update of vector p in* Greedy *is equal to $\mu(p_0)$ and at most $2\Phi_g$.*
(iii) *The minimizer $p_* \in \arg\min g$ found by* Greedy *satisfies $\|p_* - p_0\|_\infty^+ + \|p_* - p_0\|_\infty^- = \mu(p_0)$.* □

It follows from this theorem that the trajectory of vector p generated by the algorithm Greedy is a shortest path (with respect to a certain distance) from the initial solution p_0 to a nearest minimizer p_*.

Corollary 4.12 ([40])**.** *The algorithm* GreedyMinimal *outputs the unique minimal minimizer $\underline{p}_*$ after vector p is updated $\|\underline{p}_* - p_0\|_\infty^+ + \|\underline{p}_* - p_0\|_\infty^-$ times.* □

Example 4.13. We reconsider Example 3.2. When we apply the algorithm Greedy with the initial solution $p_0 = (1, 4)$, the output of the algorithm is either of $p_* = (3, 4)$ and $p_* = (2, 3)$, both of which are nearest minimizers to p_0 (with respect to the distance $\|p_* - p\|_\infty^+ + \|p_* - p\|_\infty^-$), and the number of iterations is equal to $\mu(p_0) = 2$, the distance to a nearest minimizer. Three possible trajectories of p in the algorithm Greedy are drawn

by dotted arrows in Figure 2, each of which is a shortest path from the initial solution $p_0 = (1, 4)$ to a nearest minimizer. □

Remark 4.14. As upper bounds for the number of iterations required by the algorithm Greedy, $2n\Phi_g$ is shown by Murota [30] and $2\Phi_g$ by Kolmogorov–Shioura [25]. The result presented in this section (Theorem 4.11) is a refinement of these previous results. □

Remark 4.15. Recently, Hirai [15, 16] introduced the concept of L-convex function on oriented modular graphs, as a generalization of $\mathrm{L}^\natural$-convex function on integer lattice points. In this concept, a function is defined on the set of vertices, and the neighborhood of each vertex is defined by using oriented edges. In this framework, an $\mathrm{L}^\natural$-convex function on integer lattice points can be regarded as an L-convex function on a grid graph with appropriate edge orientation. Hirai [15, 16] proposed a steepest descent algorithm for minimization of an L-convex function on an oriented modular graph, and showed that the number of iterations is equal to a certain distance between the initial solution and the minimizer output by the algorithm. This result generalizes Theorems 4.4 and 4.11 presented in this section. □

4.2. Analysis by the slope of steepest descent direction

We consider a variant of the increasing-type steepest descent algorithm GreedyUpMinimal using long step length, which we denote GreedyUpMinimal-LS. In this section, we analyze the number of iterations required by GreedyUpMinimal-LS by using slopes of steepest descent directions. For this, we first show a property concerning the slope of a steepest descent direction in each iteration of GreedyUpMinimal (i.e., the algorithm using the unit step length) and clarify the relationship between GreedyUpMinimal and GreedyUpMinimal-LS.

We show that in GreedyUpMinimal, the absolute value $|g'(p; d)|$ of the slope of a steepest descent direction d is non-increasing; moreover, if the unique minimal steepest descent direction is used, then the absolute value $|g'(p; d)|$ of the slope decreases strictly or the set $\mathrm{supp}^+(d)$ of non-zero components in a steepest descent direction d monotonically increases (in the sense of set inclusion). Note that $g'(p; d) \leq 0$ holds for every $p \in \mathrm{dom}\, g$ and every steepest descent direction d at p.

Proposition 4.16 (cf. [13])**.** *Let $p \in \mathrm{dom}\, g$, and $d \in \{0, +1\}^n$ be a steepest descent direction at p, and $\widetilde{d} \in \{0, +1\}^n$ be a steepest descent direction at $p + d$.*

(i) $0 \geq g'(p + d; \widetilde{d}) \geq g'(p; d)$ *holds.*

(ii) *Suppose that d and $\widetilde{d}$ are minimal steepest descent directions at p and $p+d$, respectively. Then, at least one of the following two conditions holds:*

(a) $g'(p+d; \widetilde{d}) > g'(p; d)$, (b) $g'(p+d; \widetilde{d}) = g'(p; d)$ *and* $\mathrm{supp}^+(\widetilde{d}) \supseteq \mathrm{supp}^+(d)$.

□

For $p \in \mathrm{dom}\, g$, let $d \in \{0, +1\}^n$ be a steepest descent direction at p. Proposition 4.16 (i) implies that d is also a steepest descent direction at $p + \lambda d$ for every $\lambda \in \mathbb{Z}_+$ with $0 \leq \lambda < \bar{c}(p; d)$. Hence, the algorithm GreedyUpMinimal-LS using long step lengths can be regarded as a variant of the algorithm GreedyUpMinimal, where the same steepest descent direction as in the previous iteration is used whenever possible. From this observation follows that the trajectory of vector p of the algorithm GreedyUpMinimal-LS is the same as the trajectory of vector p of GreedyUpMinimal, and the output of GreedyUpMinimal-LS is a minimizer of function g.

An upper bound for the number of iterations of GreedyUpMinimal-LS can be obtained by using Proposition 4.16 (ii). When vector p is updated to $\tilde{p} = p + \lambda d$ with $\lambda = \bar{c}(p; d)$, the slopes $g'(p; d)$ and $g'(\tilde{p}; d)$ satisfy $0 \geq g'(\tilde{p}; d) > g'(p; d)$ by the definition of λ. Therefore, Proposition 4.16 (ii) implies that for the unique minimal steepest descent

direction $\widetilde{d} \in \{0,+1\}^n$ at $\tilde{p}$, at least one of $g'(\tilde{p};\widetilde{d}) > g'(p;d)$ and $\mathrm{supp}^+(\widetilde{d}) \supsetneq \mathrm{supp}^+(d)$ holds. Since the slope of a steepest descent direction is equal to zero at the termination of the algorithm, the next theorem follows.

