MATHEMATICAL ENGINEERING
TECHNICAL REPORTS

Valuated Matroid-Based Algorithm for Submodular Welfare Problem

Takanori MAEHARA and Kazuo MUROTA

METR 2013–31 November 2013

DEPARTMENT OF MATHEMATICAL INFORMATICS
GRADUATE SCHOOL OF INFORMATION SCIENCE AND TECHNOLOGY
THE UNIVERSITY OF TOKYO
BUNKYO-KU, TOKYO 113-8656, JAPAN

WWW page: http://www.keisu.t.u-tokyo.ac.jp/research/techrep/index.html

Valuated Matroid-Based Algorithm for Submodular Welfare Problem

Takanori MAEHARA

National Institute of Informatics
JST, ERATO, Kawarabayashi Project
maehara@nii.ac.jp

Kazuo MUROTA

Department of Mathematical Informatics, University of Tokyo
murota@mist.i.u-tokyo.ac.jp

November 2013

Abstract

An algorithm for the submodular welfare problem is proposed based on the theory of discrete convex analysis. The proposed algorithm is a heuristic method built upon the valuated matroid partition algorithms, and gives the exact optimal solution for a reasonable subclass of submodular welfare problems. The algorithm has a guaranteed approximation ratio for a special case. Computational results show fairly good performance of the proposed algorithm.

1 Introduction

1.1 Submodular welfare problem

Let $V = \{1, \ldots, n\}$. A function $w : 2^V \to \mathbb{R}$ is said to be submodular if

$$w(X) + w(Y) \geq w(X \cap Y) + w(X \cup Y) \tag{1.1}$$

for all $X, Y \subseteq V$. The problem addressed in this paper is the so-called *submodular welfare problem* [27], which is an optimization problem of the form:

$$\begin{array}{ll} \text{maximize} & \Phi(X_1, \ldots, X_m) := w_1(X_1) + \cdots + w_m(X_m) \\ \text{subject to:} & \{X_1, \ldots, X_m\} \text{ is a partition of } V, \end{array} \tag{1.2}$$

where $w_1, \ldots, w_m : 2^V \to \mathbb{R}$ are given submodular functions. This is a fundamental problem in discrete optimization, where the name of "submodular welfare problem" is used primarily in the area of combinatorial auction.

Suppose that, in a combinatorial auction, there are m players and n items. Each player i has a *utility function* $w_i : 2^V \to \mathbb{R}$ that represents his utility of a combination of items. Each utility function may naturally be assumed to be submodular, in accordance with the principle of "decreasing marginal utility" in economics. The objective of combinatorial auction is to maximize the "social welfare," the sum of the utilities of the m players, by distributing n items to the players.

Many important optimization problems are contained in this class of problems. For example, set k-cover [1], max k-cut [20], budgeted allocation [3, 23], generalized assignment problem [19], and so on.

The submodular welfare problem is NP-hard, as it contains submodular set function maximization, which is known to be NP-hard. Accordingly, several approximation algorithms are proposed in the literature. Lehmann, Lehmann, and Nisan [27] showed that a simple greedy algorithm gives 1/2-approximation. Mirrokni, Schapira, and Vondrák [30] showed that approximating better than $1-1/e$ requires exponentially many function evaluations, regardless of the P $\neq$ NP conjecture. Vondrák [45] proposed the continuous greedy algorithm, a randomized algorithm, which provides a $(1 - 1/e)$-approximation, attaining the best possible approximation ratio.

1.2 Valuated matroid partition problem

The purpose of this paper is to design an algorithm for the submodular welfare problem by fully utilizing the algorithmic results in discrete convex analysis [35, 36, 37]. Specifically, we focus on an important subclass of submodular functions, $\mathrm{M}^\natural$-concave functions, to be called matroid valuations in this paper.

In combinatorial optimization, it is customary to regard submodular functions as a discrete analogue of convex functions. However, submodular functions also have concavity aspects [15, 29]. In particular, for a concave function $\varphi : \mathbb{R} \to \mathbb{R}$, the set function $w : 2^V \to \mathbb{R}$ defined by

$$w(X) := \varphi(|X|) \tag{1.3}$$

is submodular. Concavity aspects of submodular functions are captured by $M^\natural$-*concavity* in discrete convex analysis [21, 36].

Let us say that a set function $w : 2^V \to \mathbb{R} \cup \{-\infty\}$ is a *matroid valuation* (or $\mathrm{M}^\natural$-concave functions on 0-1 vectors) if for all $X, Y \subseteq V$ and $x \in X \setminus Y$,

$$\begin{aligned} w(X) + w(Y) \leq \max\{&w(X \setminus \{x\}) + w(Y \cup \{x\}), \\ &\max_{y \in Y \setminus X} [w(X \setminus \{x\} \cup \{y\}) + w(Y \cup \{x\} \setminus \{y\})]\}. \end{aligned} \tag{1.4}$$

This is an extension of the valuated matroid of Dress and Wenzel [13, 14], which is a function on bases, to a function on independent sets. A matroid rank function is known to be a matroid valuation, and accordingly, we may regard matroid valuation as a generalization of matroid rank function. The function (1.3) is also a matroid valuation. A matroid valuation, in general, is known to be submodular.

When all the utility functions w_i are matroid valuations, the submodular welfare problem coincides with the *valuated matroid partition problem* [31, 32, 35], which has been studied in depth in discrete convex analysis. In economic terms, a matroid valuation corresponds to a utility function of *gross substitutes* [22]. Hence the valuated matroid partition problem corresponds to a combinatorial auction of gross substitutes goods [27].

We give an illustrative example of submodular functions and matroid valuations.

Example 1.1. Let V be a finite set. For a univariate concave function $\varphi : \mathbb{R} \to \mathbb{R}$, the function $w : 2^V \to \mathbb{R}$ defined by (1.3) is a matroid valuation (hence is a submodular function). Since $w(X)$ depends only on the cardinality of X (i.e., the number of items), this is a typical case of "substitutivity" of items. Next, suppose that a family of subsets $A_1, \ldots, A_l$ of V is specified together with univariate concave functions $\varphi_1, \ldots, \varphi_l : \mathbb{R} \to \mathbb{R}$. Then the function

$$w(X) := \varphi_1(|X \cap A_1|) + \cdots + \varphi_l(|X \cap A_l|) \tag{1.5}$$

is, in general, a submodular function. This function represents the substitutivity of items in each subset A_i. It is known that if the set family $\{A_1, \ldots, A_l\}$ forms a laminar family, i.e., if for all $i, j = 1, \ldots, l$,

$$A_i \subseteq A_j, \quad \text{or} \quad A_i \supseteq A_j, \quad \text{or} \quad A_i \cap A_j = \emptyset, \tag{1.6}$$

then the function w defined by (1.5) is a matroid valuation. ■

Our focus on matroid valuations is motivated by the computational advantage that they offer. They can be maximized in polynomial time by a simple greedy algorithm. Furthermore, the *valuated matroid partition problem* can be solved, to exact optimality, in polynomial time by valuated matroid partition algorithms. In other words, the valuated matroid partition problem is a solvable subclass of the submodular welfare problem, which fact was also observed in Lehmann, Lehmann, and Nisan [27]. It is noted in this connection that the valuated matroid partition problem is a special case of the *valuated matroid intersection problem*, which is fully studied in discrete convex analysis and for which (strongly) polynomial time algorithms are available [31, 32].

1.3 Our contribution

A heuristic algorithm for the submodular welfare problem is proposed. The basic idea is to apply the valuated matroid partition algorithm to the submodular welfare problem with some appropriate modifications. If all the utility functions are indeed matroid valuations, the proposed algorithm coincides with the valuated matroid partition algorithm, and exactly solves the problem. We expect that if the utility functions are "close enough" to matroid valuations, the algorithm will find a good solution, which is verified in numerical experiments.

We here give an overview of the proposed algorithm, whereas the details are given in Section 4. As a matter of course, if the given functions are not matroid valuations, the valuated matroid partition algorithm fails to find an optimal solution, and, in the worst case, the algorithm falls into an infinite loop. We here modify the valuated matroid partition algorithm to avoid this worst case behavior as follows.

The augmenting path algorithm for the valuated matroid partition can be regarded as an extension of the standard algorithm for the maximum-size bipartite matching problem (see Section 3 for details). The algorithm iteratively finds a shortest path on an auxiliary network, and updates the solutions by augmenting the path. Here we have the following properties when the functions are all matroid valuations.

(1) If the network has a negative cycle, then an improved solution, having a better objective value, can be obtained.

(2) If the network has no negative cycles, then the current solution is optimal.

The property (1) guarantees the finite termination of the algorithm and the property (2) guarantees the (global) optimality of the solution at the termination.

If the given functions are not matroid valuations, the above two properties are not necessarily guaranteed. The failure of the property (1) is more problematic since it may cause an infinite loop. To avoid an infinite loop, we modify that part of the algorithm which deals with negative cycles. This is the main novel component of our algorithm. We also have to modify some other parts to cope with the difficulties caused by non-matroidal valuations.

The resulting algorithm has a theoretical guarantee of approximation ratio under a certain assumption, which is very often satisfied in practice. The algorithm performs quite well for a number of practical problems used in our computational experiments.

1.4 Organization of the paper

The paper is organized as follows. In Section 2, we review matroid valuations. In Section 3, we introduce algorithms for the valuated matroid partition problem. In Section 4, we describe how to modify the algorithm to cope with functions that are not matroid valuations. Computational results are shown in Section 5. We end with conclusion in Section 6.

2 Valuated matroid

In this section, we introduce valuated matroid, which is a quantitative extension of matroid.

