情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2011-AL-134 No.3
2011/3/7

２次の効用関数に関する組合せオークションにおける最適配分問題のアルゴリズム

塩 浦 昭 義[†1]　　鈴 木 瞬 也[†1]

本稿では組合せオークションにおける最適配分問題について議論する．この問題では，オークション参加者に財を配分し，参加者の効用の合計値を最大にすることを目的とする．本稿では，効用関数が 2 次関数により与えられる場合を考える．2 次の効用関数は，簡潔な表現をもつにもかかわらず，十分に一般的な効用関数のクラスを与える．本稿は，2 次の効用関数に関する最適配分問題の計算複雑度を明らかにすることを目的とする．とくに，効用関数が劣モジュラおよび優モジュラの場合について考え，NP 困難性および多項式時間の厳密（もしくは近似）アルゴリズムを示す．これらの結果は，最小（最大）カット問題や多方面カット問題などのグラフカット問題との関係を利用して示される．

Algorithms for Optimal Allocation Problem in Combinatorial Auction with Quadratic Utility Functions

Akiyoshi Shioura[†1] and Shunya Suzuki[†1]

We discuss the optimal allocation problem in combinatorial auction, 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 quadratic 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 algorithm. These results are given by using the relationship with graph cut problems such as the min/max cut problem and the multiway cut problem.

1. Introduction

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 substitutabilities and complementarities among the items on sale. 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 happiness of the bidder for each subset of the items.

Given utility functions of bidders, the auctioneer of a combinatorial auction needs to determine how to allocate items to bidders, which is called 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 $M = \{1, 2, \ldots, m\}$. Bidder i has a utility function $f_i : 2^V \to \mathbf{R}$ which is *monotone*, i.e., $f_i(X) \geq f_i(Y)$ whenever $X \supseteq Y$. The auctioneer wishes to find a partition $(S_1, S_2, \ldots, S_m)$ of the set V among the m bidders that maximizes the total utility $\sum_{i=1}^{m} f_i(S_i)$.

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 exponential 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

†1 東北大学
Tohoku University

Vol.2011-AL-134 No.3
2011/3/7

solve in short time.

Thus, we need a restricted class of utility functions which has a succinct representation but sufficiently general.⋆1 Various such classes of utility functions have been considered in the literature of combinatorial auction (see, e.g., 2) and Chapter 9 of 3)). Some examples are symmetric functions, (budgeted) additive functions, single-minded functions[21)], OR functions, XOR functions, and OR-of-XOR functions[25)].

In this paper, we consider the class of *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 $f : 2^V \to \mathbf{R}$ is said to be *quadratic* (or *of order 2*) if it is represented as

$$f(X) = \sum_{u,v \in X, u<v} a(u,v) + \sum_{v \in X} b(v) \qquad (X \subseteq V) \tag{1}$$

by using real values $a(u,v)$ $(u,v \in V, u < v)$ and $b(v)$ $(v \in V)$ (see Section 3.6 of 9)).⋆2 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 express substitutability and complementarity among items. This fact shows 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 $f_i : 2^V \to \mathbf{R}$ of bidder $i \in M$ is given as

$$f_i(X) = \sum_{u,v \in X, u<v} a_i(u,v) + \sum_{v \in X} b_i(v) \qquad (X \subseteq V) \tag{2}$$

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 of the very general case. 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 Representation of utility functions is called a *bidding language*.

⋆2 A utility function is quadratic if and only if it can be represented by a quadratic polynomial function with $\{0,1\}$-variables.

Table 1 Summary of Our Results (uf. = utility function)

type of uf. \ # of bidders m	$m = 2$	$m \geq 3$
submodular uf.	NP-hard, 0.879-inapprox. 0.874-approx.	NP-hard, 0.879-inapprox.
gross substitutes uf.	P ($O(n^2 \log n)$ time)	P ($O(mn^2 \log(mn))$ time)
supermodular uf.	P ($O(n^3/\log n)$ time)	NP-hard 0.5-approx. (2/3-approx. for $m = 3$)

1.1 Previous Results

We review 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 $f(S)$. Since the value oracle can be easily constructed for quadratic utility functions, all of the results mentioned here are valid for the case of 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.[23)] 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[20)].