Theorem 4.17. *For an integer-valued $L^\natural$-convex function g, the number of iterations of the algorithm* GreedyUpMinimal-LS *is at most* $n \cdot \max\{-g'(p_0;d) \mid d \in \{0,+1\}^n\}$. □

Remark 4.18. Consider the example of the minimum cost flow problem in Section 3.3.1. Function g_{H} is integer-valued if cost $c(u,v)$ and capacity $\gamma(u,v)$ are integer-valued for every edge $(u,v) \in E$. Hence, from Theorem 4.17 the following upper bound for the number of iterations of Hassin's algorithm can be obtained:

$$n \cdot \max\{-g'_{\mathrm{H}}(p_0;d) \mid d \in \{0,+1\}^n\}.$$

This bound coincides with the one shown in Hassin [20, Corollary 3]. Since the slope $g'_{\mathrm{H}}(p;d)$ at a vector $p \in \mathrm{dom}\, g_{\mathrm{H}}$ can be represented as

$$g'_{\mathrm{H}}(p;d) = \sum_{(u,v)\in E'} c(u,v) - \sum_{(u,v)\in E''} c(u,v)$$

for some subsets $E', E'' \subseteq E$ of edges, we have

$$g'_{\mathrm{H}}(p_0;d) \geq -\sum_{(u,v)\in E} |c(u,v)|.$$

From this we obtain an upper bound $n \cdot \sum_{(u,v)\in E} |c(u,v)|$ for the number of iterations of Hassin's algorithm, which is independent of the choice of the initial solution p_0. □

5. Minimization of L-convex Function in Continuous Variables

The concepts of L-convexity and L$^\natural$-convexity naturally extend to functions in continuous variables. Moreover, the minimization algorithms shown in Section 3 as well as their properties can also be naturally extend to L-convex and L$^\natural$-convex functions in continuous variables.

5.1. Definition of L-convex function in continuous variables

A convex function $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ in continuous variables is called an *L-convex function* [35, 36] if it satisfies the following two properties, where $\mathrm{dom}_{\mathbb{R}}\, g = \{p \in \mathbb{R}^n \mid g(p) < +\infty\}$:

[Submodularity] $g(p) + g(q) \geq g(p \vee q) + g(p \wedge q) \qquad (\forall p, q \in \mathrm{dom}_{\mathbb{R}}\, g),$

[Linearity in the Direction of 1] $\exists r \in \mathbb{R} : g(p + \alpha \mathbf{1}) = g(p) + \alpha r \quad (\forall p \in \mathbb{R}^n,\ \forall \alpha \in \mathbb{R}).$

For an L-convex function $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, the function $\check{g} : \mathbb{R}^{n-1} \to \mathbb{R} \cup \{+\infty\}$ with $n-1$ variables given as

$$\check{g}(q(1), \ldots, q(n-1)) = g(q(1), \ldots, q(n-1), 0)$$

is called an *$L^\natural$-convex function* [35, 36]. In other words, a function $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is said to be L$^\natural$-convex if the function $\tilde{g} : \mathbb{R}^{n+1} \to \mathbb{R} \cup \{+\infty\}$ with $n+1$ variables given by

$$\tilde{g}(q(1), \ldots, q(n), q(n+1)) = g(q(1) - q(n+1), \ldots, q(n) - q(n+1)) \quad (q \in \mathbb{R}^{n+1})$$

is an L-convex function. As in the case of functions in discrete variables, the function $\tilde{g}$ defined above satisfies the linearity in the direction of $\mathbf{1}$. Hence, L-convexity of function $\tilde{g}$

is equivalent to submodularity of function $\tilde{g}$. This means that $\mathrm{L}^\natural$-convexity of function g is equivalent to submodularity of function $\tilde{g}$.

As in the case of functions in discrete variables, $\mathrm{L}^\natural$-convexity of function $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ can be characterized by the translation submodularity:

[**Translation Submodularity**]

$$g(p) + g(q) \geq g((p - \alpha\mathbf{1}) \vee q) + g(p \wedge (q + \alpha\mathbf{1})) \qquad (\forall p, q \in \mathrm{dom}_{\mathbb{R}}\, g,\ \forall \alpha \in \mathbb{R}_+).$$

Remark 5.1. It should be noted that $\mathrm{L}^\natural$-convexity does not follow from the mid-point convexity

$$g(p) + g(q) \geq 2g\left(\frac{p+q}{2}\right) \qquad (\forall p, q \in \mathrm{dom}_{\mathbb{R}}\, g).$$

Under the continuity of a function, the mid-point convexity is equivalent to convexity in the ordinary sense, which is more general than $\mathrm{L}^\natural$-convexity. □

It is clear from their definitions that L-convex and $\mathrm{L}^\natural$-convex functions in continuous variables have close relationship with L-convex and $\mathrm{L}^\natural$-convex functions in discrete variables. If a function $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is an L-convex (resp., $\mathrm{L}^\natural$-convex) function in continuous variables, then its restriction $g_{\mathbb{Z}} : \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ on integer lattice points is an L-convex (resp., $\mathrm{L}^\natural$-convex) function in discrete variables, On the other hand, in the examples of L-convex (resp., $\mathrm{L}^\natural$-convex) functions in discrete variables shown in Section 2.2, if we replace the domain $\mathbb{Z}^n$ with $\mathbb{R}^n$, then we can obtain L-convex (resp., $\mathrm{L}^\natural$-convex) functions in continuous variables. In this way, when we are given L-convex and $\mathrm{L}^\natural$-convex functions in discrete variables, it is often possible to obtain L-convex and $\mathrm{L}^\natural$-convex functions in continuous variables by replacing the domain $\mathbb{Z}^n$ with $\mathbb{R}^n$. Furthermore, it is known that every L-convex (resp., $\mathrm{L}^\natural$-convex) function in discrete variables can be extended to an L-convex (resp., $\mathrm{L}^\natural$-convex) functions in continuous variables (see, e.g., [28, 29, 32, 38]).

5.2. Minimization algorithms and their properties

Minimization algorithms for $\mathrm{L}^\natural$-convex functions in discrete variables can be naturally extend to polyhedral $\mathrm{L}^\natural$-convex functions. A convex function $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is said to be *polyhedral* if its epigraph $\{(p, \beta) \mid p \in \mathbb{R}^n,\ \beta \in \mathbb{R},\ g(p) \leq \beta\}$ is a polyhedron. In this section, we assume that $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is a polyhedral $\mathrm{L}^\natural$-convex function.

For $p \in \mathrm{dom}_{\mathbb{R}}\, g$ and $d \in \mathbb{R}^n$, the *directional derivative* $g'(p; d)$ of function g at p in the direction d is given as

$$g'(p; d) = \lim_{\lambda \downarrow 0} \frac{g(p + \lambda d) - g(p)}{\lambda}.$$

Since g is a polyhedral convex function, the limit in the right-hand side exists if we allow to have $g'(p; d) = +\infty$.