Let V be a finite set. A set function $w : 2^V \to \mathbb{R} \cup \{-\infty\}$ is a *matroid valuation* (or *$M^\natural$-concave function* on 0-1 vectors) if for all $X, Y \subseteq V$ and $x \in X \setminus Y$,

$$w(X) + w(Y) \leq \max\{w(X \setminus \{x\}) + w(Y \cup \{x\}), \max_{y \in Y \setminus X} [w(X \setminus \{x\} \cup \{y\}) + w(Y \cup \{x\} \setminus \{y\})]\}. \quad (2.1)$$

We call

$$\operatorname{dom} w := \{X \subseteq V : w(X) > -\infty\}$$

the effective domain of w. A matroid valuation, as defined here, coincides with the valuated matroid (V, w) in the sense of Dress and Wenzel [13, 14] if $|X| = |Y|$ for all $X, Y \in \operatorname{dom} w$. See [35, Chapter 5] and [36, Chapter 6] for details about valuated matroids and M$^\natural$-concave functions.

In economic terms, a matroid valuation corresponds to a utility function of *gross substitutes*. Let $f : 2^V \to \mathbb{R}$ be a utility function of an agent. Given a price vector $p \in \mathbb{R}^V$, the agent solves the optimization problem

$$\text{maximize} \quad f(X) - p(X), \qquad \text{where } p(X) := \sum_{x \in X} p(x).$$

Let $D(p)$ be the set of optimal solutions of this problem, called the demand set. The gross substitutes condition, introduced by Kelso and Crawford [26], is the following:

> **(GS)** For any two price vectors p and q such that $p(x) \leq q(x)$ for all $x \in V$, and for any $X \in D(p)$, there exists $Y \in D(q)$ such that $X \cap \{x \in V : p(x) = q(x)\} \subseteq Y$.

Fujishige and Yang [22] pointed out that a function f satisfies the condition (GS) if and only if f is a matroid valuation.

Some fundamental examples of matroid valuations follow.

Example 2.1. A matroid rank function is a matroid valuation [21, 37]. ■

Example 2.2. The negative of the Hamming distance to an independent set of a matroid is a matroid valuation. That is,

$$w(X) := -\min\{|X \Delta Y| : Y \text{ is an independent set}\} \quad (2.2)$$

is a matroid valuation. Note that $w(X)$ is equal to the rank of X minus $|X|$. ■

Example 2.3. A linear function on independent sets of a matroid is a matroid valuation. The effective domain of this matroid valuation is the family of independent sets. ■

Example 2.4. A weighted matroid rank function

$$w(X) := \max\{\sum_{x\in I} c(x) : I \subseteq X \text{ is an independent set } \}$$

with a nonnegative vector $c \in \mathbb{R}^V$ is a matroid valuation [41]. ■

Example 2.5. For a concave function $\varphi : \mathbb{R} \to \mathbb{R}$, the set function $w : 2^V \to \mathbb{R}$ defined by

$$w(X) := \varphi(|X|)$$

is a matroid valuation. ■

Example 2.6. Let $\{A_1, \ldots, A_l\}$ be a laminar family of V and $\varphi_1, \ldots, \varphi_l : \mathbb{R} \to \mathbb{R}$ be univariate concave functions. Then

$$w(X) := \varphi_1(|X \cap A_1|) + \cdots + \varphi_l(|X \cap A_l|) \tag{2.3}$$

is a matroid valuation. See (1.6) for the definition of a laminar family. ■

Example 2.7. The sum of a matroid valuation $w : 2^V \to \mathbb{R}$ and a linear function $p : 2^V \to \mathbb{R}$ is also a matroid valuation. ■

Example 2.8. For a matrix $(a(x, y) : x, y \in V)$ with $a(x, y) = a(y, x) \in \mathbb{R}$, the function

$$w(X) := \sum_{x\in X, y\in X} a(x, y)$$

is a matroid valuation if and only if $a(x, y) \leq 0$ and for all $x, y \in V$ and

$$a(x, y) \leq \max\{a(x, z), a(y, z)\} \quad \text{if} \quad z \notin \{x, y\}.$$

■

Example 2.9. Let $G = (V, W; E)$ be a bipartite graph with vertex set $V \cup W$ and edge set E, and let $\gamma : E \to \mathbb{R}$ be an edge weight. Then

$$w(X) := \max\{\sum_{e\in M} \gamma(e) : M \subseteq E \text{ is a matching}, \partial M \cap V = X\}$$

for $X \subseteq V$ is a matroid valuation on V. Here ∂M denotes the set of vertices incident to (i.e., covered by) M. ■

The following three examples are from economics.

Example 2.10. A utility function of a unit demand preference [25] is defined by

$$w(X) := \max_{x \in X} a(x)$$

for a nonnegative vector $a \in \mathbb{R}^V$, where $w(\emptyset) = 0$ by definition. This is a matroid valuation. ■

Example 2.11. Let $u : 2^V \to \mathbb{R}$ be a utility function and $p \in \mathbb{R}^V$ be a vector representing the price of each item. The function

$$w(X) := u(X) - p(X)$$

denotes the "actual utility" of buying items X by price $p(X)$. Such utility is called transferable utility or quasi-linear utility.

If $u(X)$ is a matroid valuation, by Example 2.7, the corresponding transferable utility function is also a matroid valuation. A special case with $u(X) = \varphi(|X|)$ is considered by Beviá, Quinzii, and Silva [7]. ■

Example 2.12. Let V be a finite set and let $V_1, \dots, V_N$ be a partition of V and $d_1, \dots, d_N \in \mathbb{Z}_+$ be parameters. The function defined by

$$w(X) = \sum_{j=1}^{N} \min\{|X \cap V_j|, d_j\} \tag{2.4}$$

is a matroid rank function and hence is a matroid valuation (cf. Example 2.1). This is the rank function of a direct sum of uniform matroids (e.g., [18]). In the case of $d_1 = \cdots = d_N = 1$, this function represents the rank function of a partition matroid [39]. The function (2.4) admits the following economic interpretation: V corresponds to a set of items, and $V_1, \dots, V_N$ correspond to categories of items. If a buyer wants d_1 items in category V_1, d_2 items in category V_2, and so on, his utility function is represented by the function (2.4). ■

Matroid valuations have many properties analogous to concave functions. For example, a local maximality implies the global maximality, the concave conjugate (Legendre-Fenchel conjugate) is well-defined, the subdifferential exists for all $x \in \operatorname{dom} w$, and so on.

The sum of a matroid valuation and a linear function is a matroid valuation, and a supremum convolution (aggregation) of matroid valuations

$$(w_1 \square w_2)(X) := \sup_{Y} \{w_1(Y) + w_2(X \setminus Y)\}$$

is a matroid valuation. However, a sum of matroid valuations is not necessarily a matroid valuation. This is an important difference between matroid

valuations and concave functions. Nevertheless, a sum of two matroid valuations is an tractable object (see Theorem 2.13 below), whereas a sum of three or more matroid valuations is intractable.

The following problem is known as the valuated matroid partition problem [31, 32, 35]. Given matroid valuations $w_1, \ldots, w_m : V \to \mathbb{R}$,

$$\begin{array}{ll} \text{maximize} & \Phi(X_1, \ldots, X_m) := w_1(X_1) + \cdots + w_m(X_m), \\ \text{subject to:} & \{X_1, \ldots, X_m\} \text{ is a partition of } V. \end{array} \tag{2.5}$$

For a valuated matroid partition problem, the following theorem, called *valuated matroid intersection theorem*, plays a fundamental role. This theorem is an extension of the celebrated matroid intersection theorem of Edmonds [15, 21], and can also be understood as a discrete version of the hyperplane separation theorem.

Theorem 2.13 (Valuated matroid intersection [31], Theorem 5.2.40 in [35])**.** Let $w_1, w_2 : 2^V \to \mathbb{R}$ be matroid valuations, and consider the optimization problem:

$$\text{maximize } w_1(X) + w_2(X). \tag{2.6}$$

- X^* is an optimal solution of (2.6) if and only if there exists a vector $p \in \mathbb{R}^V$ such that

$$X^* \in \operatorname*{argmax}_X \{w_1(X) - p(X)\},$$
$$X^* \in \operatorname*{argmax}_X \{w_2(X) + p(X)\}.$$

- If both w_1 and w_2 are integer-valued, we can take an integer vector as the p above. ■

An adaptation of this theorem to the valuated matroid partition problem is called *valuated matroid partition theorem*.

Corollary 2.14 (Valuated matroid partition)**.** Let $w_1, \ldots, w_m : 2^V \to \mathbb{R}$ be matroid valuations. A partition $(X_1^*, \ldots, X_m^*)$ of V is an optimal solution of the valuated matroid partition problem (2.5) if and only if there exists $p \in \mathbb{R}^V$ such that $p(V) = 0$ and

$$X_1^* \in \operatorname*{argmax}_X \{w_1(X) - p(X)\},$$
$$\vdots$$
$$X_m^* \in \operatorname*{argmax}_X \{w_m(X) - p(X)\}.$$

■

In the next section, we introduce algorithms for the valuated matroid partition problem, which are based on the valuated matroid partition theorem above.

In the rest of the paper, we assume, to simplify the arguments, that a valuation w is defined (effectively) on 2^V, i.e., $w(X) > -\infty$ for all $X \subseteq V$.

3 Valuated matroid partition algorithms

In this section, we review algorithms for valuated matroid partition, which form the basis of our algorithm to be described in Section 4.

The valuated matroid partition problem (2.5) can be formulated as a valuated matroid intersection problem, which is studied well in discrete convex analysis and for which (strongly) polynomial time algorithms (of several types) are available [31, 32].

