Journal of the Operations Research Society of Japan
Vol. , No. , pp.

OPTIMAL ALLOCATION PROBLEM WITH QUADRATIC UTILITY FUNCTIONS AND ITS RELATIONSHIP WITH GRAPH CUT PROBLEM

Akiyoshi Shioura Shunya Suzuki
Tohoku University

(Received September 30, 2011; Revised December 27, 2011)

Abstract We discuss the optimal allocation problem in combinatorial auctions, where the items are allocated to bidders so that the sum of the bidders' utilities is maximized. In this paper, we consider the case where utility functions are given by quadratic functions; the class of such utility functions has a succinct representation but is sufficiently general. The main aim of this paper is to show the computational complexity of the optimal allocation problem with quadratic utility functions. We consider the cases where utility functions are submodular and supermodular, and show NP-hardness and/or polynomial-time exact/approximation algorithms. These results are given by using the relationship with graph cut problems such as the min/max cut problem and the multiway cut problem.

Keywords: Combinatorial optimization, allocation problem, submodular function, supermodular function, combinatorial auction, graph cut

1. Introduction

In this paper, we consider the optimal allocation problem arising from combinatorial auctions. A combinatorial auction is an auction such that bidders can place bids on combinations of items, rather than individual items. Combinatorial auctions can be used, for instance, to sell spectrum licenses, pollution permits, land lots, etc., and has emerged as a mechanism to improve economic efficiency when many items are on sale; see [2, 3] for comprehensive survey on combinatorial auctions.

In a combinatorial auction, bidders can present bids on bundles of items, and thus may easily express relationship among the items on sale such as substitutability and complementarity. Two items are said to be substitutes if both items have similar roles and a bidder does not obtain so much gain with having both items, compared to having each item separately; typical examples are margarine and butter, and tea and coffee. On the other hand, two items are said to be complements if neither of the two items are not useful without the other, and a bidder obtain more gain with having both items than having each item separately; typical examples are lock and key, and pencil and eraser. The function that, given a bundle, returns the bidder's value for that bundle is called a utility function. A utility function is associated with each bidder specifying the degree of satisfaction of the bidder for each subset of the items.

Given utility functions of bidders, the auctioneer of a combinatorial auction needs to solve the problem of "optimally" allocating items to bidders, which we call the *optimal allocation problem*. One natural objective for the auctioneer is to maximize the economic efficiency of the auction, which is the sum of the utilities of all the bidders. Formally, the optimal allocation problem is defined as follows. Let V be a set of n items, and M a set

of m bidders, and assume, for simplicity, that $V = \{1, 2, \ldots, n\}$ and $M = \{1, 2, \ldots, m\}$. We denote by $\mathbb{R}_+$ the set of nonnegative real numbers. Bidder i has a utility function $f_i : 2^V \to \mathbb{R}_+$ which is *monotone*, i.e., $f_i(X) \geq f_i(Y)$ whenever $X \supseteq Y$. Then, the optimal allocation problem is formulated as follows:

$$\text{Maximize} \sum_{i=1}^{m} f_i(S_i) \quad \text{subject to} \quad (S_1, S_2, \ldots, S_m) \text{ is a partition of } V.$$

Recall that $(S_1, S_2, \ldots, S_m)$ is called a partition of V if $S_1, S_2, \ldots, S_m$ are subsets of V which are mutually disjoint and satisfy $\cup_{i=1}^{m} S_i = V$.

In the literature of combinatorial auctions, it is often assumed that bidders exhibit substitutability. This assumption is widely used in theory, and often justified in practice; examples are auctions with labors, housing, cars, etc. Substitutability of items is often modeled by using submodularity of utility functions in combinatorial auction; a utility function $f : 2^V \to \mathbb{R}_+$ is said to be *submodular* if it satisfies

$$f(X \cup \{v\}) - f(X) \geq f(Y \cup \{v\}) - f(Y) \\ (\forall X, Y \in 2^V \text{ with } Y \supset X, \forall v \in V \setminus Y).$$

A utility function with submodularity is also called a decreasing marginal utility.

On the other hand, combinatorial auctions with complementary items often appear in theory and in practice. A very special case is the one with "single-minded" bidders who are interested in only in a specified bundle of items, and not in any proper subset of the bundle. Complementarity of items is often modeled by using supermodularity of utility functions in combinatorial auction; a utility function $f : 2^V \to \mathbb{R}_+$ is said to be *supermodular* if it satisfies

$$f(X \cup \{v\}) - f(X) \leq f(Y \cup \{v\}) - f(Y) \\ (\forall X, Y \in 2^V \text{ with } Y \supset X, \forall v \in V \setminus Y).$$

A utility function of a single bidder who is interested in a bundle S of items can be represented as follows by using a positive value α:

$$f(X) = \begin{cases} \alpha & (\text{if } X \supseteq S), \\ 0 & (\text{otherwise}), \end{cases}$$

which is a supermodular function.

Implementation of combinatorial auctions faces several issues to be discussed, including representation of utility functions. A utility function for a bidder requires a value for each subset of items, and therefore requires an exponential number of real values in total. This makes it difficult for bidders to reveal their preference correctly since in practice it is not possible for bidders to submit *correct* values of utilities for a exponential number of subsets of items. This also brings a difficulty to the auctioneer since the input size of utility functions becomes exponential, and the optimal allocation becomes hard to solve efficiently.

Thus, we need a restricted class of utility functions which has a succinct representation but is sufficiently general. Representation of utility functions is called a *bidding language*, and various bidding languages have been considered in the literature of combinatorial auction (see, e.g., [2], [3, Ch. 9]). Some examples are symmetric functions, (budgeted) additive functions, single-minded functions [23], OR functions, XOR functions, and OR-of-XOR functions [27].