By its definition, an $\mathrm{L}^\natural$-convex function g in continuous variables is a convex function in the ordinary sense. Hence, a vector $p \in \mathrm{dom}_{\mathbb{R}}\, g$ is a minimizer if and only if $g'(p; d) \geq 0$ holds for every direction $d \in \mathbb{R}^n$. In fact, by $\mathrm{L}^\natural$-convexity we may restrict our attention only to a finite number of directions to check the minimality of a vector.

Theorem 5.2 ([28, 29, 35])**.** *Let $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ be a polyhedral $\mathrm{L}^\natural$-convex function. Then, a vector $p \in \mathrm{dom}_{\mathbb{R}}\, g$ is a minimizer of g if and only if $g'(p; d) \geq 0$ holds for every $d \in \{0, +1\}^n \cup \{0, -1\}^n$.* □

This is a natural extension of Theorem 2.5 for $\mathrm{L}^\natural$-convex functions in discrete variables. By this theorem, the steepest descent algorithm Greedy-LS in Section 3 can be extended to polyhedral $\mathrm{L}^\natural$-convex functions. Note that it is natural to use step length in each iteration

when we apply a steepest descent algorithm to polyhedral $\mathrm{L}^\natural$-convex functions. For $p \in \mathrm{dom}_{\mathbb{R}}\, g$ and $d \in \mathbb{R}^n$ with $g'(p;d) < +\infty$, we define the value $\bar{c}(p;d) \in \mathbb{R} \cup \{+\infty\}$ by

$$\bar{c}(p;d) = \sup\{\lambda \in \mathbb{R}_+ \mid g(p+\lambda d) - g(p) = \lambda\, g'(p;d)\}. \tag{5.1}$$

Since g is a polyhedral convex function, the value $\bar{c}(p;d)$ is always positive, and $g(p+\lambda d) - g(p) = \lambda\, g'(p;d)$ holds with $\lambda = \bar{c}(p;d)$.

Algorithm 8 (Greedy-LS[$\mathbb{R}$])**.**

Step 0: Select an initial solution $p_0 \in \mathrm{dom}_{\mathbb{R}}\, g$ arbitrarily and set $p := p_0$.

Step 1: If $g'(p;d) \geq 0$ holds for every $d \in \{0,+1\}^n \cup \{0,-1\}^n$, then output p and stop.

Step 2: Find $d = d_* \in \{0,+1\}^n \cup \{0,-1\}^n$ minimizing $g'(p;d)$. Set $p := p + \lambda d_*$ with $\lambda := \bar{c}(p;d_*)$ and go to Step 1. □

Theorem 4.11 for the case of discrete variables can be extended to this algorithm. In a similar way as in Section 4.1, we define a function $\mu : \mathbb{R}^n \to \mathbb{R}_+$ by

$$\mu(p) = \min\{\|p_* - p\|_\infty^+ + \|p_* - p\|_\infty^- \mid p_* \in \arg\min_{\mathbb{R}} g\},$$

where

$$\arg\min_{\mathbb{R}} g = \{p \in \mathbb{R}^n \mid g(p) \leq g(q)\ (\forall q \in \mathbb{R}^n)\}.$$

Theorem 5.3 ([13])**.** *In each iteration of the algorithm* Greedy-LS[$\mathbb{R}$]*, the value $\mu(p)$ decreases by the step length $\bar{c}(p;d)$. In addition, the algorithm outputs a minimizer p_* of g satisfying $\|p_* - p_0\|_\infty^+ + \|p_* - p_0\|_\infty^- = \mu(p_0)$ when it terminates.* □

This theorem implies that when the algorithm Greedy-LS[$\mathbb{R}$] is applied to a polyhedral $\mathrm{L}^\natural$-convex function, its output is a minimizer p_* nearest to the initial solution, and the trajectory of vector p is a shortest path from the initial solution to p_*. It should be noted that this theorem does not guarantee the termination of the algorithm.

Theorem 2.6 (ii) for the case of discrete variables can also be extended to polyhedral $\mathrm{L}^\natural$-convex functions as follows.

Theorem 5.4 (cf. [40])**.** *Let $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ be a polyhedral $\mathrm{L}^\natural$-convex function such that a minimizer exists, and $p \in \mathrm{dom}_{\mathbb{R}}\, g$ be a vector that is a lower bound of some minimizer of g. Then, p is a minimizer of g if and only if $g'(p;d) \geq 0$ for every $d \in \{0,+1\}^n$.* □

By this theorem the increasing-type steepest descent algorithm GreedyUp-LS can be also extended to polyhedral $\mathrm{L}^\natural$-convex functions.

Algorithm 9 (GreedyUp-LS[$\mathbb{R}$])**.**

Step 0: Select an initial solution $p_0 \in \mathrm{dom}_{\mathbb{R}}\, g$ that is a lower bound of some minimizer of g, and set $p := p_0$.

Step 1: If $g'(p;d) \geq 0$ holds for every $d \in \{0,+1\}^n$, then output p and stop.

Step 2: Find $d = d_* \in \{0,+1\}^n$ minimizing $g'(p;d)$. Set $p := p + \lambda d_*$ with $\lambda := \bar{c}(p;d_*)$ and go to Step 1. □

Theorem 4.4 for the case of discrete variables can be extended to this algorithm. As in Section 4.1, for a vector $p \in \mathrm{dom}_{\mathbb{R}}\, g$ that is a lower bound of some minimizer of g, we define the value $\hat{\mu}(p) \in \mathbb{R}_+$ by

$$\hat{\mu}(p) = \begin{cases} \min\{\|p_* - p\|_\infty \mid p_* \in \arg\min_{\mathbb{R}} g,\ p_* \geq p\} & (\text{if } \{p_* \in \arg\min_{\mathbb{R}} g \mid p_* \geq p\} \neq \emptyset), \\ +\infty & (\text{otherwise}). \end{cases}$$

Theorem 5.5 (cf. [13])**.**

(i) *If vector p is updated in an iteration of the algorithm* GreedyUp-LS[$\mathbb{R}$]*, then the value $\hat{\mu}(p)$ decreases by the step length $\bar{c}(p;d)$.*

(iii) *The minimizer $p_* \in \arg\min g$ found by* GreedyUp-LS[$\mathbb{R}$] *satisfies $\|p_* - p_0\|_\infty = \hat{\mu}(p_0)$.* □

Theorems 5.2 and 5.4 imply that the algorithms GREEDY-LS[$\mathbb{R}$] and GREEDYUP-LS[$\mathbb{R}$] output minimizers when they terminate. These algorithms, however, have no guarantee to terminate in a finite number of iterations; furthermore, there is no guarantee that the sequence of vector p generated by each of the algorithm converges to some minimizer. Indeed, it is shown in [13] that for some concrete example of the function $-g_{\mathrm{H}}$ in Section 3.3.1 neither GREEDYUP-LS[$\mathbb{R}$] terminates nor the sequence of vector p in the algorithm converges to a minimizer; recall that $-g_{\mathrm{H}}$ is a polyhedral L-convex function.