We then consider the case of supermodular utility functions. Compared to the case of submodular utility functions, this case attracts less attention in the literature of combinatorial 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 $O(\frac{\sqrt{\log n}}{n})$-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 two cases where utility functions are *submodular* and *supermodular* (see Section 2 for definitions), 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 is 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 for 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. It is shown that the problem can be solved efficiently in $\mathrm{O}(mn^2 \log(mn))$ time by using the reduction to the minimum quadratic-cost flow problem.

For the case of supermodular quadratic utility functions, we firstly show the polynomial-time solvability of (SUP|m=2) by reducing it to the min cut problem in directed graphs. 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.[19] 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) which have submodularity and supermodularity. Throughout this paper we assume $a(v,u) = a(u,v)$ for every $u, v \in V$ with $u < v$.

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

$$f(X \cup \{v\}) - f(X) \geq f(Y \cup \{v\}) - f(Y)$$

for every $X, Y \in 2^V$ with $Y \subseteq X$ and $v \in V \setminus X$. Intuitively, this condition says that the marginal value of an item decreases as the set of items already acquired increases. A utility function $f : 2^V \to \mathbf{R}$ is said to be *supermodular* if $-f$ is submodular. Submodularity (resp., supermodularity) of utility functions is used to model substitutability (resp., complementarity) of items.

Theorem 2.1. *Let $f : 2^V \to \mathbf{R}$ be a quadratic utility function of the form* (1).
(i) *f is submodular if and only if $a(u,v) \leq 0$ $(\forall u, v \in V, u \neq v)$.*
(ii) *Submodular 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 f is submodular if and only if the following inequality holds:

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

which is in turn equivalent to the condition $a(u,v) \leq 0$ $(\forall u, v \in V, u \neq v)$. If f is submodular and monotone, then we have

$$0 \leq f(V) - f(V \setminus \{v\}) = b(v) + \sum_{u \in V \setminus \{v\}} a(u,v)$$

for all $v \in V$. On the other hand, if f is a submodular function satisfying the condition $b(v) + \sum_{u \in V \setminus \{v\}} a(u,v) \geq 0$ $(\forall v \in V)$, then we have

$$f(X) - f(X \setminus \{v\}) = b(v) + \sum_{u \in X \setminus \{v\}} a(u,v) \geq b(v) + \sum_{u \in V \setminus \{v\}} a(u,v) \geq 0$$

for every $X \subseteq V$ and $v \in X$, i.e., f is monotone. □

Theorem 2.2. *Let $f : 2^V \to \mathbf{R}$ be a quadratic utility function of the form* (1).
(i) *f is supermodular if and only if $a(u,v) \geq 0$ $(\forall u, v \in V,\ u \neq v)$.*
(ii) *Supermodular f is monotone if and only if $b(v) \geq 0$ $(\forall v \in V)$.*

Proof. It is well known that f is supermodular if and only if it satisfies the following inequality:

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

which is in turn equivalent to the condition $a(u,v) \geq 0$ $(\forall u, v \in V,\ u \neq v)$. It is easy to see that if the conditions $a(u,v) \geq 0$ $(\forall u, v \in V,\ u \neq v)$ and $b(v) \geq 0$

$(\forall v \in V)$ hold, then f is monotone. On the other hand, if f is monotone, then we have $0 \le f(\{v\}) - f(\emptyset) = b(v)$ 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 $f : 2^V \to \mathbf{R}$ is described as follows:

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

Intuitively, the gross substitutes condition says that the demand for an item does not decrease if the prices of other items increase.

Theorem 2.3 (cf. 14)). *A quadratic utility function $f : 2^V \to \mathbf{R}$ of the form* (1) *satisfies the gross substitutes condition if and only if $a(u, v) \le 0$ $(\forall u, v \in V, u \ne v)$ and $a(u, v) \le \max\{a(u, t), a(v, t)\}$ $(\forall u, v, t \in V, u \ne v, u \ne t, v \ne t)$.*

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

Theorem 2.4 (see 24); see also Theorem 13.5 of 3)). *A utility function $f : 2^V \to \mathbf{R}$ satisfies the gross-substitutes condition if and only if f is submodular and satisfies the following condition for every $X \subseteq V$ and distinct $u, v, t \in V \setminus X$:*

$$\begin{aligned} &f(X \cup \{u, v\}) + f(X \cup \{t\}) \\ &\quad \le \max\{f(X \cup \{u, t\}) + f(X \cup \{v\}), f(X \cup \{v, t\}) + f(X \cup \{u\})\}. \end{aligned} \tag{3}$$

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

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

Therefore, the inequality (3) in Theorem 2.4 is equivalent to $a(u, v) \le \max\{a(u, t), a(v, t)\}$. Hence, the statement of Theorem 2.3 follows from this fact and Theorems 2.1 and 2.4. □

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 game conjecture[17)].