In this paper, we consider one such class of utility functions called *quadratic* functions. In the context of combinatorial auction, the use of quadratic functions is firstly considered independently by Conitzer et al. [6] (as *2-wise dependent* functions) and by Chevaleyre et al. [5] (as *2-additive* functions). A utility function $f : 2^V \to \mathbb{R}_+$ with $f(\emptyset) = 0$ is said to be *quadratic* (or *of order 2*) if it is represented as

$$f(X) = \sum_{u,v \in X, u<v} a(u,v) + \sum_{v \in X} b(v) \qquad (X \subseteq V) \tag{1.1}$$

by using real values $a(u,v)$ $(u,v \in V, u < v)$ and $b(v)$ $(v \in V)$ (see [9, Section 3.6]). Note that every quadratic utility function of the form (1.1) has a natural one-to-one correspondence with a quadratic polynomial function with $\{0,1\}$-variables of the following form:

$$\varphi_f(x) = \sum_{u,v \in V, u<v} a(u,v)x(u)x(v) + \sum_{v \in V} b(v)x(v) \qquad (x \in \{0,1\}^V).$$

While a quadratic utility function is simple and can be represented in a succinct way, it is sufficiently general so that by using the term $a(u,v)$ it can easily express substitutability and complementarity among items (see Section 2). These facts indicate that quadratic utility functions constitute an important class of utility functions.

The main aim of this paper is to reveal the computational complexity of the optimal allocation problem with quadratic utility functions. That is, we consider the case where a utility function $f_i : 2^V \to \mathbb{R}_+$ of bidder $i \in M$ is given as

$$f_i(X) = \sum_{u,v \in X, u<v} a_i(u,v) + \sum_{v \in X} b_i(v) \qquad (X \subseteq V) \tag{1.2}$$

by using real values $a_i(u,v)$ $(u,v \in V, u < v)$ and $b_i(v)$ $(v \in V)$. The same problem is considered in [5, 6], where they only show the NP-hardness in the case of general quadratic utility functions. In contrast, we classify the optimal allocation problem according to the type of utility functions (substitutes or complements) and the number of bidders (2 or more), analyze the computational complexity of each case, and present exact/approximation algorithms.

1.1. Previous results

We review the computational complexity results of the optimal allocation problem with *general* utility functions. We here consider only the case where a utility function f is given implicitly by a *value oracle*, which, given a set $S \subseteq V$, returns a function value $f(S)$. It is noted that the value oracle can be easily constructed for quadratic utility functions.

We firstly consider the case of submodular utility functions. The problem is NP-hard, even if $m = 2$. Moreover, there exists no polynomial-time approximation algorithm with a ratio better than $1 - 1/e$, unless P=NP [18]. Mirrokni et al. [25] also show that an approximation algorithm with a ratio better than $1-(1-1/m)^m$ requires exponentially many calls to the value oracle, implying, without any assumption, that there exists no polynomial-time approximation algorithm with a ratio better than $1 - 1/e$. For the class of gross-substitutes utility functions, which is known to be an important subclass of submodular utility functions [12, 16], the optimal allocation problem can be solved in polynomial time [22].

We then consider the case of supermodular utility functions. Supermodularity of utility functions is often used to model complementarity of items. Compared to the case of submodular utility functions, this case attracts less attention in the literature of combinatorial

Table 1: Summary of Our Results

type of utility function	# of bidders $m = 2$	# of bidders $m \geq 3$
submodular	NP-hard 0.874-approximation	NP-hard
gross substitutes	P ($O(n^2 \log n)$ time)	P ($O(mn^2 \log(mn))$ time)
supermodular	P ($O(n^3/\log n)$ time)	NP-hard 0.5-approximation (2/3-approximation for $m = 3$)

auction, and much is not known yet for this case. If $m = 2$, then the optimal allocation problem can be easily reduced to the submodular function minimization problem, which can be solved in polynomial time [11]. On the other hand, if $m \geq 3$ then the problem is NP-hard (see, e.g., [6]). While an $O(\sqrt{\log n}/n)$-approximation algorithm is given [15], no inapproximability result is known.

1.2. Our results

We analyze the computational complexity of the optimal allocation problem with quadratic utility functions. We consider the two important cases where utility functions are *submodular* and *supermodular*, and for each case we also consider subcases where the number m of bidders are equal to 2 and more than 2. That is, we consider 4 cases, each of which is denoted as (SUB|m=2), (SUB|m>2), (SUP|m=2), and (SUP|m>2). The results obtained in this paper are summarized in Table 1. These results are shown by using the relationship with graph cut problems such as the min/max cut problem and the multiway (un)cut problem.

For the case of submodular quadratic utility functions, we show the NP-hardness even in the case (SUB|m=2) by using the reduction of the max cut problem in *undirected* graphs. On the other hand, we present the reduction of the case (SUB|m=2) to the max cut problem in *directed* graphs. This reduction yields a 0.874-approximation algorithm for (SUB|m=2), which is better than the approximation ratio $1 - 1/e \simeq 0.632$ for the case of general submodular utility functions.

We also consider the special case of gross-substitutes quadratic utility functions as an important subclass of submodular utility functions. As mentioned in Section 1.1, this case can be solved in polynomial time, even for general utility functions which are not necessarily quadratic. It is shown that this case can be reduced to the minimum quadratic-cost flow problem and solved in $O(mn^2 \log(mn))$ time, which means that this case can be solved much faster by a relatively simple algorithm than the general case.