Let $V^{(1)}, \ldots, V^{(m)}$ be disjoint copies of V. For $v \in V$, we write $v^{(i)}$ for the copy of v in $V^{(i)}$. Let $\Phi : 2^{V^{(1)} \cup \cdots \cup V^{(m)}} \to \mathbb{R}$ be a matroid valuation on $V^{(1)} \cup \cdots \cup V^{(m)}$ defined as the direct sum of $w_1, \ldots, w_m$ by

$$\Phi(X_1, \ldots, X_m) := w_1(X_1) + \cdots + w_m(X_m), \tag{3.1}$$

where $X_i \subseteq V^{(i)}$ $(i = 1, \ldots, m)$, and let δ be another matroid valuation on $V^{(1)} \cup \cdots \cup V^{(m)}$ defined by

$$\delta(X_1, \ldots, X_m) := \begin{cases} 0, & \sum_{i=1}^{m} |X_i \cap \{v^{(i)}\}| \leq 1 \quad (\forall v \in V), \\ -\infty, & \text{otherwise.} \end{cases} \tag{3.2}$$

Note that δ is the indicator function of a partition matroid. Then the problem:

$$\text{maximize} \quad \Phi(X_1, \ldots, X_m) + \delta(X_1, \ldots, X_m) \tag{3.3}$$

is a valuated matroid intersection problem (2.6) that is equivalent to the valuated matroid partition problem (2.5); see Remark 3.1.

Remark 3.1. In (2.6), we have assumed that the functions w_1 and w_2 are finite-valued for all subsets. To meet this condition we can use

$$\delta_\beta(X) := -\beta \cdot \min\{|X \Delta Y| : \delta(Y) = 0\} \tag{3.4}$$

with a sufficiently large β. Then the problem (3.3) is equivalent to the following problem:

$$\text{maximize} \quad \Phi(X_1, \ldots, X_m) + \delta_\beta(X_1, \ldots, X_m) \tag{3.5}$$

with finite-valued functions Φ and δ_β. ■

The algorithms for the valuated matroid partition problem are based on this reduction to the valuated matroid intersection problem. We here introduce two types of algorithms: cycle-canceling algorithm (primal algorithm) and augmenting path algorithm (primal-dual algorithm).

We construct a network as a bipartite graph G that represents the constraint of the partition matroid. The left vertices are V. The right vertices are $V^{(1)} \cup \cdots \cup V^{(m)}$. The (undirected) edges are defined by

$$E = \{\{v, v^{(i)}\} : v \in V, i = 1, \ldots, m\}.$$

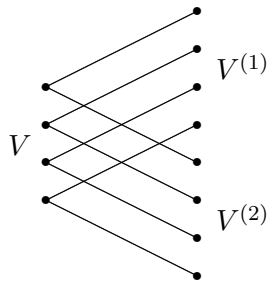

Figure 3.1: The bipartite network G for the valuated matroid partition problem ($m = 2$).

Then a matching M of G corresponds to a subpartition of V by

$$(X_1, \ldots, X_m) := (\partial M \cap V^{(1)}, \ldots, \partial M \cap V^{(m)}), \tag{3.6}$$

where ∂M denotes the set of matched vertices and $v^{(i)}$ and v are identified. See Figure 3.1 for the bipartite network. Note that $v \in X_i$ if and only if $\{v, v^{(i)}\} \in M$. To simplify the notation, we write $\Phi(M)$ for the function (3.1) with the understanding above. That is,

$$\begin{aligned} \Phi(M) &:= \Phi(\partial M \cap V^{(1)}, \ldots, \partial M \cap V^{(m)}) \\ &= w_1(\partial M \cap V^{(1)}) + \cdots + w_m(\partial M \cap V^{(m)}). \end{aligned} \tag{3.7}$$

Just as in the standard treatment of bipartite matching, we orient the edges E as

$$(u, u^{(i)}) \qquad \text{if } \{u, u^{(i)}\} \notin M, \tag{3.8}$$

$$(u^{(i)}, u) \qquad \text{if } \{u, u^{(i)}\} \in M. \tag{3.9}$$

Basically, the valuated matroid partition algorithm finds the maximum weight maximum cardinality matching on this network, whose edge weights represent the matroid valuation.

We encode the matroid valuations to weights γ of the network. Let M be a current matching and $(X_1, \ldots, X_m)$ be the corresponding subpartition. We add a source-vertex s, a sink-vertex t, and the following edges to the network:

$$E_\circ := \{(s, u) : u \notin \partial M\}, \tag{3.10}$$

$$E_\bullet := \{(v, s) : v \in \partial M\}, \tag{3.11}$$

$$E_+ := \{(u^{(i)}, t) : u^{(i)} \notin \partial M, i = 1, \ldots, m\}, \tag{3.12}$$

$$E_- := \{(t, v^{(i)}) : v^{(i)} \in \partial M, i = 1, \ldots, m\}, \tag{3.13}$$

$$E_\times := \{(u^{(i)}, v^{(i)}) : u^{(i)} \notin \partial M, v^{(i)} \in \partial M, i = 1, \ldots, m\}. \tag{3.14}$$

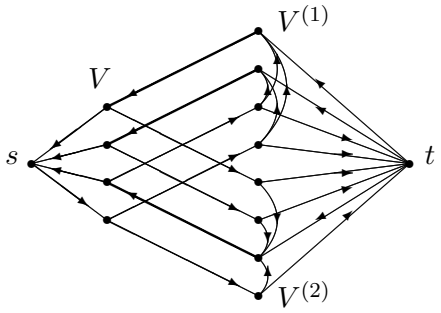

Figure 3.2: The auxiliary network $G(M)$ for the valuated matroid partition problem.

Here, the edge (s, u) in $E_\circ$ denotes that u is in some X_i $(i = 1, \ldots, m)$ and the edge (v, s) in $E_\bullet$ denotes that v is not in any X_i $(i = 1, \ldots, m)$. The edge $(u^{(i)}, t)$ in E_+ corresponds to the operation of *inserting* element u to X_i, the edge $(t, v^{(i)})$ in E_- corresponds to the operation of *removing* element v from X_i, and the edge $(u^{(i)}, v^{(i)})$ in $E_\times$ corresponds to the operation of *exchanging* elements $u \notin X_i$ and $v \in X_i$. We define edge weights γ of these edges corresponding to these operations as follows:

$$\gamma(u^{(i)}, t) = w_i(X) - w_i(X \cup \{u\}), \tag{3.15}$$

$$\gamma(t, v^{(i)}) = w_i(X) - w_i(X \setminus \{v\}), \tag{3.16}$$

$$\gamma(u^{(i)}, v^{(i)}) = w_i(X) - w_i(X \setminus \{v\} \cup \{u\}), \tag{3.17}$$

and $\gamma(e) = 0$ for other edges. We write $G(M)$ for this weighted network, i.e.,

$$\begin{aligned} G(M) &:= (V(G(M)), E(G(M))), \\ V(G(M)) &:= \{s\} \cup V \cup V^{(1)} \cup \cdots \cup V^{(m)} \cup \{t\}, \\ E(G(M)) &:= E \cup E_\circ \cup E_\bullet \cup E_+ \cup E_- \cup E_\times. \end{aligned} \tag{3.18}$$

See Figure 3.2 for the auxiliary network. We emphasize that $G(M)$ is no longer bipartite, but we talk of a matching to mean a matching on the bipartite graph induced on E. Both the *cycle-canceling algorithm* and the *augmenting path algorithm* work on this network.

We first explain the cycle-canceling algorithm, which starts from a maximum cardinality matching and iteratively updates the matching to an optimal matching. As in the standard argument for bipartite matchings, a cycle C in $G(M)$ induces a new matching

$$M' = (M \Delta C) \cap E, \tag{3.19}$$

where Δ denotes the symmetric difference, i.e., $M \Delta C := (M \setminus C) \cup (C \setminus M)$.

For the valuated matroid partition problem, the most important property is the following theorem that states the relation between optimality for the problem (2.5) and the existence of negative cycles in $G(M)$.

Theorem 3.2.

(1) Let C be a negative cycle with the fewest edges in $G(M)$. Then

$$\Phi(M') = \Phi(M) - \gamma(C). \tag{3.20}$$

(2) If $G(M)$ does not contain negative cycles, then M is an optimal solution. ■

In the above theorem, (1) is proved by "unique max lemma", and (2) is proved by the "upper bound lemma". For more details, see [32, 35].

Theorem 3.2 suggests an algorithm for the valuated matroid partition problem, which iteratively finds a negative cycle with the fewest edges, and forms the symmetric difference of the current matching with the cycle. This type of an algorithm is called the *cycle-canceling algorithm*. Here, for our purpose, we introduce a variant of the valuated matroid partition problem with a cardinality constraint and describe the algorithm for that form.

For a nonnegative integer k, we consider the following problem:

$$\begin{array}{ll} \text{maximize} & \Phi(X_1, \ldots, X_m) := w_1(X_1) + \cdots + w_m(X_m), \\ \text{subject to:} & \{X_1, \ldots, X_m\} \text{ is a subpartition of } V, \\ & |X_1 \cup \cdots \cup X_m| = k. \end{array} \tag{3.21}$$

If $k = |V|$, this problem coincides with the original problem (2.5). It is important that Theorem 3.2 remains true for the problem (3.21).

The cycle-canceling algorithm for (3.21) is shown in Algorithm 1. To

Algorithm 1 Cycle-canceling algorithm

1: Let M be an initial matching of size k (e.g., $M = \{(v_1, v_1^{(1)}), \ldots, (v_k, v_k^{(1)})\}$ for $v_1, \ldots, v_k \in V$)
2: **loop**
3: Find a negative cycle C with the fewest edges
4: **if** there are no negative cycles **then**
5: **return** M as an optimal solution
6: **else**
7: $M \leftarrow (M \Delta C) \cap E$
8: **end if**
9: **end loop**

find a negative cycle with the fewest edges (line 3 of Algorithm 1), we can use the Bellman–Ford shortest path algorithm.