On the other hand, we can show that the algorithm terminates in a finite number of iterations if we use a special choice of a steepest descent direction in each iteration. For example, use of minimal steepest descent directions in GREEDYUP-LS[$\mathbb{R}$] guarantees the finite termination.

Algorithm 10 (GREEDYUPMINIMAL-LS[$\mathbb{R}$])**.**

Step 0: Select an initial solution $p_0 \in \operatorname{dom} g$ that is a lower bound of the minimal minimizer of g, and set $p := p_0$.

Step 1: If $g'(p; d) \geq 0$ holds for every $d \in \{0, +1\}^n$, then output p and stop.

Step 2: Find a (unique) minimal minimizer $d = d_* \in \{0, +1\}^n$ of $g'(p; d)$. Set $p := p + \lambda d_*$ with $\lambda := \bar{c}(p; d_*)$ and go to Step 1. □

For this algorithm a property similar to Proposition 4.16 (ii) for L$^\natural$-convex function in discrete variables holds.

Proposition 5.6. *Let $p \in \operatorname{dom}_{\mathbb{R}} g$, and $d \in \{0, +1\}^n$ be the unique minimal steepest descent direction at p. Also, let $\tilde{p} = p + \bar{c}(p; d)d$, and $\tilde{d}$ be the unique minimal steepest descent direction at $\tilde{p}$. Then, at least one of the following two conditions holds:*

(a) $g'(\tilde{p}; \tilde{d}) > g'(p; d)$, (b) $g'(\tilde{p}; \tilde{d}) = g'(p; d)$ *and* $\operatorname{supp}^+(\tilde{d}) \supsetneq \operatorname{supp}^+(d)$.

□

For a polyhedral convex function g, the set of possible values of the directional derivatives $g'(p; d)$ for $p \in \operatorname{dom}_{\mathbb{R}} g$ and $d \in \{0, +1\}^n$ is finite. This fact, together with the proposition above, implies the finite termination of GREEDYUPMINIMAL-LS[$\mathbb{R}$].

Theorem 5.7. *For a polyhedral L$^\natural$-convex function $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, the algorithm* GREEDYUPMINIMAL-LS[$\mathbb{R}$] *terminates in a finite number of iterations and outputs a minimizer of g. Moreover, the algorithm outputs the unique minimal minimizer of g if the initial solution p_0 is a lower bound of the minimal minimizer.* □

6. Application to Auction Theory

In this section, we show that results on L-convex function minimization algorithms explained so far can be applied to iterative auctions in auction theory. Iterative auctions are used to compute equilibrium prices of goods. For a certain general model discussed in Section 6.1, iterative auctions can be seen as minimization algorithms applied to a special L$^\natural$-convex function, as shown in Section 6.2.1.

6.1. Model of auction and fundamental concepts

Our auction model and fundamental concepts used in the following discussion are explained below. Let us consider an auction with n types of indivisible goods. We denote the set of goods by $N = \{1, 2, \ldots, n\}$. We suppose that m bidders attend the auction, and denote the set of bidders by $M = \{1, 2, \ldots, m\}$. Each bidder $j \in M$ has his/her own valuation function $f_j : 2^N \to \mathbb{R}$, where the value $f_j(X)$ represents the degree of satisfactory for the set of goods $X \subseteq N$. Given a price vector $p \in \mathbb{R}^n_+$, each bidder $j \in M$ wants to buy a set of goods X_j that maximizers the utility $f_j(X_j) - p(X_j)$. For each bidder $j \in M$, we denote by $V_j(p) \in \mathbb{R}$ the maximum utility of the bidder j and by $D_j(p) \subseteq 2^N$ a family of sets of

goods maximizing utility. That is,

$$V_j(p) = \max\{f_j(X) - \sum_{i \in X} p(i) \mid X \subseteq N\} \qquad (p \in \mathbb{R}^n_+), \tag{6.1}$$

$$D_j(p) = \arg\max\{f_j(X) - \sum_{i \in X} p(i) \mid X \subseteq N\} \qquad (p \in \mathbb{R}^n_+). \tag{6.2}$$

Function V_j is called the *indirect utility function* of bidder j, and the set family $D_j(p)$ for $p \in \mathbb{R}^n_+$ is called the *demand set* (or *demand correspondence*) of bidder j.

The goal of an auction is to find a partition $\{X^*_1, X^*_2, \ldots, X^*_m\}$ (i.e., an allocation of goods to bidders) of goods in N and a price vector p_* satisfying the condition $X^*_j \in D_j(p_*)$ $(j \in M)$. That is, we want to find an allocation of goods and a price vector such that each bidder is fully satisfied and all goods are sold out. Such a pair of a partition of goods $\{X^*_1, X^*_2, \ldots, X^*_m\}$ and a price vector p_* is called a *Walrasian equilibrium* (or *competitive equilibrium*); p_* is called a *Walrasian equilibrium price vector*.

Although Walrasian equilibrium may not exist in general, it exists if each valuation function f_j satisfies a certain natural condition called the gross substitutes condition [18, 23]. The *gross substitutes condition* for a valuation function f_j is described as follows by using a demand set $D_j(p)$:

(GS) $\forall p, q \in \mathbb{R}^n_+$ with $p \leq q$, $\forall X \in D_j(p)$, $\exists Y \in D_j(q)$ such that $Y \supseteq \{i \in X \mid p(i) = q(i)\}$.

The gross substitutes condition means that when prices of some goods increase, the only goods that may drop from the optimal set of goods are those with increased prices. The gross substitutes condition is first introduced by Kelso–Crawford [23] in the setting of a fairly general two-sided job matching model. Since then, this condition has been widely used in various models such as matching, housing, and labor markets (see, e.g., [2, 4, 7, 18, 19]).

An *iterative auction* is an algorithm (more precisely, a protocol) to compute an equilibrium by repeatedly updating a price vector [4, 7]. An iterative auction with prices monotonically increasing in each iteration is called an ascending auction. An ascending auction for multiple goods is a generalization of the well-known English auction for a single good (see, e.g., [4, 7]). In addition, an ascending auction is natural from the economic viewpoint, and easy to understand and to be implemented.