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

$$a_i(u, v) = \begin{cases} -w(u, v) & ((u, v) \in E), \\ 0 & (\text{otherwise}), \end{cases} \quad (u, v \in V, u < v),$$
$$b_i(v) = (1/2) \sum \{w(u, v) \mid (u, v) \in E, u \in V \setminus \{v\}\} \quad (v \in V)$$

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

$$\begin{aligned} &-\sum_{i=1}^{2} \sum \{w(u, v) \mid (u, v) \in E,\ u, v \in V_i\} + \sum_{(u,v) \in E} w(u, v) \\ &= \sum \{w(u, v) \mid (u, v) \in E, u \in V_1, v \in V_2\}, \end{aligned}$$

i.e., the total weight of cut edges in G. Hence, the max cut problem on undirected graphs is reduced to (SUB$|m$=2), and this reduction preserves the approximation ratio. This fact, together with the result in 17), implies the following:

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

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

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 a partition of the vertex set $V \cup \{s,t\}$ satisfying $s \in S, t \in T$. Then, the objective function value of (S,T) is equal to

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

Hence, (SUB$|m{=}2$) is reduced to the max s-t cut problem in G, and this reduction preserves the approximation ratio. It is shown by Lewin et al.[22] 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 Polynomial-Time 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. Corollary 3.4 of 14)). *A quadratic utility function* $f : 2^V \to \mathbf{R}$ *of the form* (1) *satisfies the gross substitutes condition if and only if it is represented as* $f(X) = -\sum_{S \in \mathcal{F}} c_S |X \cap S|^2$ *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* $c_S \geq 0$ $(S \in \mathcal{F}, |S| \geq 2)$.

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 $f_i(X) = -\sum_{S \in \mathcal{F}_i} c^i_S |X \cap S|^2$, where $\mathcal{F}_i \subseteq 2^V$ is a laminar family and c^i_S $(S \in \mathcal{F}_i)$ are real numbers satisfying the conditions in 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 $V_i = \{v^i_S \mid S \in \mathcal{F}_i\}$. Note that $v^i_{\{u\}} \in V_i$ for each $i \in M$ and $u \in V$. We also define $\hat{E} = E_0 \cup \bigcup_{i=1}^m E_i$, where

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

and 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 $(u, v^i_{\{u\}})$, its capacity is given by the interval $[0,1]$, and its cost is 0. For each edge (v^i_X, v^i_Y) or (v^i_X, r) in E_i, its capacity is $[0, +\infty]$, and its cost function is given by $c^i_X \varphi^2$, where φ is the flow value on the edge. We also define supply/demand values of vertices to be 1 for each $u \in V$, $-n$ for r, and 0 for other vertices.

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 $\mathrm{O}(mn)$ vertices and $\mathrm{O}(mn)$ edges, the minimum cost flow problem can be solved in $\mathrm{O}(mn \log(mn)) \times n = \mathrm{O}(mn^2 \log(mn))$ 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* $\mathrm{O}(mn^2 \log(mn))$ *time.*

4. Results for Supermodular Utility Functions

We firstly show that the problem (SUP|m=2) can be solved in polynomial time.

Lemma 4.1 (Theorem 1 of 13)). *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.*

In the following, we consider the problem (SUP|m>2).

4.1 Hardness

To show the NP-hardness of the problem (SUB|m>2), we show that the multiway (un)cut problem on undirected graphs[7),19)] can be reduced to (SUB|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, i.e., $\sum\{w(u, v) \mid (u, v) \in E,\ u \in V_i,\ , v \in V_j,\ i \neq j\}$, 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 (SUB|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 $\sum_{i=1}^{k} \sum\{w(u, v) \mid (u, v) \in E,\ u, v \in V_i\}$, which we want to maximize. Hence, an optimal solution for (SUB|m>2) is an optimal solution for the multiway (un)cut problem, and vice versa.

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

4.2 Approximability

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

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

The algorithm firstly compute an optimal solution of the LP. 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.