If all valuations $w_1, \ldots, w_m$ are integer-valued, Algorithm 1 terminates in a finite number of iterations because, by Theorem 3.2 (1), the objective value $\Phi(M)$ increases monotonically. This means that Algorithm 1 is a pseudo-polynomial time algorithm. Note that there exists a (strongly) polynomial time algorithm of cycle-canceling type; see [32, 35].

Next, we introduce the augmenting path algorithm. The algorithm starts from the empty matching and in the k-th step, it constructs an optimal solution for the problem (3.21).

Let M be a current matching of size $k-1$. Consider a simple path P that starts from the source-vertex s and terminates at the sink-vertex t. If we update M to

$$M' := (M \Delta P) \cap E \tag{3.22}$$

by augmenting this path, we obtain a new matching of size k. An important observation is the following.

Proposition 3.3. Suppose that $G(M)$ has no negative cycles. Let P be a shortest path that starts from the source-vertex s and terminates at the sink-vertex t, and let $M' := (M \Delta P) \cap E$. Then $G(M')$ has no negative cycles. ■

By the optimality criterion (Theorem 3.2), if we iteratively augment by the shortest paths, the obtained matchings are optimal solution of (3.21) for each size k. This is indeed the augmenting path algorithm [32, 35], which is described as Algorithm 2.

Algorithm 2 Augmenting path algorithm

1: Let $M = \emptyset$ be an initial solution
2: **for** $k = 1, \ldots, n$ **do**
3: Find a shortest path P that starts from the source-vertex s and terminates at the sink-vertex t
4: Update $M \leftarrow (M \Delta P) \cap E$
5: **end for**

Note that, by Proposition 3.3, we can use Dijkstra's shortest path algorithm for line 3 of Algorithm 2.

Algorithms 1 and 2 exactly solve the valuated matroid partition problem. Hence, in particular, the submodular welfare problem with utility functions introduced in Examples 2.1–2.12, which are matroid valuations, can be solved to exact optimality by the valuated matroid partition algorithms.

4 Proposed algorithm

We here propose a heuristic algorithm for the submodular welfare problem based on the valuated matroid partition algorithms. Basically, the algorithm is a combination of Algorithms 1 and 2 with some heuristics to cope with difficulties caused by non-matroidal valuations.

The idea of our algorithm is an optimistic expectation that, if the utility functions are "close enough" to matroid valuations, the algorithms designed for matroid valuations will find reasonably good solutions. As a specific example, we may consider utility functions of the form of (1.5): $w(X) = \varphi_1(|X \cap A_1|) + \cdots + \varphi_l(|X \cap A_l|)$. Such a function is submodular in general, and is a matroid valuation if the underlying set family $\{A_1, \ldots, A_l\}$ is a laminar family (Example 2.6). Our optimistic expectation is that, if the set family is "almost laminar," the valuated matroid partition algorithms will perform reasonably well, and hence can be used as building blocks in designing an algorithm that works reasonably well for general submodular utility functions.

We first observe what will happen if not every utility function w_i is a matroid valuation. Recall that the valuated matroid partition algorithms rely on Theorem 3.2 for the optimality characterization in terms of negative cycles in $G(M)$. When w_i are not matroid valuations, this characterization is no longer true. We have to modify the algorithm to resolve this problem.

We outline the proposed algorithm. Consider the augmenting path algorithm (Algorithm 2) and suppose that we have an optimal matching M of size k. Since the optimality criterion (Theorem 3.2) is not valid, there may possibly be negative cycles in $G(M)$ and hence we may not be able to compute the shortest path in $G(M)$ efficiently. To overcome this situation, we apply the cycle-canceling algorithm (Algorithm 1) to remove negative cycles, before we try to find a shortest path.

Let us discuss the detailed procedure for the cycle-canceling part which is the core of our algorithm. The most crucial issue is the failure of the first statement of Theorem 3.2. Even if we take a negative cycle C with the fewest edges, the weight of the updated matching M' can be less or greater than $\Phi(M) - \gamma(C)$; recall (3.1) and (3.15)–(3.17) for the definitions of Φ and γ. This means, in particular, that the straightforward cycle-canceling procedure, i.e., iteratively updating a matching with a negative cycle with the fewest edges, may not terminate in a finite number of iterations. To guarantee the finite termination, we introduce the following heuristics:

> Repeat the following until there exists no negative cycles. Let C be a negative cycle with the fewest edges and set $M' := (M \Delta C) \cap E$. If $\Phi(M') > \Phi(M)$, then update M to M' and reconstruct the network. Otherwise, modify γ by setting $\gamma(e) := 0$ for all $e \in C$.

Clearly this procedure terminates in a finite number of iterations if all valu-

ations $w_1, \ldots, w_m$ are integer-valued. Note that, in a graph-theoretic view, the operation "$\gamma(e) := 0$ for all $e \in C$" corresponds to the *contraction* of C. The proposed algorithm is shown in Algorithm 3.

We analyze the complexity of the algorithm. For simplicity, we assume that all w_i are nonnegative integer-valued. Let $W := \max_M \Phi(M)$ with the maximum taken over all matchings M. In each iteration of the outer for-loop (line 2 to line 21) for a fixed k, the inner loop (line 4 to line 16) is executed at most $O(|E(G(M))|W) = O(n^2mW)$ times, because either the number of negative edges is decreased (by line 14) or the objective value is increased (by line 11). Line 5 takes $O(n|V(G(M))||E(G(M))|) = O(n^4m^2)$ time to perform the Bellman–Ford algorithm for each left vertex since every cycle of $G(M)$ contains at least one left vertex. Therefore the inner loops take $O(n^6m^3W)$ time in total, and hence the algorithm takes $O(n^7m^3W)$ time. This estimated complexity appears very expensive, but in our experiments, the algorithm has turned out to be sufficiently efficient for practical problems.

Algorithm 3 Proposed algorithm for submodular welfare problem

```
 1: Let M = ∅ be an initial solution
 2: for k = 1, ..., n + 1 do
 3:   G ← G(M) in (3.18)
 4:   loop
 5:     Find a negative cycle C with the fewest edges in G
 6:     if there are no negative cycles then
 7:       break
 8:     end if
 9:     Let M' := (MΔC) ∩ E
10:     if Φ(M') > Φ(M) then
11:       M ← M'
12:       G ← G(M) in (3.18)
13:     else
14:       Set γ(e) ← 0 for all e ∈ C in G
15:     end if
16:   end loop
17:   if k < n then
18:     Find a shortest path P of G that starts from the source-vertex s
        and terminates at the sink-vertex t
19:     Update M ← (MΔP) ∩ E
20:   end if
21: end for
```

As for the approximation ratio, we can establish only a partial theoretical bound. The contraction operation (line 14) is a heuristic operation that is necessary to guarantee the finite termination, but it prevents us from

establishing an approximation ratio of the algorithm. Under the assumption that this contraction does not occur at the last stage of the algorithm, i.e., the network G at the termination of the algorithm is identical with the auxiliary network $G(M)$ defined in (3.18), we can prove the approximation bound with factor 2.

Theorem 4.1. Suppose that all valuations w_i are nonnegative monotone nondecreasing submodular functions. If $G = G(M)$ at the end of Algorithm 3, the obtained solution $(X_1, \ldots, X_m)$ satisfies

$$2\Phi(X_1, \ldots, X_m) \geq \Phi(X_1^*, \ldots, X_m^*), \tag{4.1}$$

where $(X_1^*, \ldots, X_m^*)$ is an optimal solution of the submodular welfare problem (1.2). ■

For the proof of Theorem 4.1, we can make use of a result of Fisher, Nemhauser, and Wolsey (Theorem 5.1 in [17]). To be self-contained, we here state an adaption of their result[1] together with a proof.

Lemma 4.2. Suppose that all valuations w_i are nonnegative monotone nondecreasing submodular functions. If a partition $(X_1, \ldots, X_m)$ of V satisfies

$$\Phi(X_1, \ldots, X_m) \geq \Phi(X_1, \ldots, X_i \setminus \{u\}, \ldots, X_j \cup \{u\}, \ldots, X_m) \tag{4.2}$$

for all $u \in X_i$ and $j \neq i$, we have

$$2\Phi(X_1, \ldots, X_m) \geq \Phi(X_1^*, \ldots, X_m^*). \tag{4.3}$$

where $(X_1^*, \ldots, X_m^*)$ is an optimal solution of the submodular welfare problem (1.2).

Proof. Denote by X the subset of $V^{(1)} \cup \cdots \cup V^{(m)}$ that corresponds to $(X_1, \ldots, X_m)$, and similarly for X^*. Since Φ is monotone submodular, we have the following inequality (e.g., Proposition 2.4 in [38]):

$$\Phi(X^*) \leq \Phi(X) + \sum_{v^{(i)} \in X^* \setminus X} \Big(\Phi(X \cup \{v^{(i)}\}) - \Phi(X) \Big).$$

Since X represents a partition of V, for each $v^{(i)} \in X^* \setminus X$, there exists $v^{(j)} \in X \setminus X^*$ for some $j \neq i$. By submodularity of Φ and (4.2),

$$\begin{aligned} \Phi(X \cup v^{(i)}) - \Phi(X) &\leq \Phi(X \cup \{v^{(i)}\} \setminus \{v^{(j)}\}) - \Phi(X \setminus \{v^{(j)}\}) \\ &\leq \Phi(X) - \Phi(X \setminus \{v^{(j)}\}). \end{aligned}$$

[1] Fisher, Nemhauser, and Wolsey [17] deals with independence systems in general.