For an auction model with gross substitutes valuation functions, an ascending auction is proposed by Gul–Stacchetti [19]. Then, Ausubel [2] featured the concept of Lyapunov function, which is a function in a price vector given by

$$L(p) = \sum_{j=1}^{m} V_j(p) + \sum_{i=1}^{n} p(i) \qquad (p \in \mathbb{R}^n_+). \tag{6.3}$$

Ausubel [2] showed that the Lyapunov function enjoys various nice properties.

Theorem 6.1 ([2]). *Suppose that each valuation function f_j satisfies the gross substitutes condition.*

(i) *A vector $p \in \mathbb{R}^n_+$ is a minimizer of the Lyapunov function if and only if it is an equilibrium price vector vector.*

(ii) *Suppose that each f_j is integer-valued. Then, the Lyapunov function has an integral minimizer. In particular, a minimal and maximal minimizer of the Lyapunov function are integral vectors.* □

Based on these facts, Ausubel [2] proposed an ascending auction and a descending auction that find an equilibrium price vector through the minimization of the Lyapunov function. Note that the ascending auction of Ausubel [2] can be seen as a reformulation of the one by Gul–Stacchetti [19].

6.2. Analysis of iterative auctions

6.2.1. Connection between auction theory and discrete convex analysis

We discuss the connection between auction theory and discrete convex analysis in this section. Discrete convex analysis is a theory on discrete convexity, where another class of discrete convex functions called *M$^\natural$-convex functions* [34] in addition to L$^\natural$-convex functions plays a primary role. Similar to L$^\natural$-convexity, M$^\natural$-convexity is defined for functions on integer lattice points. The definition of M$^\natural$-convexity, when specialized to a set function $f : 2^N \to \mathbb{R}$, is described as follows:

> a function $f : 2^N \to \mathbb{R}$ is called an *M$^\natural$-convex function* if for every $X, Y \subseteq N$ and every $i \in X \setminus Y$, at least one of (a) and (b) holds:
> (a) $f(X) + f(Y) \geq f(X - u) + f(Y + u)$,
> (b) $\exists v \in Y \setminus X$ such that $f(X) + f(Y) \geq f(X - u + v) + f(Y + u - v)$,

where $X - u + v = (X \setminus \{u\}) \cup \{v\}$. A function $f : 2^N \to \mathbb{R}$ is called an *M$^\natural$-concave function* if $-f$ is M$^\natural$-convex.

The connection between auction theory and discrete convex analysis is initiated by the following theorem due to Fujishige–Yang [14]:

Theorem 6.2 ([14]). *A function$f : 2^N \to \mathbb{R}$ satisfies the gross substitutes condition if and only if it is an M$^\natural$-concave function.* □

Based on this theorem, known results in discrete convex analysis are used in various fields in mathematical economics such as auction theory and game theory, while mathematical economics provides discrete convex analysis with interesting application.

The following conjugacy relation holds between M$^\natural$-concavity and L$^\natural$-convexity. We say that a polyhedral L$^\natural$-convex function g is *domain-integral* if the set $\arg\min\{g(p) - p^\top x \mid p \in \mathbb{R}^n\}$ is an empty set or an integral polyhedron for every $x \in \mathbb{R}^n$.

Theorem 6.3 ([27, 29, 35]). *For a set function$f : 2^N \to \mathbb{R}$, define a function $g : \mathbb{R}^n \to \mathbb{R}$ by*

$$g(p) = \max\{f(X) - \sum_{i \in X} p(i) \mid X \subseteq N\}.$$

(i) *f is an M$^\natural$-concave function if and only if g is a polyhedral L$^\natural$-convex function.*
(ii) *Suppose that f is an integer-valued M$^\natural$-concave function. Then, g is a domain-integral polyhedral L$^\natural$-convex function.* □

L$^\natural$-convexity of the indirect utility function and the Lyapunov function can be obtained from Theorems 6.2 and 6.3.

Theorem 6.4 ([41, 42]). *Suppose that the valuation function $f_j : 2^N \to \mathbb{R}$ of each bidder $j \in M$ satisfies the gross substitutes condition. Then, for each $j \in M$, the indirect utility function $V_j : \mathbb{R}^n_+ \to \mathbb{R}$ and the Lyapunov function $L : \mathbb{R}^n_+ \to \mathbb{R}$ are polyhedral L$^\natural$-convex functions. Moreover, if each f_j is an integer-valued function, then the Lyapunov function L is a domain-integral polyhedral L$^\natural$-convex function.* □

Thus, an iterative auction using the Lyapunov function can be seen as a minimization algorithm for a special L$^\natural$-convex function.

6.2.2. Ascending auction and descending auction

We analyze the ascending and descending auctions of Ausubel [2] from the viewpoint of discrete convex analysis. The ascending auction, which finds the unique minimal (integral) equilibrium price vector $\underline{p}^*$, can be described as follows:

Algorithm 11 (AscendingAuction)**.**

Step 0: Select an initial price vector $p_0 \in \mathbb{Z}^n_+$ with $p_0 \leq \underline{p}^*$, and set $p := p_0$.

Step 1: If $L'(p; d) \geq 0$ holds for every $d \in \{0, +1\}^n$, then output p and stop.

Step 2: Let $d = d_* \in \{0, +1\}^n$ be a (unique) minimal vector among all minimizers of $L'(p; d)$. Set $p := p + d_*$ and go to Step 1. □

The algorithm AscendingAuction can be interpreted in auction terms as follows:

Step 0: The auctioneer sets the initial price vector $p := p^\circ$, where $p^\circ \in \mathbb{Z}^n$ satisfies $p^\circ \leq \underline{p}^*$.

Step 1: The auctioneer asks the bidders to report their demand sets $D_j(p)$ $(j \in M)$, and using the information of demand sets, checks if $L'(p; d) \geq 0$ holds for every $d \in \{0, +1\}^n$. If it holds, the auctioneer reports p as the final price vector and stop.

Step 2: The auctioneer finds a (unique) minimal minimizer $d = d_* \in \{0, +1\}^n$ of $L(p + d)$, sets $p := p + d_*$, and returns to Step 1. □

Remark 6.5. As mentioned above, if the number of goods n is equal to one, then this ascending auction coincides with the classical English auction. That is, the ascending auction of Ausubel [2] is a natural generalization of English auction. □

Remark 6.6. We discuss an implementation issue of iterative auctions. In iterative auctions, it is often the case that values of valuation functions f_j are not explicitly given since they are bidders' private information. In such a case, the value of the Lyapunov function cannot be computed. Instead, in the ascending auction of Ausubel [2] we can obtain the information of demand sets $D_j(p)$ for a given price vector $p \in \mathbb{Z}^n_+$, by which we can compute the slope $L'(p; d)$ of the Lyapunov function for any given direction $d \in \{0, +1\}^n \cup \{0, -1\}^n$ (see [2, Appendix B] or the technical report version of [41]). Therefore, in the ascending auction it is possible to check the local optimality of the Lyapunov function in Step 1 and to compute a steepest descent direction in Step 2. □

Ausubel [2] showed that the algorithm AscendingAuction terminates in a finite number of iterations, and at the termination the algorithm outputs the unique minimal equilibrium price vector $\underline{p}_*$. By the results in Section 4.1, we can obtain the exact bound for the number of iterations.

