Decentralized Combinatorial Auctions for Multi-Unit Resource Allocation

Li-Hsing Yen and Guang-Hong Sun
Department of Computer Science, College of Computer Science
National Chiao Tung University, Hsinchu, Taiwan 300, R.O.C.
Email: lhyen@cs.nctu.edu.tw, martian206@gmail.com

***Abstract*—Auction has been used to allocate resources or tasks to processes, machines or other autonomous entities in distributed systems. When different bidders have different demands and valuations on different types of resources or tasks, the auction becomes a combinatorial auction (CA), for which finding an optimal auction result that maximizes total winning bid is NP-hard. Many time-efficient approximations to this problem work with a bid ranking function (BRF). However, existing approximations are all centralized and mostly for single-unit resource. In this paper, we propose the first decentralized CA schemes for multi-unit resources. It includes a BRF-based winner determination scheme that enables every agent to locally compute a critical bid value for her to win the CA and accordingly take her best response to other agent's win declaration. It also includes a critical-value-based pricing scheme for each winner to locally compute her payment. We analyze stabilization, correctness, and consistency properties of the proposed approach. Simulation results confirms that the proposed approach identifies exactly the same set of winners as the centralized counterpart regardless of initial bid setting, but at the cost of lower total winning bid and payment.**

I. Introduction

Auction is a trading process that allows someone with items to sell (i.e., a seller) to identify potential buyers and the prices the buyers are willing to pay. We may use auctions as resource and task allocation schemes. Unlike conventional approaches that assume zero or fixed cost of resource or task, auction-based approaches can allocate resource/task to requesters in a way that reflects actual demand and supply conditions. For this reason, auctions have been used to allocate different types of resources or tasks to a fleet of autonomous, self-interest agents. Existing examples include but not limited to allocations of wireless spectrum [1], [2], cloud resource [3], [4], [5], [6], [7], and tasks of robots [8], [9].

The key to the success of auction-based resource allocations is to identify bidders' valuations on resources. Generally speaking, different requesters have different valuations on the same resource. For example, a cloud user with a computation-intensive task may value computation resource more than other users with storage-intensive tasks. One possible goal of auction design is to maximize the aggregated valuation from all winning bidders, a quantity called *social welfare*. An auction mechanism that always maximizes the social welfare is *economically efficient*, which is desirable when we have to maximize the utilities of requesters rather than the revenue of the seller.

A challenge to the design of an economically-efficient mechanism is that bidder's valuations on resources are considered local and private (i.e., not revealed to other bidders and the auctioneer who conducts the auction). The auctioneer can only get aware of such information indirectly through bid values. In this regard, bids indicate bidder's preferences for the items to sell. But bids are beyond preferences, because bids should somehow be associated with payments to prevent bidders that do not value an item the most from winning it by placing an excessive bid. If every bidder places a bid that is equivalent to her true valuation, a property called *truthful bidding*, the social welfare is equivalent to the collection of bids from all winners. Therefore, an auction mechanism is economically efficient if it both ensures truthful bidding and maximizes the collection of bids from all winners.

In the simplest form of auction, there is only item to sell. Some studies assumed a single type of resource with multiple supplying units [10], [4], [7]. We consider a more general form called *combinatorial auction* (CA) [11], [12], where more than one types of items are allocated to bidders at a time via an auction. A typical example is in cloud environment, where we have computation, memory, storage, network, and other types of resources. Each agent (i.e., bidder) has an interest in and thus places a bid on a particular combination of items (in the form of virtual machines, for example). Unlike what happens in a single-item auction, more than one agents could be winners in a CA (if these agents have non-conflicting interests). In a *single-unit* CA, there is only one instance for each type of item. In a *multi-unit* CA, where there can be multiple identical instances for one type of item, two or more agents can be winners at the same time even if they have conflicting interests. Identifying a feasible set of winning bids with the highest total bid in CA is a task called *winner determination problem* (WDP). WDP is NP-hard even for single-unit CA [11]. So most existing approaches are approximations. We particularly consider approximations that are based on a *bid ranking function* (BRF), which defines a total order on bid requests. BRF-based approximations are time efficient but cannot guarantee economic efficiency.