Then, the case of supermodular quadratic utility functions is considered. As shown in Section 1.1, it is known that (SUP|m=2) can be solved in polynomial time. We show in this paper that (SUP|m=2) can be reduced to the min cut problem in directed graphs, which implies that (SUP|m=2) can be solved in $O(n^3/\log n)$ time. We then show the NP-hardness of (SUP|m>2) by using the reduction of the multiway (un)cut problem. For this problem, we also present a 0.5-approximation algorithm based on randomized LP rounding, where we use the technique in Langberg et al. [21] for the multiway uncut problem.

The organization of this paper is as follows. Characterizations of submodular/supermodular quadratic utility functions are given in Section 2. In Section 3, we present our results for (SUB|m=2) and (SUB|m>2), while the results for (SUP|m=2), and (SUP|m>2) are given in Section 4.

2. Characterizations of Quadratic Utility Functions

We give characterizations of quadratic utility functions of the form (1.1) which have submodularity and supermodularity. In this section, we consider utility functions which may take negative values, although utility functions are originally assumed to be nonnegative-valued functions. Throughout this paper we assume $a(v,u) = a(u,v)$ for every $u, v \in V$ with $u < v$.

A utility function $f : 2^V \to \mathbb{R}$ is said to be *submodular* if it satisfies the following condition:

$$f(X \cup \{v\}) - f(X) \geq f(Y \cup \{v\}) - f(Y) \\ (\forall X, Y \in 2^V \text{ with } Y \supset X, \forall v \in V \setminus Y). \tag{2.1}$$

This condition is known to be equivalent to the following condition:

$$f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y) \qquad (\forall X, Y \in 2^V).$$

Intuitively, the condition (2.1) says that the marginal value of an item decreases as the set of items already acquired increases. A utility function $f : 2^V \to \mathbb{R}$ is said to be *supermodular* if $-f$ is submodular. Submodularity (resp., supermodularity) of utility functions is often used to model substitutability (resp., complementarity) of items in the context of combinatorial auction. A utility function $f : 2^V \to \mathbb{R}$ is said to be *monotone* if it satisfies $f(X) \leq f(Y)$ for every $X, Y \in 2^V$ with $X \subseteq Y$.

Theorem 2.1. *Let $f : 2^V \to \mathbb{R}$ be a quadratic utility function of the form* (1.1).
(i) *f is submodular if and only if $a(u,v) \leq 0$ $(\forall u, v \in V, u \neq v)$.*
(ii) *A submodular function f is monotone if and only if $b(v) + \sum_{u \in V \setminus \{v\}} a(u,v) \geq 0$ $(\forall v \in V)$.*

Proof. It is well known that the condition (2.1) is equivalent to the following:

$$f(X \cup \{u\}) + f(X \cup \{v\}) \geq f(X \cup \{u,v\}) + f(X) \quad (\forall X \subseteq V,\ \forall u, v \in V \setminus X,\ u \neq v).$$

Since

$$\{f(X \cup \{u,v\}) + f(X)\} - \{f(X \cup \{u\}) + f(X \cup \{v\})\} = a(u,v),$$

the inequality above is equivalent to the condition that $a(u,v) \leq 0$ for every distinct $u, v \in V$.

We then suppose that f is submodular. The condition (2.1) implies that f is monotone if and only if $f(V) - f(V \setminus \{v\}) \geq 0$ for all $v \in V$, which is in turn equivalent to the condition $b(v) + \sum_{u \in V \setminus \{v\}} a(u,v) \geq 0$ $(\forall v \in V)$ since

$$f(V) - f(V \setminus \{v\}) = b(v) + \sum_{u \in V \setminus \{v\}} a(u,v).$$

□

Theorem 2.2. *Let $f : 2^V \to \mathbb{R}$ be a quadratic utility function of the form* (1.1).
(i) *f is supermodular if and only if $a(u,v) \geq 0$ $(\forall u, v \in V,\ u \neq v)$.*
(ii) *A supermodular function f is monotone if and only if $b(v) \geq 0$ $(\forall v \in V)$.*

Proof. The statement (i) follows immediately from Theorem 2.1 (i) since f is supermodular if and only if $-f$ is submodular.

Suppose that f is supermodular. Then, f is monotone if and only if $0 \leq f(\{v\}) - f(\emptyset) = b(v)$ for all $v \in V$. □

We also consider an important subclass of submodular utility functions, called utility functions with gross substitutes condition [12, 16]. The *gross substitutes condition* of a utility function $f : 2^V \to \mathbb{R}$ is described as follows:

$\forall p, q \in \mathbb{R}^V$ with $p \leq q$, $\forall X \in \arg\max_{S \subseteq V}\{f(S) - p(S)\}$,
$\exists Y \in \arg\max_{S \subseteq V}\{f(S) - q(S)\}$ s.t. $X \cap \{v \in V \mid p(v) = q(v)\} \subseteq Y$,

where p and q are price vectors. We here denote $x(S) = \sum_{v \in S} x(v)$ for a vector $x \in \mathbb{R}^V$ and a subset $S \subseteq V$. Intuitively, the gross substitutes condition says that a bidder still wants to get items that do not change in price after the prices on other items increase.

Theorem 2.3 (cf. [14]). *A quadratic utility function $f : 2^V \to \mathbb{R}$ of the form (1.1) satisfies the gross substitutes condition if and only if the following condition holds for all distinct $u, v, t \in V$:*

$$a(u, v) \leq \max\{a(u, t), a(v, t)\} \leq 0. \tag{2.2}$$

We note that the condition (2.2) implies, in particular, that for every distinct $u, v, t \in V$, at least two values in $\{a(u, v), a(u, t), a(v, t)\}$ achieve the maximum among the three values.