By Theorem 6.4, the Lyapunov function is a polyhedral $\mathrm{L}^\natural$-convex function. Since the unique minimal equilibrium price vector is an integral vector, we can restrict the Lyapunov function on the integer lattice points; the resulting function can be seen as an $\mathrm{L}^\natural$-convex function in discrete variables. Therefore, the algorithm AscendingAuction can be seen a minimization algorithm for a special $\mathrm{L}^\natural$-convex function (i.e., Lyapunov function) and coincides with the algorithm GreedyUpMinimal applied to the Lyapunov function. This observation makes it possible to apply Corollary 4.5 to AscendingAuction to obtain the exact bound for the number of iterations.

Theorem 6.7 ([41, 42])**.** *The algorithm* AscendingAuction *updates a price vector exactly* $\|\underline{p}^* - p^\circ\|_\infty$ *times and then terminates by outputting the unique minimal equilibrium price vector* $\underline{p}_*$. □

Any ascending auction algorithm requires at least $\|\underline{p}^* - p^\circ\|_\infty$ iterations if the algorithm increases the price of each good by at most one. This means that Ausubel's ascending auction finds the minimal equilibrium price vector using the smallest number of iterations, i.e., it is the "fastest" auction algorithm among all ascending auction algorithms.

We then explain the descending auction of Ausubel [2], which is quite similar to the ascending auction. The algorithm iteratively decreases prices and finds the unique maximal equilibrium price vector $\overline{p}^*$.

Algorithm 12 (DescendingAuction)**.**

Step 0: Select an initial price vector $p_0 \in \mathbb{Z}^n_+$ with $p_0 \leq \overline{p}^*$, and set $p := p_0$.

Step 1: If $L'(p; d) \geq 0$ holds for every $d \in \{0, -1\}^n$, then output p and stop.

Step 2: Let $d = d_* \in \{0, -1\}^n$ be a (unique) maximal vector among all minimizers of $L'(p; d)$. Set $p := p + d_*$ and go to Step 1. □

Remark 6.8. If the number of goods n is equal to one, then this descending auction coincides with the classical Dutch auction. That is, the descending auction of Ausubel [2] is a natural generalization of Dutch auction. □

The algorithm DescendingAuction coincides with the algorithm GreedyDown-Maximal applied to the Lyapunov function. Hence, we can analyze DescendingAuction in a similar way as AscendingAuction to obtain the following theorem.

Theorem 6.9 ([41, 42])**.** *The algorithm* DescendingAuction *updates a price vector exactly* $\|p^\circ - \overline{p}^*\|_\infty$ *times and then terminates by outputting the unique maximal equilibrium price vector* $\overline{p}_*$. □

Remark 6.10. A variant of auction algorithm in Gul–Stacchetti [19] uses line search to compute an equilibrium price vector (see [19, p. 77]). It can be shown that the behavior of the auction algorithm coincides with that of the increasing-type steepest descent algorithm GreedyUpMinimal-LS using long step length applied to the Lyapunov function. Hence, we can apply Theorem 4.17 to obtain the number of iterations of the auction algorithm of Gul–Stacchetti [19].

Recall that the Lyapunov function is an integer-valued function under the assumption that each f_j are integer-valued functions. By the definition of the Lyapunov function (6.3), the slope of the Lyapunov function can be represented as

$$L'(p; d) = -\sum_{j=1}^{m} d(X_j) + d(N) \qquad (d \in \{0, +1\}^n)$$

with some subsets $X_1, X_2, \ldots, X_m$ of N. Therefore we have $L'(p; d) \geq -(m-1)n$. Hence, the auction algorithm of Gul–Stacchetti [19] terminates at most $(m-1)n^2$ iterations; this bound is also shown in Saito [44]. □

6.2.3. Greedy auction and two-phase auction

An important advantage of ascending and descending auctions is that the change of a price vector is monotone. On the other hand, the initial price vector of ascending (resp., descending) auctions should be a lower (resp., upper) bound of an equilibrium price vector, which should be sufficiently small (resp., large) since an equilibrium price vector is not known in advance. This is a disadvantage of ascending and descending auctions since it makes the number of iterations of the algorithm fairly large.

To eliminate the disadvantage, Murota–Shioura–Yang [41, 42] proposed two kinds of auction algorithms which are based on the understanding of L-convex function minimization. We explain below the details of the algorithms.

The first algorithm, which we call the greedy auction, is an iterative auction obtained by applying GreedyMinimal in Section 3.4 to the Lyapunov function.

Algorithm 13 (GreedyAuction)**.**

Step 0: Select an initial price vector $p_0 \in \mathbb{Z}^n_+$ arbitrarily, and set $p := p_0$.

Step 1: Let $d = d_* \in \{0, +1\}^n \cup \{0, -1\}^n$ be a minimal minimizer of $L'(p; d)$.
Step 2: If $d_* = \mathbf{0}$ then output p and stop. Otherwise, set $p := p + d_*$ and go to Step 1. □

Remark 6.11. The greedy auction is a slight modification of the one in Murota–Shioura–Yang [41] so that a minimal equilibrium price vector can be found. □

The greedy auction has an advantage that any price vector can be used initially. Therefore, if a good estimate of an equilibrium price vector is available from the output of auctions in the past, we can use it as the initial price vector, resulting in the reduction of the number of iterations. Corollary 4.12 for the algorithm GreedyMinimal shows that the number of iterations of GreedyAuction is equal to a distance between the initial price vector p_0 and the minimal equilibrium price vector $\underline{p}_*$.

Theorem 6.12 ([41, 42])**.** *The algorithm* GreedyAuction *updates a price vector exactly* $\|\underline{p}_* - p_0\|_\infty^+ + \|\underline{p}_* - p_0\|_\infty^-$ *times and then terminates by outputting the unique minimal equilibrium price vector vector* $\underline{p}_*$*.* □

The algorithm GreedyAuction, however, has a disadvantage that the change of a price vector is not monotone. On the other hand, the two-phase auction described below has advantages that any initial vector can be used and that changes of prices are almost monotone. As is clear from its name, the two-phase auction consists of two phases, the ascending phase and the descending phase.