Therefore, together with monotonicity, we obtain

$$\begin{aligned}\Phi(X^*) &\leq \Phi(X) + \sum_{x \in X \setminus X^*} (\Phi(X) - \Phi(X \setminus \{x\})) \\ &\leq \Phi(X) + \sum_{x \in X} (\Phi(X) - \Phi(X \setminus \{x\})) . \end{aligned} \tag{4.4}$$

For the summation in (4.4), by letting $X = \{x_1, \ldots, x_{|X|}\}$, we have

$$\begin{aligned}&(\Phi(X) - \Phi(X \setminus \{x_1\})) + (\Phi(X) - \Phi(X \setminus \{x_2\})) + \cdots \\ \leq &(\Phi(X) - \Phi(X \setminus \{x_1\})) + (\Phi(X \setminus \{x_1\}) - \Phi(X \setminus \{x_1, x_2\})) + \cdots \\ = &\Phi(X) - \Phi(\emptyset) \leq \Phi(X).\end{aligned}$$

Substituting this estimate into (4.4), we obtain (4.3). □

Proof of Theorem 4.1. Suppose that $G = G(M)$ at the end of the algorithm. When the algorithm is terminated in line 7, the network G has no negative cycles. Since $G(M) = G$, by assumption, $G(M)$ has no negative cycles. By the construction of the graph, for each $u \in X_i$ and $j \neq i$, there exists a cycle C that

$$\Phi(X_1, \ldots, X_m) - \Phi(X_1, \ldots, X_i \setminus \{u\}, \ldots, X_j \cup \{u\}, \ldots, X_m) = \gamma(C), \tag{4.5}$$

where $\gamma(C) \geq 0$ since there are no negative cycles. Therefore we can apply Lemma 4.2 to prove the theorem. □

Remark 4.3. The condition $G = G(M)$ in Theorem 4.1 is satisfied if line 14 is not executed after the last substitution of $G \leftarrow G(M)$. ■

Remark 4.4. Lehmann, Lehmann, and Nisan [27] proved that a simple greedy algorithm attains 2-approximation ratio for general submodular utilities. Our result is not as good as theirs because we need the assumption of $G = G(M)$, but in practice this assumption is very often satisfied and our algorithm outperforms their algorithm. See Subsection 5.2. ■

Remark 4.5. The proposed algorithm (Algorithm 3) can be applied for non-submodular (i.e., general) utilities. In such case, of course, we cannot prove any approximation bound. ■

Remark 4.6. The proposed algorithm (Algorithm 3) can be regarded as a kind of *ejection chain method* [24] in meta-heuristics, which generalizes alternating path type algorithms of some graph algorithms. Ejection chain method combines simple neighborhoods (e.g., swapping two elements) to build more general neighborhoods (e.g., swapping $2k$ elements simultaneously). Depending on the problem, there are many ways to construct ejection chains and some methods detect negative cycles in the state space. See [2, 24] for more details.

5 Computational results

5.1 Set k-cover problem

Abrams, Goel, and Plotkin [1] considered the following problem, the set k-cover problem[2], for wireless sensor monitoring.

> We are given n sensors and l locations. Each sensor x covers a subset of locations S_x. The problem is to find a k-partition of sensors $X_1, \dots, X_k$ that maximizes the total number of locations covered, i.e.,
>
> $$\begin{array}{ll} \text{maximize} & w(X_1) + \cdots + w(X_k) \\ \text{subject to} & X_1, \dots, X_k\text{: partition of } \{1, \dots, n\} \\ \text{where} & w(X) = |\bigcup_{x \in X} S_x|. \end{array}$$

Since the coverage function

$$w(X) = \left| \bigcup_{x \in X} S_x \right| \tag{5.1}$$

is submodular, this problem can be regarded as a submodular welfare problem (1.2). Set k-cover problem is NP-hard, and $(15/16 + \epsilon)$-approximation is NP-complete [1].

Here we remark that the coverage function (5.1) can be represented as a sum of matroid rank functions. This follows from an alternative expression

$$w(X) = \sum_{j=1}^{l} r_j(X)$$

with matroid rank functions

$$r_j(X) := \begin{cases} 1, & j \in \bigcup_{x \in X} S_x, \\ 0, & \text{otherwise.} \end{cases}$$

A function that can be represented as a sum of matroid rank functions is called *matroid rank sum function*. Matroid rank sum functions are regarded as a class of submodular functions with some useful properties (see [41]).

A matroid rank function is a matroid valuation (Example 2.1), but a matroid rank sum function is not necessarily a matroid valuation, and maximization of a matroid rank sum function is NP-hard since it contains the set k-cover problem. In spite of this, we hope that matroid rank sum functions are often "close" to matroid valuations, so that our algorithm performs fairly well for instances in this class.

Here we ran experiments for two types of datasets, following [12].

[2] "Set k-cover problem" is originally considered by Slijepcevic and Potkonjak [43], which is slightly different from this problem.

- Data set "RANDOM": Fix the number of sensors n and the number of locations l. For each location j, an integer d_j is chosen uniformly in a prescribed integer interval $[d_{\min}, d_{\max}]$. Each of the d_j sensors covering j is then randomly selected. We use $n = 20$, $l = 50$, and $[d_{\min}, d_{\max}] = [3, 5]$, and the experiments were run with $k = 5$.

- Data set "GEOMETRIC": Fix the number of sensors n. Put n sensors and l' locations as random points in the unit square. A sensor covers all locations within distance r. Locations that are covered by less than $d_{\min}(= 4)$ sensors are deleted (hence the number l of locations is less than or equal to l'). We use $n = 20$, $l' = 50$, and $r = 0.4$, and the experiments were run with $k = 5$.

We compare our algorithm with the simple randomized algorithm proposed by Abrams, Goel, and Plotkin [1] (AGP for short). Their algorithm assigns each location to a randomly chosen sensor. The algorithm is a $(1 - 1/e)$-approximation algorithm (in expectation). We ran this algorithm 100 times and took the best.

Since the instances are not so large, we can compute the exact solution of these instances via integer programming. The outputs of the AGP and our algorithms are compared with the exact solutions computed by ILOG CPLEX 11.200.

The results are shown in Tables 5.1 and 5.2. The ratios to the optimal values are attached in the parentheses. The "Cycle" column shows the number of detected negative cycles (i.e., the number of executions of line 9 of Algorithm 3) and the number of the contractions (i.e., the number of executions of line 14 of Algorithm 3) in the format of "(number of contractions) / (number of detected negative cycles)". These values, which are equal to zero for the valuated matroid partition problem, can be regarded as a measure of "closeness" to the valuated matroid partition problem. In the column of "Guarantee" we put T if $G = G(M)$ at the termination of the algorithm and F otherwise. Recall that this condition is the assumption for the approximation guarantee (Theorem 4.1).

For all instances, the proposed algorithm outperforms the AGP algorithm and has a 2-approximation guarantee (i.e., $G = G(M)$ at the end of termination). The proposed algorithm finds only a small number of negative cycles for all the instances except RND.3. This means these instances are "close" to the valuated matroid partition problem.

5.2 Budgeted allocation problem

The budgeted allocation problem, proposed by Garg, Kumar, and Pandit [23], also known as budgeted constrained auction problem, is the following problem.

Table 5.1: Set k-cover problem (RANDOM instances)

Problem	Optimal	AGP (ratio)	Proposed (ratio)	Cycle	Guarantee
RND.0	175	146 (0.834)	171 (0.977)	0/0	T
RND.1	182	150 (0.824)	178 (0.978)	0/3	T
RND.2	178	149 (0.837)	176 (0.989)	0/0	T
RND.3	178	150 (0.842)	175 (0.983)	94/99	T
RND.4	176	146 (0.829)	172 (0.977)	0/0	T
RND.5	176	148 (0.840)	170 (0.966)	0/1	T
RND.6	175	143 (0.817)	171 (0.977)	0/3	T
RND.7	180	153 (0.850)	173 (0.961)	0/0	T
RND.8	184	148 (0.804)	178 (0.967)	0/1	T
RND.9	179	148 (0.826)	176 (0.983)	0/0	T
Average		(0.830)	(0.975)		

"Cycle": (number of contractions) / (number of detected negative cycles)
"Guarantee": T if $G = G(M)$, and F otherwise.

Consider m bidders and n items V. Each bidder i is willing to pay an amount $b_{ix} > 0$ for item $x \in V$ and has a budget B_i at which his total payment is capped. The problem is to assign each item to at most one bidder in order to maximize the total payment obtained:

$$\begin{array}{ll} \text{maximize} & w_1(X_1) + \cdots + w_m(X_m) \\ \text{subject to} & X_1, \ldots, X_m\text{: subpartition of } V, \\ \text{where} & w_i(X) = \min\{\sum_{x \in X} b_{ix}, B_i\}, \quad i = 1, \ldots, m. \end{array}$$

The budgeted allocation problem is NP-hard (in fact, approximating to a factor better than 15/16 is NP-hard [5]). There is a 3/4-approximation algorithm based on linear programming relaxation [44].

We compare our algorithm with the simple greedy algorithm for 1/2-approximation by Lehmann, Lehmann, and Nisan [27] (LLN for short). We generated the following instances and ran the experiment with them.

(1) R102570 instances are generated with $m = 10$, $n = 25$, $B_i = 70$, and $b_{ix} \in [10, 30]$ uniformly random.

(2) R204060 instances are generated with $m = 20$, $n = 40$, $B_i = 60$, and $b_{ix} \in [20, 40]$ uniformly random.