Another task of CA is *pricing scheme*, which decides the payment of each winner. A pricing scheme is *incentive compatible* if every agent's dominant strategy is to bid truthfully. Incentive-compatible pricing scheme together with an optimal winner determination can ensure economical efficiency. The most well-known incentive-compatible pricing scheme for CA is the Vickrey-Clarke-Groves (VCG) mechanism [13], [14], [15]. VCG relies on the optimal solution to the winner determination problem, so it is not computationally feasible. We consider a computationally-efficient heuristic called critical-value-based payment that depends on the definition of the associated BRF. It has been proved that if the associated BRF is monotone, a critical-value-based payment is incentive compatible [16].

Almost all CA approaches implicitly assume a single entity, i.e., an auctioneer, to conduct auctions. Such a centralized approach is not robust and scalable because the auctioneer is a single point of failure and can be a performance bottleneck. So some approaches attempt distributing the load of auctioneer to sets of brokers [17], [18]. Another issue with centralized CA approach is that decentralized winner determination may be preferable when supplies of and requests for resources exhibit *locality* property as exemplified below.

- Resource is only accessible to "local" users. An example is wireless spectrum resource.
- Users only have interest in locally-accessible resource. An example is virtualized resource provided by edge servers in mobile edge computing environment.

In these cases, potential competitors contending for the same type of resource tend to cluster together and it is nature for they themselves to coordinately determine the set of resource winners. Furthermore, in applications like robot task allocations, it is more desirable to let robots themselves coordinate their tasks because, compared with a central coordination approach that decides a global task allocation, the decentralized approach has a shorter reaction time.

In this paper, we propose a *decentralized* BRF-based winner determination and associated pricing schemes for multi-unit CAs where bidders autonomously decide whether they themselves are winners and how much they should pay. This work is not just to duplicate or partition auctioneer's computation load to a set of participants. The key to the benefit and the feasibility of the decentralization is that *only* bidders that place bids on a common item have conflicting interests and thus need to interact with one another for winner determination. The competitions among bidders can be captured by a *conflict graph*, where bidders are nodes and there is an edge for each pair of competing bidders. Fig. 1 shows an example of conflict graph for seven bidders in a CA. Here, for bidding agent a_5 to determine whether she can win her bid, she only needs to know the bidding information and the status of her competitors (a_4, a_6, and a_7). Agent a_5 is free to declare her win in the CA without information from other bidders (a_1, a_2, and a_3). This suggests a localized, autonomous winner determination mechanism, which is more robust and scalable than a centralized one.

Without a centralized control, bidders in a decentralized CA have to exchange bidding information for collaborative winner determinations. Bidding protocols under this framework face several challenges. First, if bidders can gain extra payoff by having knowledge of other bids before they place their own

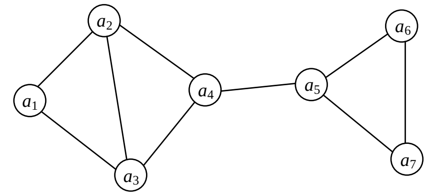


Fig. 1. A conflict graph for seven bidders in a CA

bids, they may intentionally postpone their decision makings until they receive bidding information of their competitors. The consequence is that the whole system may enter a deadlock state simply because no bidder wants to place her bid first. Second, bidders progresses asynchronously due to the lack of a synchronization scheme among them. The non-deterministic interactions among bidders may not converge to a stabilized result without appropriate regulations. Third, even if the protocol reaches a stabilized outcome, the outcome may not be the same as that of the corresponding centralized counterpart using the same BRF.

A. Contribution and Organization

The contribution of this work is summarized as follows.