In the proof of Theorem 2.3, we use the following characterization of gross-substitutes utility functions.

Theorem 2.4 ([26]; see also [3, Theorem 13.5]). *A utility function $f : 2^V \to \mathbb{R}$ satisfies the gross-substitutes condition if and only if f is submodular and satisfies the following condition for all $X \in 2^V$ and all distinct $u, v, t \in V$:*

$$\begin{aligned} & f(X \cup \{u, v\}) + f(X \cup \{t\}) \\ & \quad \leq \max\{f(X \cup \{u, t\}) + f(X \cup \{v\}), f(X \cup \{v, t\}) + f(X \cup \{u\})\}. \end{aligned} \tag{2.3}$$

Proof of Theorem 2.3. For every $X \subseteq V$ and distinct $u, v, t \in V \setminus X$, it holds that

$$\begin{aligned} & \{f(X \cup \{u, v\}) - f(X)\} + \{f(X \cup \{t\}) - f(X)\} \\ & \quad = a(u, v) + \sum_{s \in X} a(s, u) + \sum_{s \in X} a(s, v) + \sum_{s \in X} a(s, t) + \{b(u) + b(v) + b(t)\}. \end{aligned}$$

Therefore, the inequality (2.3) in Theorem 2.4 is equivalent to

$$a(u, v) \leq \max\{a(u, t), a(v, t)\}.$$

Hence, the statement of Theorem 2.3 follows from this fact and Theorem 2.1 (i). □

3. Results for Submodular Utility Functions

3.1. Hardness

We show the hardness of the problem (SUB|m=2) by the reduction of the max cut problem in undirected graphs. The max cut problem is a famous NP-hard problem; moreover, it is NP-hard to compute a solution with approximation ratio better than 0.879, under the assumption of the unique games conjecture [17].

As an instance of the max cut problem, let us consider an undirected graph $G = (V, E)$ with edge weight $w(u, v) \geq 0$ ($(u, v) \in E$). We define an instance of (SUB|m=2) by regarding V as the item set and by using quadratic utility functions such that

$$a_i(u, v) = \begin{cases} -w(u, v) & ((u, v) \in E), \\ 0 & (\text{otherwise}), \end{cases} \qquad b_i(v) = \sum_{(u,v) \in E, u \in V \setminus \{v\}} w(u, v)$$

for $i = 1, 2$. The definitions of a_i and b_i imply that the resulting quadratic utility functions f_1 and f_2 are monotone and submodular by Theorem 2.1. Moreover, for every partition (V_1, V_2) of V, the objective function value $f_1(V_1) + f_2(V_2)$ is equal to

$$-\sum_{i=1}^{2} \sum_{\substack{(u,v)\in E \\ u,v\in V_i}} w(u,v) + 2 \sum_{(u,v)\in E} w(u,v) \;=\; \sum_{\substack{(u,v)\in E \\ u\in V_1, v\in V_2}} w(u,v) + \sum_{(u,v)\in E} w(u,v),$$

i.e., the total weight of cut edges in G plus a constant value. Hence, the max cut problem on undirected graphs is reduced to (SUB|m=2), although this reduction does not preserve approximation ratio.

To obtain a reduction preserving approximation ratio, we need to use *non-monotone* utility functions f_1 and f_2 by changing the definition of $b_i(v)$ as follows:

$$b_i(v) = \frac{1}{2} \sum_{(u,v)\in E, u\in V\setminus\{v\}} w(u,v) \quad (v \in V),$$

which is the half of the original value. The resulting functions f_1 and f_2 are still submodular and take nonnegative values, but are non-monotone. With this change, we obtain the formula

$$f_1(V_1) + f_2(V_2) = \sum_{\substack{(u,v)\in E \\ u\in V_1, v\in V_2}} w(u,v).$$

This shows that the max cut problem on undirected graphs is reduced to (SUB|m=2) with non-monotone utility functions, which preserves approximation ratio.

Theorem 3.1. *The problems* (SUB|m=2) *and* (SUB|m>2) *are NP-hard. Moreover, for both problems with non-monotone utility functions, it is NP-hard to compute a solution with approximation ratio better than 0.879, under the assumption of the unique games conjecture.*

3.2. Approximability

We present an approximability result for the problem (SUB|m=2) by showing the reduction to the max s-t cut problem in directed graphs.

Given an instance of (SUB|m=2), we define a directed graph $G = (V \cup \{s,t\}, E)$ by

$$E = \{(u,v) \mid u,v \in V, u < v\} \cup \{(s,u) \mid u \in V\} \cup \{(v,t) \mid v \in V\}.$$

For each edge $(u,v) \in E$, its weight $w(u,v)$ is defined as follows:

$$w(s,u) = b_2(u) + \sum_{v\in V, v>u} a_2(u,v), \quad w(v,t) = b_1(v) + \sum_{u\in V, u<v} a_1(u,v) \quad (v \in V),$$
$$w(v,u) = -a_1(u,v) - a_2(u,v) \quad (u,v \in V, u < v).$$

Theorem 2.1 (i) implies $w(u,v) \geq 0$, while (ii) implies $w(s,u) \geq 0$ and $w(v,t) \geq 0$. Hence, all of edge weights are nonnegative.