We analyze the performance of the algorithm. For $v \in V$ and $i \in M$, let $X^i(v)$ be a random variable that is 1 if the item v is assigned to the bidder i and 0 otherwise. Similarly, for $u, v \in V$ and $i \in M$, let $Y^i(u, v)$ be a random variable that is 1 if both of the items u and v are assigned to the bidder i and 0 otherwise. We also denote $y(u, v) = \sum_{i=1}^{m} y^i(u, v)$ for $u, v \in V$ with $u < v$.

Lemma 4.4 (Fact 3.1 of 19)). *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 (Claim 3.2 of 19)). $\Pr[X^i(v) = 1] = x^i(v)$ *for* $v \in V$ *and* $i \in M$*.*

Lemma 4.6.

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

holds for $u, v \in V$ *with* $u < v$ *and* $i \in M$*.*

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.

The probability that u and v is assigned to an 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

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

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

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

Hence,

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

This implies the claim of the lemma. □

We consider the expected value of the objective function for a solution obtained by the algorithm. By Lemmas 4.5 and 4.6, it holds that

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

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

Theorem 4.7. *A 0.5-approximate solution of the problem* (SUB$|m$>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 $\varepsilon = \min\{y(u,v) \mid (u,v)\in E,\ y(u,v)>0\}$ (analyze the cases $y(u,v)=0$ and $y(u,v)>0$ separately).

Our analysis shows that the integrality gap of the LP is at least 0.5. On the other hand, an instance with integrality gap 2/3 can be easily constructed. 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 $\min\{y(u,v) \mid (u,v)\in E,\ y(u,v)>0\}$.

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 by using this fact. 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* (SUB$|m$>2) *with* $m=3$ *can be computed in polynomial time.*

For 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 `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 `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 this 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.

References

1) R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice Hall, 1993.
2) L. Blumrosen and N. Nisan, Combinatorial auction, Ch.11 of Algorithmic Game Theory, N. Nisan, et al.(eds.), Cambridge Univ.Press, 2007.
3) P. Cramton, Y. Shoham, R. Steinberg, Combinatorial Auctions, MIT Press, 2006.
4) J. Cheriyan, T. Hagerup, K. Mehlhorn, An o(n)-time maximum-flow algorithm, SIAM J.Computing 25 (1996) 1144–1170.
5) Y. Chevaleyre, U. Endriss, S. Estivie, and N. Maudet, Multiagent resource alloca-

tion in k-additive domains: preference representation and complexity, Annals OR 163 (2008) 49–62.
6) V. Conitzer, T. Sandholm, and P. Santi, Combinatorial auctions with k-wise dependent valuations, Proc.AAAI 2005, 248–254.
7) E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, M. Yannakakis, The complexity of multiterminal cuts, SIAM J.Computing 23 (1994) 864–894.
8) M.L. Fredman and R.E. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, J. ACM 34 (1987) 596–615.
9) S. Fujishige, Submodular Function and Optimization, 2nd Edition, Elsevier, 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, J.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 J.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, R.O'Donnell, Optimal inapproximability results for max-cut and other 2-variable CSPs?, Proc. 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) M. Langberg, Y. Rabani, and C. Swamy, Approximation algorithms for graph homomorphism problems, Proc.APPROX-RANDOM 2006, 176–187.
20) B. Lehmann, D. Lehmann, and N. Nisan, Combinatorial auctions with decreasing marginal utilities, Games and Economic Behavior 55 (2006) 270–296.
21) D. Lehmann, L. O'Callaghan, and Y. Shoham, Truth revelation in approximately efficient combinatorial auctions, Proc.EC 1999, 96–102.
22) M. Lewin, D.Livnat, and U.Zwick, Improved rounding techniques for the MAX 2-SAT and MAX DI-CUT problems, Proc.IPCO 2002, 67–82
23) V.Mirrokni, M.Schapira, and J.Vondrák, Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions, Proc.EC 2008, 70–77.
24) H. Reijniese, A. van Gellekom, and J. A. M. Potters, Verifying gross substitutability, Economic Theory 20 (2002) 767–776.
25) T. Sandholm, Algorithm for optimal winner determination in combinatorial auctions, Artificial Intelligence 135 (2002) 1–54.
26) J. Vondrák, Optimal approximation for the submodular welfare problem in the value oracle model, Proc.STOC 2008, 67–74.