Algorithm 14 (TwoPhaseAuction)**.**
Step 0: Select an initial price vector $p_0 \in \mathbb{Z}_+^n$ arbitrarily, set $p := p_0$, and go to Ascending Phase.
[Ascending Phase]
Step A1: If $L'(p; d) \geq 0$ holds for every $d \in \{0, +1\}^n$, then go to Descending Phase.
Step A2: Let $d = d_* \in \{0, +1\}^n$ be a (unique) minimal minimizer of $L'(p; d)$.
Set $p := p + d_*$ and go to Step A1.
[Descending Phase]
Step D1: If $L'(p; d) \geq 0$ holds for every $d \in \{0, -1\}^n$, then output p and stop.
Step D2: Let $d = d_* \in \{0, -1\}^n$ be a (unique) minimal minimizer of $L'(p; d)$.
Set $p := p + d_*$ and go to Step D1. □

While the two-phase auction has advantages as mentioned above, the number of iterations can be larger than that of the greedy auction. We can show that the number of iterations of the two-phase auction is at most three times of that of greedy auction.

Theorem 6.13 ([41, 42])**.** *The algorithm* TwoPhaseAuction *terminates with outputting the unique minimal equilibrium price vector* $\underline{p}_*$*. The number of updates of a price vector is at most* $\|\underline{p}_* - p_0\|_\infty^+ + \|\underline{p}_* - p_0\|_\infty^-$ *in the ascending phase and at most* $2(\|\underline{p}_* - p_0\|_\infty^+ + \|\underline{p}_* - p_0\|_\infty^-)$ *in the descending phase.* □

Acknowledgement

The author thanks Kazuo Murota for communicating Propositions 4.1 and 4.7 and for giving valuable comments on the manuscript. This work is supported by JSPS/MEXT KAKENHI Grand Numbers 26280004, 15K00030, 15H00848.

A. Appendix: Proof of Proposition 4.1

Let $p_* \in \arg\min g$ be the minimal minimizer under the condition $p_* \geq p$; such p_* is uniquely determined and satisfies $\|p_* - p\|_\infty = \hat{\mu}(p)$. By the definitions of $\hat{\mu}(p)$ and $\mu(p)$, we have

$$\hat{\mu}(p) = \|p_* - p\|_\infty = \|p_* - p\|_\infty^+ = \|p_* - p\|_\infty^+ + \|p_* - p\|_\infty^- \geq \mu(p). \tag{A.1}$$

To show that the inequality (A.1) holds with equality, we assume, to the contrary, that $\|p_* - p\|_\infty^+ + \|p_* - p\|_\infty^- > \mu(p)$ and derive a contradiction.

Let $q_* \in \arg\min g$ be a minimizer with $\|q_* - p\|_\infty^+ + \|q_* - p\|_\infty^- = \mu(p)$; we assume that q_* minimizes the value $\|q_* - p\|_\infty^-$ among all such vectors. Since

$$\|q_* - p\|_\infty^+ + \|q_* - p\|_\infty^- = \mu(p) < \|p_* - p\|_\infty^+ + \|p_* - p\|_\infty^- = \|p_* - p\|_\infty,$$

we have $p_* \neq q_*$ and $\|q_* - p\|_\infty^- > 0$. Hence, $q_*(j) < p(j)$ holds for some $j \in N$.

By the translation submodularity (2.4) with $\alpha = \max_{i \in N}\{p_*(i) - q_*(i)\} - 1$, we have

$$g(p_*) + g(q_*) \geq g(p_* \wedge (q_* + \alpha \mathbf{1})) + g((p_* - \alpha \mathbf{1}) \vee q_*) = g(p_* - \chi_Y) + g(q_* + \chi_Y), \quad \text{(A.2)}$$

where $Y = \arg\max_{i \in N}\{p_*(i) - q_*(i)\}$. Note that $\alpha \geq 0$ holds since $p_*(j) \geq p(j) > q_*(j)$. Since $p_*, q_* \in \arg\min g$, the inequality (A.2) implies $p_* - \chi_Y, q_* + \chi_Y \in \arg\min g$. We have

$$\|(q_* + \chi_Y) - p\|_\infty^+ \leq \|q_* - p\|_\infty^+ + \|\chi_Y\|_\infty^+ = \|q_* - p\|_\infty^+ + 1.$$

As shown below, we have $X \subseteq Y$ for $X = \arg\max_{i \in N}\{p(i) - q_*(i)\}$, implying that

$$\|(q_* + \chi_Y) - p\|_\infty^- = \|q_* - p\|_\infty^- - 1. \quad \text{(A.3)}$$

Hence, we have

$$\|(q_* + \chi_Y) - p\|_\infty^+ + \|(q_* + \chi_Y) - p\|_\infty^- \leq \|q_* - p\|_\infty^+ + \|q_* - p\|_\infty^- = \mu(p),$$

which, together with the definition of $\mu(p)$, implies $\|(q_*+\chi_Y)-p\|_\infty^+ + \|(q_*+\chi_Y)-p\|_\infty^- = \mu(p)$. This, however, contradicts the choice of q_* since (A.3) holds. Therefore, the inequality in (A.1) holds with equality.

To conclude the proof, we show $X \subseteq Y$. Since $p_* - \chi_Y \in \arg\min g$, the choice of p_* implies that $p_* - \chi_Y \not\geq p$. Therefore, it holds that $p_*(h) = p(h)$ for some $h \in Y$. For each $i \in X$, we have

$$p_*(i) - q_*(i) \geq p(i) - q_*(i) \geq p(h) - q_*(h) = p_*(h) - q_*(h),$$

where the first inequality is by $p_* \geq p$ and the second by the definition of X. The inclusion $X \subseteq Y$ follows from the inequality above, $h \in Y$, and the definition of Y.

References

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

[2] L. M. Ausubel: An efficient dynamic auction for heterogeneous commodities, American Economic Review 96 (2006) 602–629.

[3] J. M. Bioucas-Dias and G. Valadão: Phase unwrapping via graph cuts, IEEE Transactions on Image Processing 16 (2007) 698–709.

[4] L. Blumrosen and N. Nisan: Combinatorial auction, in: N. Nisan, T. Roughgarden, É. Tardos, and V. V. Vazirani (eds.) Algorithmic Game Theory, Cambridge University Press, Cambridge, 2007, pp. 267–299.