Let (S,T) be an s-t cut, i.e., a partition of the vertex set $V \cup \{s,t\}$ satisfying $s \in S, t \in T$. Then, the weight of the cut (S,T) is equal to

$$\begin{aligned}
&\sum_{v\in S\cap V} w(v,t) + \sum_{u\in T\cap V} w(s,u) + \sum_{\substack{v\in S\cap V \\ u\in T\cap V, u<v}} w(v,u) \\
&= \sum_{v\in S\cap V} b_1(v) + \sum_{u,v\in S\cap V, u<v} a_1(u,v) + \sum_{v\in T\cap V} b_2(v) + \sum_{u,v\in T\cap V, u<v} a_2(u,v) \\
&= f_1(S\cap V) + f_2(T\cap V),
\end{aligned}$$

where we use the fact that $S \cap V = V \setminus T$ and $T \cap V = V \setminus S$. Hence, (SUB|m=2) is reduced to the max s-t cut problem in G, and this reduction preserves approximation ratio. It is shown by Lewin et al. [24] that a 0.874-approximate solution of the max s-t cut problem can be computed in polynomial time. Therefore, we obtain the following result:

Theorem 3.2. *A 0.874-approximate solution of the problem* (SUB|m=2) *can be computed in polynomial time.*

3.3. Fast exact algorithm for special case

We consider a special case where utility functions satisfy the gross substitutes condition, and show that the optimal allocation problem in this case can be reduced to the minimum quadratic-cost flow problem. The reduction is based on the following property of gross-substitutes quadratic utility functions. A set family $\mathcal{F} \subseteq 2^V$ is said to be *laminar* if it satisfies $X \subseteq Y$, $X \supseteq Y$, or $X \cap Y = \emptyset$ holds for every $X, Y \in \mathcal{F}$.

Lemma 3.3 (cf. [14]). *A quadratic utility function $f : 2^V \to \mathbb{R}$ of the form* (1.1) *satisfies the gross substitutes condition if and only if it is represented as*

$$f(X) = -\sum_{S \in \mathcal{F}} c_S |X \cap S|^2 \tag{3.1}$$

by using a laminar family $\mathcal{F} \subseteq 2^V$ and real numbers c_S ($S \in \mathcal{F}$) satisfying $\{v\} \in \mathcal{F}$ ($v \in V$) and $c_S \geq 0$ ($S \in \mathcal{F}, |S| \geq 2$). Moreover, given a gross-substitutes quadratic utility function f of the form (1.1)*, we can compute the representation* (3.1) *in* $\mathrm{O}(n^2)$ *time.*

This lemma implies that the function value of a gross-substitutes quadratic utility function can be represented as the (quadratic) flow cost on a tree network.

We now explain the reduction to the minimum quadratic-cost flow problem. Suppose that a utility function f_i of bidder i is of the form $f_i(X) = -\sum_{S \in \mathcal{F}_i} c_S^i |X \cap S|^2$, where $\mathcal{F}_i \subseteq 2^V$ is a laminar family and c_S^i ($S \in \mathcal{F}_i$) are real numbers satisfying the conditions in Lemma 3.3. Note that such representations can be computed in $\mathrm{O}(mn^2)$ time by Lemma 3.3. We construct a graph $\hat{G} = (\hat{V}, \hat{E})$ as follows.

Define $\hat{V} = \{r\} \cup V \cup \bigcup_{i=1}^m V_i$, where V_i ($i \in M$) is given as $V_i = \{v_S^i \mid S \in \mathcal{F}_i\}$. Note that $v_{\{u\}}^i \in V_i$ for each $i \in M$ and $u \in V$. Vertices in V are source vertices, while r is the unique sink vertex.

We also define $\hat{E} = E_0 \cup \bigcup_{i=1}^m E_i$, where

$$E_0 = \{(u, v_{\{u\}}^i) \mid u \in V,\ i \in M\},$$
$$E_i = \{(v_X^i, r) \mid X \in \mathcal{F}, \text{maximal in } \mathcal{F}\} \cup \{(v_X^i, v_{\rho(X)}^i) \mid X \in \mathcal{F}_i, \text{not maximal in } \mathcal{F}_i\},$$

where for every non-maximal set $X \in \mathcal{F}_i$, we denote by $\rho(X)$ the unique minimal set $Y \in \mathcal{F}_i$ with $Y \supset X$. Note that edge set E_i for $i \in M$ constitutes a rooted tree with root r.

For each edge in E_0, its flow capacity is given by the interval $[0, 1]$, and its flow cost is 0. For each edge (v_X^i, r) or $(v_X^i, v_{\rho(X)}^i)$ in E_i, its flow capacity is $[0, +\infty]$, and its flow cost function is given by $c_X^i \varphi^2$, where φ is the flow value on the edge. We also define supply/demand values of source/sink vertices to be 1 for each $u \in V$ and $-n$ for r.

We consider the minimum (quadratic-)cost flow problem on the network $\hat{G}$ under the capacity constraint and the supply/demand constraint. It is not difficult to see that integral feasible flows on the network have one-to-one correspondence to partitions of the set V, and the cost of the flow is equal to the negative of the total utilities for the corresponding partition. Hence, we can obtain an optimal allocation by solving the minimum cost flow problem.

The minimum quadratic-cost flow problem can be solved by iteratively augmenting flows along a shortest path in the so-called "auxiliary network," and the number of iterations is n (see, e.g., [1]). Since the graph $\hat{G}$ has $O(mn)$ vertices and $O(mn)$ edges, the minimum cost flow problem can be solved in $O(mn\log(mn)) \times n = O(mn^2\log(mn))$ time by using the shortest-path algorithm of Fredman and Tarjan [8] as a subroutine.

Theorem 3.4. *The optimal allocation problem with gross-substitutes quadratic utility functions can be solved in* $O(mn^2\log(mn))$ *time.*

4. Results for Supermodular Utility Functions

4.1. Polynomial-time solvable case

We firstly show that the problem (SUP|m=2) can be reduced to the minimum s-t cut problem in a directed graph.