Since the instances are not so large, we can compute the exact solution of these instances via integer programming. The outputs of the LLN and

Table 5.2: Set k-cover problem (GEOMETRIC instances)

Problem	Optimal	AGP (ratio)	Proposed (ratio)	Cycle	Guarantee
GEO.0	198	175 (0.883)	193 (0.975)	0/1	T
GEO.1	192	178 (0.927)	187 (0.974)	0/3	T
GEO.2	223	191 (0.856)	216 (0.969)	0/5	T
GEO.3	218	193 (0.885)	213 (0.977)	0/7	T
GEO.4	213	188 (0.882)	208 (0.977)	0/5	T
GEO.5	207	185 (0.893)	203 (0.981)	0/2	T
GEO.6	224	191 (0.852)	219 (0.978)	0/9	T
GEO.7	210	171 (0.814)	202 (0.962)	0/4	T
GEO.8	206	184 (0.893)	201 (0.976)	0/3	T
GEO.9	181	156 (0.861)	175 (0.966)	0/2	T
Average		(0.874)	(0.973)		

"Cycle": (number of contractions) / (number of detected negative cycles)
"Guarantee": T if $G = G(M)$, and F otherwise.

our algorithms are compared with the exact solutions computed by ILOG CPLEX 11.200. The results show that our algorithm performs very well, better than LLN for all instances. In particular, R204060 instances can be solved exactly by the proposed algorithm without resorting to the negative cycle canceling heuristics. This means that these instances are very close to the valuated matroid partition problem.

5.3 Generalized assignment problem

Generalized assignment problem (GAP) is a problem similar to the budgeted allocation problem, and is described as follows:

> There are m bidders and n items V. Each bidder i is willing to pay an amount $b_{ix} > 0$ for item $x \in V$ and has a budget B_i at which his total payment is capped. When a bidder i gets item x, he gets profit c_{ix}. The problem is to maximize the total profit of bidders by distributing n items with the budget constraint:

$$\begin{array}{ll} \text{maximize} & w_1(X_1) + \cdots + w_m(X_m) \\ \text{subject to} & X_1, \ldots, X_m\text{: partition of } V, \\ \text{where} & w_i(X) = \begin{cases} \sum_{x \in X} c_{ix} & \text{if } \sum_{x \in X} b_{ix} \leq B_i, \\ -\infty & \text{otherwise.} \end{cases} \end{array}$$

The above problem, called Max-GAP in particular, is a submodular welfare problem since w_i is a submodular function. If bidder i's pricing b_{ix} of x does

Table 5.3: Budgeted allocation problem (R102570 instances)

Problem	Optimal	LLN (ratio)	Proposed (ratio)	Cycle	Guarantee
R102570.0	646	633 (0.979)	643 (0.995)	0/5	T
R102570.1	649	635 (0.978)	646 (0.995)	0/1	T
R102570.2	650	640 (0.984)	650 (1.000)	0/2	T
R102570.3	639	625 (0.978)	636 (0.995)	0/3	T
R102570.4	648	631 (0.973)	648 (1.000)	0/2	T
R102570.5	649	639 (0.984)	649 (1.000)	0/2	T
R102570.6	648	635 (0.979)	648 (1.000)	0/2	T
R102570.7	646	636 (0.984)	645 (0.998)	0/1	T
R102570.8	649	644 (0.992)	649 (1.000)	0/1	T
R102570.9	643	631 (0.981)	642 (0.998)	0/1	T
Average		(0.981)	(0.998)		

"Cycle": (number of contractions) / (number of detected negative cycles)
"Guarantee": T if $G = G(M)$, and F otherwise.

Table 5.4: Budgeted allocation problem (R204060 instances)

Problem	Optimal	LLN (ratio)	Proposed (ratio)	Cycle	Guarantee
R204060.0	1159	1135 (0.979)	1159 (1.000)	0/0	T
R204060.1	1174	1119 (0.953)	1174 (1.000)	0/0	T
R204060.2	1173	1152 (0.982)	1173 (1.000)	0/0	T
R204060.3	1165	1122 (0.963)	1165 (1.000)	0/0	T
R204060.4	1167	1136 (0.973)	1167 (1.000)	0/0	T
R204060.5	1166	1118 (0.958)	1166 (1.000)	0/0	T
R204060.6	1176	1131 (0.961)	1176 (1.000)	0/0	T
R204060.7	1172	1129 (0.963)	1172 (1.000)	0/0	T
R204060.8	1158	1106 (0.955)	1158 (1.000)	0/0	T
R204060.9	1165	1117 (0.958)	1165 (1.000)	0/0	T
Average		(0.964)	(1.000)		

"Cycle": (number of contractions) / (number of detected negative cycles)
"Guarantee": T if $G = G(M)$, and F otherwise.

not depend on the item x, i.e., $b_{ix} = b_{iy}$ for all $x, y \in V$, his utility function w_i is a matroid valuation. Hence, if each bidder's pricing of items does not vary so much across items, his utility function w_i is "close" to a matroid valuation.

For consistency with the existing literature it is convenient to consider the minimization variant of GAP, called Min-GAP:

$$\begin{array}{ll} \text{minimize} & w_i(X_1) + \cdots + w_m(X_m) \\ \text{subject to} & X_1, \ldots, X_m\text{: partition of } V, \\ \text{where} & w_i(X) = \begin{cases} \sum_{x \in X} c_{ix} & \text{if } \sum_{x \in X} b_{ix} \leq B_i, \\ +\infty & \text{otherwise.} \end{cases} \end{array} \tag{5.2}$$

Obviously, we can transform Max-GAP and Min-GAP to each other by changing the sign of w_i $(i = 1, \ldots, m)$. We emphasize that, in Min-GAP, the objective functions w_i are supermodular.

The difficulty of Min-GAP comes from the feasibility constraint. To relax the feasibility, we here use the penalty function method. Let β be a large number and consider the following problem:

$$\begin{array}{ll} \text{minimize} & w_{i\beta}(X_1) + \cdots + w_{m\beta}(X_m) \\ \text{subject to} & X_1, \ldots, X_m\text{: partition of } V, \\ \text{where} & w_{i\beta}(X) = \sum_{x \in X} c_{ix} + \beta \max\{0, \sum_{x \in X} b_{ix} - B_i\}. \end{array} \tag{5.3}$$

If β is sufficiently large, the problem (5.3) is equivalent to the problem (5.2). Note that the function $w_{i\beta}$ is supermodular if $\beta \geq 0$.

We here use the datasets from ORLIB [6][3] to evaluate our algorithm. There are four types of instances, called type a, type b, type c, and type d. See [8] for detailed description of the instances. For example, the instance "a10200" is of type a with $m = 10$ and $n = 200$. Since these problems are Min-GAP, the ratios (Proposed)/(Best known) are greater than or equal to 1. The results are shown in Tables 5.5 and 5.6, where $*$-marked values are known to be optimal.

In the literature, instances of types a and b are considered relatively easy and instances of types c and d are considered relatively hard. Our algorithm finds many negative cycles for instances of types c and d.

5.4 Max cut problem

Let $G = (V, E)$ be an undirected graph with (possibly negative) edge weight $l : E \to \mathbb{R}$. An m-partition $(X_1, \ldots, X_m)$ of V is called m-cut, and the cut value of this m-partition is defined as the sum of the edge weights connecting different parts:

$$l(X_1, \ldots, X_m) := \sum_{u \in X_i, v \in X_j, i \neq j} l(u, v).$$

[3] `http://people.brunel.ac.uk/~mastjjb/jeb/orlib/gapinfo.html` [8]

Table 5.5: Generalized assignment problem Min-GAP (instances of types a and b)

Problem	Best known	Proposed (ratio)	Cycle	Guarantee
a05100	1698*	1698 (1.000)	0/0	T
a05200	3235*	3235 (1.000)	0/0	T
a10100	1360*	1360 (1.000)	0/0	T
a10200	2623*	2623 (1.000)	0/0	T
a20100	1158*	1158 (1.000)	0/0	T
a20200	2339*	2341 (1.001)	0/0	T
b05100	1843	1855 (1.006)	113/230	T
b05200	3552	3565 (1.003)	4968/5197	T
b10100	1407	1413 (1.004)	0/7	T
b10200	2828	2848 (1.007)	1764/1918	T
b20100	1166	1168 (1.002)	0/3	T
b20200	2340	2343 (1.001)	0/9	T

"Cycle": (number of contractions) / (number of detected negative cycles)
"Guarantee": T if $G = G(M)$, and F otherwise
$*$: known to be optimal.

Then the max (multi-)cut problem is the following problem:

$$\begin{array}{ll} \text{maximize} & l(X_1, \ldots, X_m) \\ \text{subject to} & X_1, \ldots, X_m\text{: partition of } V. \end{array}$$

This problem can be written as a general partition problem as follows. Let

$$w(X) := -\sum_{u,v \in X} l(u, v). \tag{5.4}$$

Then we have

$$l(X_1, \ldots, X_m) = w(X_1) + \cdots + w(X_m) - w(V). \tag{5.5}$$

Therefore the max cut problem is equivalent to the following:

$$\begin{array}{ll} \text{maximize} & w(X_1) + \cdots + w(X_m) \\ \text{subject to} & X_1, \ldots, X_m\text{: partition of } V. \end{array}$$

Proposition 5.1. If $l(e)$ is nonnegative for all $e \in E$, then the function w in (5.4) is submodular. If $l(e)$ is nonpositive for all $e \in E$, then w is supermodular. ■

In the context of combinatorial auction, the utility function of the form

$$w(X) = \sum_{u \in X} a(u) + \sum_{u,v \in X} b(u, v)$$

Table 5.6: Generalized assignment problem Min-GAP (instances of types c and d)

Problem	Best known	Proposed (ratio)	Cycle	Guarantee
c05100	1931*	1969 (1.019)	63/167	T
c05200	3456*	3482 (1.007)	4261/4509	T
c10100	1402*	1415 (1.009)	531/567	T
c10200	2806*	2820 (1.004)	4863/5011	T
c20100	1243*	1257 (1.011)	247/226	T
c20200	2391	2402 (1.004)	1241/1311	F
d05100	6353*	6729 (1.059)	9/264	T
d05200	12743	13272 (1.041)	2759/3337	T
d10100	6349	6662 (1.049)	683/849	F
d10200	12436	13058 (1.050)	2131/2631	T
d20100	6196	6528 (1.053)	645/699	T
d20200	12264	12974 (1.057)	2167/2473	T

"Cycle": (number of contractions) / (number of detected negative cycles)
"Guarantee": T if $G = G(M)$, and F otherwise
$*$: known to be optimal.

is called quadratic function [42] (or 2-wise dependent function [11], or 2-additive function [10]). Such w is supermodular if b is nonnegative, and submodular if b is nonpositive. Shioura and Suzuki [42] discussed computational complexity of the welfare problem with quadratic utilities.