[5] Y. B. Boykov, O. Veksler, and R. Zabih: Fast approximate energy minimization via graph cuts, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (2001) 1222–1239.

[6] N.-k. Chung and D.-w. Tcha: A dual algorithm for submodular flow problems, Operations Research Letters 10 (1991) 489–495.

[7] P. Cramton, Y. Shoham, and R. Steinberg: Combinatorial Auctions, MIT Press, Cambridge, MA, 2006.

[8] J. Darbon and M. Sigelle: Image restoration with discrete constrained total variation, part I: fast and exact optimization, Journal of Mathematical Imaging and Vision 26 (2006) 261–276.

[9] A. W. M. Dress and W. Wenzel: Valuated matroid: A new look at the greedy algorithm, Applied Mathematics Letters 3 (1990), 33–35.

[10] A. W. M. Dress and W. Wenzel: Valuated matroids, Advances in Mathematics 93 (1992), 214–250.

[11] S. Fujishige: Submodular Functions and Optimization, 2nd edition, Annals of Discrete Mathematics 58, Elsevier, Amsterdam, 2005.

[12] S. Fujishige and K. Murota: Notes on L-/M-convex functions and the separation theorems, Mathematical Programming 88 (2000) 129–146.

[13] S. Fujishige, K. Murota, and A. Shioura: Monotonicity in steepest ascent algorithms for polyhedral L-concave functions, Journal of Operations Research Society of Japan 58 (2015) 184–208.

[14] S. Fujishige and Z. Yang: A note on Kelso and Crawford's gross substitutes condition, Mathematics of Operations Research 28 (2003) 463–469.

[15] H. Hirai: Discrete convexity for multiflows and 0-extensions, in: Proceedings of the 8th Japanese–Hungarian Symposium on Discrete Mathematics and Its Applications (2013) 209–223.

[16] H. Hirai: L-extendable functions and a proximity scaling algorithm for minimum cost multiflow problem, Discrete Optimization 18 (2015) 1–37.

[17] S. Geman and D. Geman: Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, IEEE Transactions on Pattern Analysis and Machine Intelligence 6 (1984) 721–741.

[18] F. Gul and E. Stacchetti: Walrasian equilibrium with gross substitutes, Journal of Economic Theory 87 (1999) 95–124.

[19] F. Gul and E. Stacchetti: The English auction with differentiated commodities, Journal of Economic Theory 92 (2000) 66–95.

[20] R. Hassin: The minimum cost flow problem: a unifying approach to dual algorithms and a new tree-search algorithm, Mathematical Programming 25 (1983) 228–239.

[21] W.T. Huh and G. Janakiraman: On the optimal policy structure in serial inventory systems with lost sales, Operations Research 58 (2010) 486–491.

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

[23] A. S. Kelso and V. P. Crawford: Job matching, coalition formation and gross substitutes, Econometrica 50 (1982) 1483–1504.

[24] V. Kolmogorov: Primal-dual algorithm for convex markov random fields, Technical Report MSR-TR-2005-117, Microsoft, September 2005.

[25] V. Kolmogorov and A. Shioura: New algorithms for convex cost tension problem with application to computer vision, Discrete Optimization 6 (2009) 378–393.

[26] K. Murota: Convexity and Steinitz's exchange property, Advances in Mathematics 124 (1996) 272–311.
[27] K. Murota: Discrete convex analysis, Mathematical Programming 83 (1998) 313–371.
[28] K. Murota: Discrete Convex Analysis, Kyoritsu Shuppan, Tokyo, 2001 (in Japanese).
[29] K. Murota: Discrete Convex Analysis, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2003.
[30] K. Murota: On steepest descent algorithms for discrete convex functions, SIAM Journal on Optimization 14 (2003) 699–707.
[31] 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.
[32] K. Murota: Primer of Discrete Convex Analysis, Kyoritsu Shuppan, Tokyo, 2007 (in Japanese).
[33] K. Murota: Recent developments in discrete convex analysis, in: W. Cook, L. Lovász and J. Vygen (eds.) Research Trends in Combinatorial Optimization, Springer, Berlin, 2009, Chapter 11, pp. 219–260.
[34] K. Murota and A. Shioura: M-convex function on generalized polymatroid, Mathematics of Operations Research 24 (1999) 95–105.
[35] K. Murota and A. Shioura: Extension of M-convexity and L-convexity to polyhedral convex functions, Advances in Applied Mathematics 25 (2000) 352–427.
[36] K. Murota and A. Shioura: Conjugacy relationship between M-convex and L-convex functions in continuous variables, Mathematical Programming 101 (2004) 415–433.
[37] K. Murota and A. Shioura: Quadratic M-convex and L-convex functions, Advances in Applied Mathematics 33 (2004) 318–341.
[38] K. Murota and A. Shioura: Discrete Convex Analysis and Optimization Algorithms, Asakura Shoten, Tokyo, 2013 (in Japanese).
[39] K. Murota and A. Shioura: Dijkstra's algorithm and L-concave function maximization, Mathematical Programming 145 (2014) 163–177.
[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, A. Shioura, and Z. Yang: Computing a Walrasian equilibrium in iterative auctions with multiple differentiated items, extended abstract in: Proceedings of the 24th International Symposium on Algorithms and Computation (ISAAC 2013), LNCS 8283, 468–478; full paper version in: Technical Report METR 2013-10, University of Tokyo, June 2013.
[42] K. Murota, A. Shioura, and Z. Yang: Time bounds for iterative auctions: a unified approach by discrete convex analysis, Discrete Optimization 19 (2016) 36–62.
[43] R.T. Rockafellar: Convex Analysis, Princeton University Press, Princeton, NJ, 1970.
[44] H. Saito: The combinatorial English auction of Gul and Stacchetti viewed from discrete convex analysis, Master's Thesis, Osaka University, 2003.
[45] A. Schrijver: A combinatorial algorithm minimizing submodular functions in strongly polynomial time, Journal of Combinatorial Theory B 80 (2000) 346–355.
[46] A. Schrijver: Combinatorial Optimization—Polyhedra and Efficiency, Springer, Berlin, 2003.
[47] A. Shioura: L-convex function minimization algorithms: connection between discrete convex analysis and other fields of research, Proceedings of 27th RAMP Symposium (Operations Research Society of Japan, 2015), 17–46 (in Japanese).

[48] P. Zipkin: On the structure of lost-sales inventory models, Operations Research 56 (2008) 937–944.

[49] A. Zomet, A. Levin, S. Peleg, and Y. Weiss: Seamless image stitching by minimizing false edges, IEEE Transactions on Image Processing 15 (2006) 969–977.