Lemma 4.1 ([13, Theorem 1], [20, Theorem 4.1])**.** *Given an instance of* (SUP|m=2)*, we can construct in* $O(n^2)$ *time an edge-weighted directed graph* $G = (V \cup \{s,t\}, E)$ *such that for every* $X \subseteq V$*, the cut value of* $(X \cup \{s\}, (V \setminus X) \cup \{t\})$ *is equal to* $f_1(X) + f_2(V \setminus X)$*.*

This lemma shows that (SUP|m=2) can be reduced to the minimum s-t cut problem in G. Note that the graph G has $O(n)$ vertices and $O(n^2)$ edges. Hence, the minimum s-t cut problem can be solved in $O(n^3/\log n)$ time by the algorithm of Cheriyan et al. [4].

Theorem 4.2. *The problem* (SUP|m=2) *can be solved in* $O(n^3/\log n)$ *time.*

4.2. Hardness

To show the NP-hardness of the problem (SUP|m>2), we show that the multiway (un)cut problem on undirected graphs [7, 21] can be reduced to (SUP|m>2).

Input of the *multiway (un)cut problem* is an undirected graph $G = (V, E)$ with distinct terminals $s_1, s_2, \ldots, s_k \in V$ $(k \geq 2)$ and edge weight $w(u,v) \geq 0$ $((u,v) \in E)$. In the multiway *cut* problem, we find a partition $(V_1, V_2, \ldots, V_k)$ of V with $s_i \in V_i$ $(i = 1, 2, \ldots, k)$ *minimizing* the total weight of cut edges given as

$$\sum\{w(u,v) \mid (u,v) \in E,\ u \in V_i,\ , v \in V_j,\ i \neq j\},$$

while in the multiway *uncut* problem, we want to *maximize* the total weight of *uncut* edges. The multiway (un)cut problem is known to be NP-hard, even when $k = 3$ [7].

Given an instance of the multiway (un)cut problem, we define an instance of (SUP|m>2) by regarding V as the item set and by

$$a^i(u,v) = \begin{cases} w(u,v) & ((u,v) \in E), \\ 0 & (\text{otherwise}), \end{cases} \qquad b^i(v) = \begin{cases} \Gamma & (v = s_i), \\ 0 & (\text{otherwise}), \end{cases}$$

where Γ is a sufficiently large positive number. Let $(V_1, V_2, \ldots, V_k)$ be a partition of V which is an optimal solution of this instance. Then, each V_i contains the vertex s_i since $b^i(s_i)$ is a sufficiently large number. Moreover, the objective function value is given as

$$\sum_{i=1}^{k} \sum_{(u,v)\in E, u,v\in V_i} w(u,v),$$

which we want to maximize. Hence, an optimal solution for (SUP|m>2) is an optimal solution for the multiway (un)cut problem, and vice versa.

Theorem 4.3. *The problem* (SUP|m>2) *is NP-hard, even when* $m = 3$*.*

4.3. Approximation algorithm by LP rounding

We propose a 0.5-approximation algorithm for the problem (SUP|m>2). Our algorithm is based on a natural linear programming (LP) relaxation:

$$\begin{aligned}
\text{Maximize} \quad & \sum_{i=1}^{m} \sum_{u,v \in V, u<v} a^i(u,v) y^i(u,v) + \sum_{i=1}^{m} \sum_{v \in V} b^i(v) x^i(v) \\
\text{subject to} \quad & \sum_{i=1}^{m} x^i(v) = 1 \quad (v \in V), \\
& y^i(u,v) \leq x^i(u),\ y^i(u,v) \leq x^i(v) \quad (u,v \in V, u<v), \\
& x^i(v) \geq 0\ (v \in V), \quad y^i(u,v) \geq 0\ (u,v \in V, u<v).
\end{aligned}$$

It should be noted that a pair of inequalities $y^i(u,v) \leq x^i(u),\ y^i(u,v) \leq x^i(v)$ can be replaced with an equation $y^i(u,v) = \min\{x^i(u), x^i(v)\}$ without loss of generality since $a^i(u,v) \geq 0$ holds.

The algorithm firstly computes an optimal solution of the LP relaxation. Then, the algorithm chooses a bidder $i \in \{1,2,\ldots,m\}$ and a value $\rho \in [0,1]$ uniformly at random, and assigns each item $v \in V$ to the bidder i if $x^i(v) \geq \rho$. The algorithm repeats this step until all items are assigned to one of the bidders. Although this algorithm is randomized, it can be derandomized by using the technique in Kleinberg and Tardos [19].

We analyze the performance of the algorithm. For $v \in V$ and $i \in M$, let $X^i(v) \in \{0,1\}$ be a random variable such that

$$X^i(v) = \begin{cases} 1 & \text{(if the item } v \text{ is assigned to the bidder } i\text{)}, \\ 0 & \text{(otherwise)}. \end{cases}$$

Similarly, for distinct $u, v \in V$ and $i \in M$, let $Y^i(u,v) \in \{0,1\}$ be a random variable such that

$$Y^i(u,v) = \begin{cases} 1 & \text{(if both of the items } u \text{ and } v \text{ are assigned to the bidder } i\text{)}, \\ 0 & \text{(otherwise)}. \end{cases}$$

We denote $y(u,v) = \sum_{i=1}^{m} y^i(u,v)$ for $u, v \in V$ with $u < v$.