We use the instances from Biq Mac Library[4]. Since these instances are for the maximum (two-)cut problem, we have $m = 2$ for these instances. We compare our algorithm with the optimal solution also collected in Biq Mac Library.

5.4.1 Random-weighted instances

We first examine the random-weighted instances. There are two types of random instances in Biq Mac Library:

- w instances: a random graph of $n = 100$ vertices with edge density 0.1, 0.5, or 0.9 whose edges have weights $[-10, 10]$.
- pw instances (p stands for "positive"): a random graph of $n = 100$ vertices with edge density 0.1, 0.5, or 0.9 whose edges have weights $[0, 10]$.

[4] `http://biqmac.uni-klu.ac.at/biqmaclib.html` [4].

For example, the instance named "w01_100.3" is the 3rd of the w instances with $p = 0.1$ and $n = 100$. The results are shown in Tables 5.7 and 5.8. The ratios denote (Proposed)/(Optimal). The results show that the positive instances are closer to the valuated matroid partition problem.

5.4.2 Instances from statistical physics

We next examine instances used in statistical physics (spin glasses). There are two types of instances available at Biq Mac Library[5]:

- ising instances: one-dimensional Ising chain. For example, the instance named "ising2.5-100_5555" is for the chain of length $n = 100$ with model parameter 2.5 with random seed 5555.
- t2g or t3g instances: two- or three-dimensional toroidal grid. For example, the instances named "t2g10_5555" is for two-dimensional toroidal grid of size $n = 10 \times 10$ with random seed 5555.

For a detailed description of these instances, see [4, 28].

[5] `http://biqmac.uni-klu.ac.at/biqmaclib.html` [4].

Table 5.7: Maximum cut (random instances)

Problem	Optimal	Proposed (ratio)	Cycle	Guarantee
w01_100.0	651	624 (0.958)	107/204	T
w01_100.1	719	679 (0.944)	194/270	T
w01_100.2	676	646 (0.955)	95/186	T
w01_100.3	813	771 (0.948)	59/140	T
w01_100.4	668	598 (0.895)	74/160	T
w01_100.5	643	553 (0.860)	130/219	T
w01_100.6	654	584 (0.892)	68/161	T
w01_100.7	725	684 (0.943)	122/210	T
w01_100.8	721	690 (0.957)	100/203	T
w01_100.9	729	707 (0.969)	145/246	T
Average		(0.932)		

Problem	Optimal	Proposed (ratio)	Cycle	Guarantee
w05_100.0	1646	1538 (0.934)	337/455	T
w05_100.1	1606	1580 (0.983)	481/601	T
w05_100.2	1902	1778 (0.934)	505/650	T
w05_100.3	1627	1505 (0.925)	629/745	T
w05_100.4	1546	1451 (0.938)	673/804	T
w05_100.5	1581	1456 (0.920)	449/533	T
w05_100.6	1479	1380 (0.933)	610/712	T
w05_100.7	1987	1907 (0.959)	703/806	T
w05_100.8	1311	1219 (0.929)	870/975	T
w05_100.9	1752	1740 (0.993)	533/654	T
Average		(0.944)		

Problem	Optimal	Proposed (ratio)	Cycle	Guarantee
w09_100.0	2121	2040 (0.961)	698/798	T
w09_100.1	2096	1944 (0.927)	774/900	T
w09_100.2	2738	2558 (0.934)	537/644	T
w09_100.3	1990	1983 (0.996)	703/837	T
w09_100.4	2033	1970 (0.969)	817/932	T
w09_100.5	2433	2433 (1.000)	925/1053	T
w09_100.6	2220	1998 (0.900)	991/1143	T
w09_100.7	2252	2238 (0.993)	807/921	T
w09_100.8	1843	1796 (0.974)	813/963	T
w09_100.9	2043	1876 (0.918)	865/998	T
Average		(0.957)		

"Cycle": (number of contractions) / (number of detected negative cycles)
"Guarantee": T if $G = G(M)$, and F otherwise.

Table 5.8: Maximum cut (random positive instances)

Problem	Optimal	Proposed (ratio)	Cycle	Guarantee
pw01_100.0	2019	1987 (0.984)	0/45	T
pw01_100.1	2060	2027 (0.983)	234/261	T
pw01_100.2	2032	2012 (0.990)	0/31	T
pw01_100.3	2067	2052 (0.992)	0/49	T
pw01_100.4	2039	1982 (0.972)	2/32	T
pw01_100.5	2108	2068 (0.981)	59/97	T
pw01_100.6	2032	2017 (0.992)	0/36	T
pw01_100.7	2074	2056 (0.991)	42/78	T
pw01_100.8	2022	1989 (0.983)	0/28	T
pw01_100.9	2005	1981 (0.988)	143/165	T
Average		(0.985)		

Problem	Optimal	Proposed (ratio)	Cycle	Guarantee
pw05_100.0	8190	8143 (0.994)	0/73	T
pw05_100.1	8045	7924 (0.984)	186/260	T
pw05_100.2	8039	7963 (0.990)	652/705	T
pw05_100.3	8139	8034 (0.987)	474/541	T
pw05_100.4	8125	8056 (0.991)	90/133	T
pw05_100.5	8169	8116 (0.993)	177/256	T
pw05_100.6	8217	8193 (0.997)	388/447	T
pw05_100.7	8249	8190 (0.992)	0/51	T
pw05_100.8	8199	8074 (0.984)	0/67	T
pw05_100.9	8099	8049 (0.993)	369/287	T
Average		(0.990)		

Problem	Optimal	Proposed (ratio)	Cycle	Guarantee
pw09_100.0	13585	13462 (0.990)	0/42	T
pw09_100.1	13417	13370 (0.996)	0/77	T
pw09_100.2	13461	13408 (0.996)	0/71	T
pw09_100.3	13656	13548 (0.992)	0/70	T
pw09_100.4	13514	13514 (1.000)	234/293	T
pw09_100.5	13574	13547 (0.998)	2/63	T
pw09_100.6	13640	13533 (0.992)	0/59	T
pw09_100.7	13501	13452 (0.996)	117/174	T
pw09_100.8	13593	13553 (0.997)	87/124	T
pw09_100.9	13658	13587 (0.994)	82/157	T
Average		(0.995)		

"Cycle": (number of contractions) / (number of detected negative cycles)
"Guarantee": T if $G = G(M)$, and F otherwise.

Table 5.9: Maximum cut (ising instances)

Problem	Optimal	Proposed (ratio)	Cycle	Guarantee
ising2.5-100_5555	2460049	2197899 (0.893)	268/378	T
ising2.5-100_6666	2031217	1851543 (0.911)	355/458	T
ising2.5-100_7777	3363230	3274129 (0.973)	423/537	T
ising2.5-150_5555	4363532	4044852 (0.926)	1023/1202	T
ising2.5-150_6666	4057153	3661955 (0.902)	1271/1261	T
ising2.5-150_7777	4243269	4101118 (0.966)	812/1021	T
ising2.5-200_5555	6294701	5971157 (0.948)	801/1024	T
ising2.5-200_6666	6795365	6544015 (0.963)	1098/1346	T
ising2.5-200_7777	5568272	5253044 (0.943)	733/986	T
ising2.5-250_5555	7919449	7326828 (0.925)	637/968	T
ising2.5-250_6666	6925717	6447957 (0.931)	775/1048	T
ising2.5-250_7777	6596797	6104229 (0.925)	1392/1719	T
ising2.5-300_5555	8579363	8010655 (0.933)	618/1002	T
ising2.5-300_6666	9102033	8374765 (0.920)	1337/1790	T
ising2.5-300_7777	8323804	7925367 (0.952)	1128/1590	T
ising3.0-100_5555	2448189	2236489 (0.913)	323/430	T
ising3.0-100_6666	1984099	1829937 (0.922)	559/663	T
ising3.0-100_7777	3335814	3142714 (0.942)	424/524	T
ising3.0-150_5555	4279261	3963090 (0.926)	400/589	T
ising3.0-150_6666	3949317	3627200 (0.918)	682/848	T
ising3.0-150_7777	4211158	4085580 (0.970)	551/736	T
ising3.0-200_5555	6215531	5879563 (0.945)	570/808	T
ising3.0-200_6666	6756263	6544388 (0.968)	773/1037	T
ising3.0-200_7777	5560824	5244404 (0.943)	153/364	T
ising3.0-250_5555	7823791	7247703 (0.926)	825/1146	T
ising3.0-250_6666	6903351	6299186 (0.912)	548/862	T
ising3.0-250_7777	6418276	6099705 (0.950)	1122/1434	T
ising3.0-300_5555	8493173	7987702 (0.940)	1458/1865	T
ising3.0-300_6666	8915110	8185811 (0.918)	1019/1472	T
ising3.0-300_7777	8242904	7947532 (0.964)	722/1120	T

"Cycle": (number of contractions) / (number of detected negative cycles)
"Guarantee": T if $G = G(M)$, and F otherwise.