Lemma 4.4 ([21, Fact 3.1]). *Let $v \in V$ and $i \in M$. Assume that item v is not assigned to any bidder before some iteration. Then, the probability that v is assigned to bidder i in the iteration is $(1/m)x^i(v)$.*

Lemma 4.5 ([21, Claim 3.2]). *For $v \in V$ and $i \in M$, we have $\Pr[X^i(v) = 1] = x^i(v)$.*

Lemma 4.6. *For distinct $u, v \in V$ and $i \in M$, we have*

$$\Pr[Y^i(u,v) = 1] \geq \frac{y^i(u,v)}{2 - y(u,v)}.$$

Proof. Let $\Psi \in [0,1]$ be the probability that both of u and v are assigned to the bidder i in the *same iteration*. Then, $\Pr[Y^i(u,v) = 1] \geq \Psi$ holds. In the following, we show the equation $\Psi = y^i(u,v)/\{2 - y(u,v)\}$.

The probability that u and v is assigned to a bidder $i \in M$ in some iteration is $(1/m)\min\{x^i(u), x^i(v)\} = (1/m)y^i(u,v)$. Similarly, the probability that at least one of u and v is assigned to any bidder in some iteration is

$$\begin{aligned}
\frac{1}{m}\sum_{i=1}^{m} \max\{x^i(u), x^i(v)\} &= \frac{1}{m}\left[\sum_{i=1}^{m}\{x^i(u) + x^i(v)\} - \sum_{i=1}^{m} \min\{x^i(u), x^i(v)\}\right] \\
&= \frac{1}{m}\{2 - y(u,v)\}.
\end{aligned}$$

Hence, the probability that u and v are assigned to a bidder $i \in M$ in the k-th iteration is

$$\left[1 - \frac{1}{m}\{2 - y(u,v)\}\right]^{k-1} \times \frac{1}{m} \cdot y^i(u,v).$$

Hence,

$$\begin{aligned} \Psi &= \sum_{k=1}^{\infty} \left[1 - \frac{1}{m}\{2 - y(u,v)\}\right]^{k-1} \times \frac{1}{m} \cdot y^i(u,v) \\ &= \frac{m}{2 - y(u,v)} \cdot \frac{1}{m} \cdot y^i(u,v) \;=\; \frac{y^i(u,v)}{2 - y(u,v)}. \end{aligned}$$

This implies the claim of the lemma. □

We calculate the expectation of the objective function value of an approximate solution obtained by the algorithm. By Lemmas 4.5 and 4.6, it holds that

$$\begin{aligned} &\sum_{i=1}^{m} \sum_{u,v \in V, u<v} a^i(u,v) \Pr[Y^i(u,v) = 1] + \sum_{i=1}^{m} \sum_{v \in V} b^i(v) \Pr[X^i(v) = 1] \\ &\geq \sum_{i=1}^{m} \sum_{u,v \in V, u<v} a^i(u,v) \cdot \frac{y^i(u,v)}{2 - y(u,v)} + \sum_{i=1}^{m} \sum_{v \in V} b^i(v) x^i(v) \\ &\geq 0.5 \cdot \mathrm{OPT}_{\mathrm{LP}}, \end{aligned} \tag{4.1}$$

where $\mathrm{OPT}_{\mathrm{LP}}$ denotes the optimal value of the LP relaxation. Since $\mathrm{OPT}_{\mathrm{LP}}$ is an upper bound of the optimal value of (SUP$|m{>}2$), we obtain the following result.

Theorem 4.7. *A 0.5-approximate solution of the problem* (SUP$|m{>}2$) *can be computed in polynomial time.*

With a more careful analysis, we can show that the approximation ratio is $1/(2-\varepsilon)$ (> 0.5), where $\varepsilon = \min\{y(u,v) \mid (u,v) \in E,\ y(u,v) > 0\}$; this bound is obtained by analyzing the cases $y(u,v) = 0$ and $y(u,v) > 0$ separately in the inequality (4.1).

Our analysis shows that the integrality gap of the LP relaxation is at least 0.5. On the other hand, an instance with integrality gap 2/3 can be easily constructed, as follows. Let us consider an instance of the problem with $V = \{a, b, c\}$, $m = 3$, and

$$\begin{aligned} &a^1(a,b) = 1,\ a^1(b,c) = 0,\ a^1(a,c) = 0, \\ &a^2(a,b) = 0,\ a^2(b,c) = 1,\ a^2(a,c) = 0, \\ &a^3(a,b) = 0,\ a^3(b,c) = 0,\ a^3(a,c) = 1, \\ &b^i(v) = 0\ (i = 1,2,3,\ v = a,b,c). \end{aligned}$$

For this instance, the optimal value of the original problem is 1, while the optimal value of the LP relaxation is 3/2, implying that the integrality gap is equal to 2/3. An open problem is to close the gap between 0.5 and 2/3. A possible approach for a better approximation algorithm is to construct a new LP formulation which has a larger value of $\min\{y(u,v) \mid (u,v) \in E,\ y(u,v) > 0\}$.

4.4. Other approximation algorithms

We then consider alternative approximation algorithms by using the fact that (SUP$|m{=}2$) can be solved in polynomial time.

If $m = 3$, then we can obtain a 2/3-approximate solution easily as follows. We compute an optimal allocation $(V_1^{(12)}, V_2^{(12)}, \emptyset)$ of items to bidders 1 and 2, where bidder 3 is ignored. In the same way, we compute optimal allocations $(V_1^{(13)}, \emptyset, V_3^{(13)})$ for bidders 1 and 3 and $(\emptyset, V_2^{(23)}, V_3^{(23)})$ for bidders 2 and 3. Then, we choose the best allocation among the three, which is a 2/3-approximate solution of the original problem.

Theorem 4.8. *A 2/3-approximate solution of the problem* (SUP$|m{>}2$) *with* $m = 3$ *can be computed in polynomial time.*

For the general case with $m \geq 3$, it is natural to consider the following heuristic based on local search. Given a partition $(V_1, V_2, \ldots, V_m)$ of V and bidders $i, j \in M$, we denote by $\texttt{realloc}(i, j)$ an operation which optimally re-allocates items in $V_i \cup V_j$ to bidders i and j. Our heuristic is as follows: start with an arbitrarily chosen initial partition, and repeatedly apply the operation $\texttt{realloc}(i, j)$ to arbitrarily chosen two bidders $i, j \in M$ until no improvement is possible by this operation.

Although our preliminary computational experiment shows that the local-search heuristic always outputs a near-optimal solution, we can construct a family of instances for which the approximation ratio can be arbitrarily close to 0. Let us consider an instance of the problem with $V = \{a, b, c\}$, $m = 3$, and

$$a^i(u, v) = \begin{cases} \varepsilon & (\text{if } i = 1, (u, v) = (a, b)), \\ 1 & (\text{if } i = 3, (u, v) = (b, c)), \\ 0 & (\text{otherwise}), \end{cases}$$

$$b^i(v) = \begin{cases} \varepsilon & (\text{if } i = 2, v = c), \\ 0 & (\text{otherwise}), \end{cases}$$

where ε is a sufficiently small positive number. Suppose that the initial partition is $(\{a, b, c\}, \emptyset, \emptyset)$. By applying the operation $\texttt{realloc}(1, 2)$, the partition becomes $(\{a, b\}, \{c\}, \emptyset)$. Then, the partition never changes even if we apply the operation $\texttt{realloc}(i, j)$ for any $i, j \in \{1, 2, 3\}$. The objective function value of the partition $(\{a, b\}, \{c\}, \emptyset)$ is 2ε, while the optimal value is 1, implying that the approximation ratio is 2ε.

Acknowledgement

An extended abstract of this paper is appeared in Proceedings of the 8th Annual Conference on Theory and Applications of Models of Computation (TAMC 2011), Lecture Notes in Computer Science (Vol. 6648), Springer 2011. This work is partially supported by a Grant-in-Aid for Scientific Research from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

[1] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin: *Network Flows: Theory, Algorithms, and Applications* (Prentice Hall, Upper Saddle River, NJ, 1993).

[2] 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, New York, NY, 2007), 267–299.

[3] P. Cramton, Y. Shoham, and R. Steinberg: *Combinatorial Auctions* (MIT Press, Boston, MA, 2006).

[4] J. Cheriyan, T. Hagerup, and K. Mehlhorn: An o(n)-time maximum-flow algorithm. *SIAM Journal on Computing*, **25** (1996), 1144–1170.

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

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

[7] E. Dahlhaus, D. S. Johnson, C.H. Papadimitriou, P.D. Seymour, and M. Yannakakis: The complexity of multiterminal cuts. *SIAM Journal on Computing*, **23** (1994), 864–894.

[8] M.L. Fredman and R.E. Tarjan: Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of ACM*, **34** (1987), 596–615.

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

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

[11] M. Grötschel, L. Lovász, and A. Schrijver: The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, **1** (1984), 169–197.

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

[13] P.L. Hammer: Some network flow problems solved with pseudo-Boolean programming. *Operations Research*, **13** (1965), 388–399.

[14] H. Hirai and K. Murota: M-convex functions and tree metrics. *Japan Journal of Industrial and Applied Mathematics*, **21** (2004), 391–403.

[15] R. Holzman, N. Kfir-Dahav, D. Monderer, and M. Tennenholtz: Bundling equilibrium in combinatorial auctions. *Games and Economic Behavior*, **47** (2004), 104–123.

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

[17] S. Khot, G. Kindler, E. Mossel, and R. O'Donnell: Optimal inapproximability results for max-cut and other 2-variable CSPs. *Proceedings of 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, (2004), 146–154.

[18] S. Khot, R.J. Lipton, E. Markakis, and A. Mehta: Inapproximability results for combinatorial auctions with submodular utility functions. *Algorithmica*, **52** (2008), 3–18.

[19] J. Kleinberg and É. Tardos: Approximation algorithms for classification problems with pairwise relationships: metric labeling and Markov random fields. *Journal of ACM*, **49** (2002), 616–639.

[20] V. Kolmogorov: What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26** (2004), 147–159.

[21] M. Langberg, Y. Rabani, and C. Swamy: Approximation algorithms for graph homomorphism problems. *Proceedings of 9th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems & 10th International Workshop on Randomization and Computation (APPROX & RANDOM)*, (2006), 176–187.

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

[23] D. Lehmann, L. O'Callaghan, and Y. Shoham: Truth revelation in approximately efficient combinatorial auctions. *Proceedings of 1st ACM Conference on Electronic Commerce (EC)*, (1999), 96–102.

[24] M. Lewin, D. Livnat, and U. Zwick: Improved rounding techniques for the MAX 2-SAT and MAX DI-CUT problems. *Proceedings of 9th Conference on Integer Programming and Combinatorial Optimization (IPCO)*, (2002), 67–82.

[25] V. Mirrokni, M. Schapira, and J. Vondrák: Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. *Proceedings of 7th ACM Conference on Electronic Commerce (EC)*, (2008), 70–77.

[26] H. Reijniese, A. van Gellekom, and J.A.M. Potters: Verifying gross substitutability. *Economic Theory*, **20** (2002), 767–776.

[27] T. Sandholm: Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, **135** (2002), 1–54.

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

Akiyoshi Shioura
Graduate School of Information Sciences
Tohoku University
Aramaki aza Aoba 6-3-09, Aoba-ku
Sendai 980-8579, Japan
E-mail: `shioura@dais.is.tohoku.ac.jp`