Table 5.10: Maximum cut (t2g/t3g instances)

Problem	Optimal	Proposed (ratio)	Cycle	Guarantee
t2g10_5555	6049461	5695406 (0.941)	16/125	T
t2g10_6666	5757868	5244491 (0.910)	31/104	T
t2g10_7777	6509837	5998835 (0.921)	72/172	T
t2g15_5555	15051133	13703423 (0.910)	49/289	T
t2g15_6666	15763716	13761173 (0.872)	27/282	T
t2g15_7777	15269399	14801682 (0.969)	49/289	T
t2g20_5555	24838942	23227409 (0.935)	77/542	T
t2g20_6666	29290570	27514389 (0.939)	80/593	T
t2g20_7777	28349398	25318886 (0.893)	129/647	T
t3g5_5555	10933215	9690353 (0.886)	50/186	T
t3g5_6666	11582216	10813582 (0.933)	75/194	T
t3g5_7777	11552046	10871750 (0.941)	85/216	T
t3g6_5555	17434469	16232171 (0.931)	184/430	T
t3g6_6666	20217380	19190413 (0.949)	123/358	T
t3g6_7777	19475011	18181794 (0.933)	102/363	T
t3g7_5555	28302918	26220536 (0.926)	176/635	T
t3g7_6666	33611981	31625244 (0.940)	40/410	T
t3g7_7777	29118445	27196173 (0.933)	225/650	T

"Cycle": (number of contractions) / (number of detected negative cycles)
"Guarantee": T if $G = G(M)$, and F otherwise.

6 Conclusion

We have proposed a heuristic algorithm for the submodular welfare problem. The basic idea is to apply the valuated matroid partition algorithm to the submodular welfare problem with some appropriate modifications. The resulting algorithm has a theoretical guarantee of approximation ratio under a certain assumption, which is very often satisfied in practice. The algorithm performs quite well for a number of practical problems used in our computational experiments.

Acknowledgement

This work is supported by KAKENHI (21360045) and the Aihara Project, the FIRST program from JSPS.

References

[1] Z. Abrams, A. Goel, and S. Plotkin (2004): Set k-cover algorithms for energy efficient monitoring in wireless sensor networks. Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks, pp. 424–432.

[2] R. K. Ahuja, Ö. Ergun, J. B. Orlin, and A. P. Punnen (2002): A survey of very large-scale neighborhood search techniques. Discrete Applied Mathematics, vol. 123, pp. 75–102.

[3] N. Andelman, and Y. Mansour (2004): Auctions with budget constraints. Proceedings of the 9th Scandinavian Workshop on Algorithm Theory (SWAT), pp. 26–38.

[4] W. Angelika (2007): Biq Mac Library–A collection of Max-Cut and quadratic 0–1 programming instances of medium size. `http://biqmac.uni-klu.ac.at/biqmaclib.pdf`.

[5] Y. Azar, B. Birnbaum, and A. Karlin (2008): Improved approximation algorithms for budgeted allocations. Automata Language and Programming, pp. 186–197.

[6] J. E. Beasley (1990): ORLIB: Operations Research Library. `http://people.brunel.ac.uk/~mastjjb/jeb/info.html` (accessed: November 25, 2013).

[7] C. Beviá, M. Quinzii, and J. A. Silva (1999): Buying several indivisible goods. Mathematical Social Sciences, vol. 37, pp. 1–23.

[8] P. C. Chu and J. E. Beasley (1997): A genetic algorithm for the generalised assignment problem. Computers & Operations Research, vol. 24, pp. 17–23.

[9] D. Chakrabarty and G. Goel (2010): On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and GAP. SIAM Journal on Computing, vol. 39, pp. 2189–2211.

[10] Y. Chevaleyre, U. Endriss, S. Estivie, and N. Maudet (2008): Multiagent resource allocation in k-additive domains: preference representation and complexity. Annals of Operations Research, vol. 163, pp. 49–62.

[11] V. Conitzer, T. Sandholm, and P. Santi (2005): Combinatorial auctions with k-wise dependent valuations. Proceedings of the 20th National Conference on Artificial Intelligence, pp. 248–254.

[12] A. Deshpande and S. Khuller (2008): Energy efficient monitoring in sensor networks. LATIN' 08 Proceedings of the 8th Latin American Conference on Theoretical informatics, pp. 436–448.

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

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

[15] J. Edmonds, J (1970): Submodular functions, matroids and certain polyhedra. In: R. Guy, H. Hanani, N. Sauer, and J. Schönheim, eds., Combinatorial Structures and Their Applications, Gordon and Breach, New York, pp. 69–87. Also in: M. Jünger, G. Reinelt, and G. Rinaldi, eds., Combinatorial Optimization—Eureka, You Shrink! Springer, Berlin (2003), pp. 11–26.

[16] U. Feige and J. Vondrák (2010): The submodular welfare problem with demand queries. Theory of Computing, vol. 6, pp. 247–290.

[17] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey (1978): An analysis of approximations for maximizing submodular set functions-II. Mathematical Programming Study, vol. 8, pp. 73–87.

[18] T. Fleiner (2003): A fixed-point approach to stable matchings and some applications. Mathematics of Operations Research, vol. 28, pp. 103–126.

[19] L. Fleischer, M. X. Goemans, V. S. Mirrokni, and M. Sviridenko (2006): Tight approximation algorithms for maximum general assignment problems. Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithm, pp. 611–620.

[20] A. M. Frieze and M. Jerrum (1995): Improved approximation algorithms for MAX k-CUT and MAX BISECTION. Proceedings of the 4th International IPCO Conference, Springer, Berlin, pp. 1–13.

[21] S. Fujishige (2005): Submodular Functions and Optimization. 2nd ed., Annals of Discrete Mathematics, vol. 58, Elsevier, Amsterdam.

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

[23] R. Garg, V. Kumar, and V. Pandit (2001): Approximation algorithms for budget-constrained auctions. APPROX'01/RANDOM'01, Proceedings of the 4th International Workshop on Approximation Algorithms

for Combinatorial Optimization Problems and 5th International Workshop on Randomization and Approximation Techniques in Computer Science: Approximation, Random, pp. 102–113.

[24] F. Glover (1996): Ejection chains, reference structures and alternating path methods for traveling salesman problems. Discrete Applied Mathematics, vol .65, pp. 223–253.

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

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

[27] B. Lehmann, D. Lehmann, and N. Nisan (2006): Combinatorial auctions with decreasing marginal utilities. Games and Economic Behavior, vol. 55, pp. 270–296.

[28] F. Liers, M. Jünger, G. Reinelt, and G. Rinaldi (2004): Computing exact ground states of hard ising spin glass problems by branch-and-cut. In: A. Hartmann and H. Rieger, eds., New Optimization Algorithms in Physics, Wiley, Weinheim, pp. 47–68.

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

[30] V. Mirrokni, M. Schapira, and J. Vondrák (2008): Tight information-theoretic lower bounds for welfare maximization in combinatorial auction. ACM Conference on Electronic Commerce 2008, pp. 70–77.

[31] K. Murota (1996): Valuated matroid intersection, I: optimality criteria. SIAM Journal on Discrete Mathematics, vol. 9, pp. 545–561.

[32] K. Murota (1996): Valuated matroid intersection, II: algorithms. SIAM Journal on Discrete Mathematics, vol. 9, pp. 562–576.

[33] K. Murota (1997): Matroid valuation on independent sets. Journal of Combinatorial Theory, Series B, vol. 69, pp. 59–78.

[34] K. Murota (1998): Discrete convex analysis. Mathematical Programming, vol. 83, pp. 313–371.

[35] K. Murota (2000): Matrices and Matroids for Systems Analysis. Springer, Berlin.

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

[37] K. Murota (2009): Recent developments in discrete convex analysis. In: W. Cook., L. Lovász, J. Vygen, eds., Research Trends in Combinatorial Optimization, Springer, Berlin, Chapter 11, pp. 219–260.

[38] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher (1978): An analysis of approximations for maximizing submodular set functions-I. Mathematical Programming, vol. 14, pp. 265–294.

[39] J. G. Oxley (1992): Matroid Theory. Oxford University Press, Oxford.

[40] F. Rendl, G. Rinaldi, and A. Wiegele (2007): A branch and bound algorithm for Max-Cut based on combining semidefinite and polyhedral relaxations. Proceedings of the 14th International IPCO Conference, Springer, Berlin, pp. 295–309.

[41] A. Shioura (2012): Matroid rank functions and discrete concavity. Japan Journal of Industrial and Applied Mathematics, vol. 29, pp. 535–546.

[42] A. Shioura and S. Suzuki (2012): Optimal allocation problem with quadratic utility functions and its relationship with graph cut. Journal of Operations Research Society of Japan, vol. 55, pp. 92–105.

[43] S. Slijepcevic and M. Potkonjak (2001): Power efficient organization of wireless sensor networks. Proceeding of IEEE International Conference on Communications 2001, Helsinki, pp. 472–476.

[44] A. Srinivasan (2008): Budgeted allocations in the full-information setting. In: M. Goemans, K. Jansen, J. D. P. Rolim, L. Trevisan, eds., Approximation, Randomization and Combinatorial Optimization: Algorithms and Techniques, Springer, Berlin, pp. 247–253.

[45] J. Vondrák (2008): Optimal approximation for the submodular welfare problem in the value oracle model. Proceedings of the 40th ACM Symposium on Theory of Computing, pp. 67–74.