- We propose a protocol for WDP in multi-unit CA which works for any given BRF that is monotone. This protocol is deadlock free because it allows bidders to revise and update their bid requests whenever they want to react to other bidder's updates (as their best response). This protocol guarantees stabilization in the face of dynamic bidder interactions. For a specific BRF, the proposed decentralized approach can yield the same set of winners as the centralized counterpart, despite that the ranks of bid requests and payments may be different in the two approaches.
- The associated pricing scheme is no longer incentive compatible because bidders may change their bids *after* observing bidding information of others, i.e., it is not a static (one-shot) game. However, the proposed critical-value-based payment is still *strategy-proof*, meaning that no winner can decrease her payment by unilaterally manipulating her own bid.

To the best knowledge of the authors, this is the first decentralized CA approach that possesses these properties. We have conducted extensive simulations to investigate the performance of the proposed approaches.

The rest of this paper is organized as follows. Sec. II covers the background and related work of the problem. In Sec. III, we present the proposed scheme in details and analyze its properties. Sec. IV contains the simulation results that confirm the advantage of our algorithm. The last section concludes this paper.

II. Problem Definition and Related Work

We consider a set of n bidding agents (bidders) $A = \{a_1, a_2, \ldots, a_n\}$ and m different types of resources $R =$

$\{r_1, r_2, \ldots, r_m\}$. Let $\mathbf{q} = (q_1, q_2, \ldots, q_m)$ be a supply vector such that $q_i \geq 1$ is the number of identical instances of resource type r_i. For single-unit CA, $q_i = 1$ for all i. Each agent a_i may submit a *request vector* $\mathbf{s}_i = (s_i^1, s_i^2, \ldots, s_i^m)$, where $s_i^j \leq q_j$ is the number of instances of resource type r_j requested by a_i. For single-unit CA, $\mathbf{s}_i$ reduces to a set (named *bundle*) $S_i \subseteq R$. Each agent a_i has an evaluation on $\mathbf{s}_i$ denoted by $v_i(\mathbf{s}_i)$, which is private. We do not allow for externalities, which means that $v_i(\cdot)$ does not depend on any $v_j(\cdot)$ with $j \neq i$. We also assume that agents are *single-minded* [16], i.e., every agent a_i is interested in only one particular request $\mathbf{s}_i$ and does not accept any winning result that does not comprise $\mathbf{s}_i$. Therefore, for every request vector $\mathbf{t}_i = (t_i^1, t_i^2, \ldots, t_i^m)$, $v_i(\mathbf{t}_i) = v_i \in \mathbb{R}^+$ if $s_i^j \leq t_i^j$ for all j and $v_i(\mathbf{t}_i) = 0$ otherwise. The bid a_i places on $\mathbf{s}_i$ is denoted by $b_i(\mathbf{s}_i)$ or simply b_i, which together with $\mathbf{s}_i$ forms a_i's *bid request* $(\mathbf{s}_i, b_i)$. We assume that an agent only submits one bid request. If an agent may submit multiple requests (i.e., OR bids [19]), we can treat A as a set of requests rather than agents. If an agent is allowed to submit but not to win multiple requests (i.e., XOR bids [19]), we may manually add mutual-exclusive relation between each pair of requests submitted by the same agent[1].

A. Winner Determination Problem

Given a set of bid requests $\mathcal{B} = \{(\mathbf{s}_i, b_i(\mathbf{s}_i))\}_{i=1}^n$, the winner determination problem (WDP) is to find a setting of $X = (x_1, x_2, \ldots, x_n)$, where $x_i \in \{0, 1\}$ for all i, that maximizes the *total winning bid*

$$\sum_{x_i=1} b_i(\mathbf{s}_i) \tag{1}$$

subject to the *resource capacity constraint* defined as

$$\sum_{i=1}^{n} \left(x_i \cdot s_i^k\right) \leq q_k \text{ for all } k = 1, \ldots, m. \tag{2}$$

For single-unit CA, the WDP an instance of the *maximum weight set packing problem*, which is known to be NP-hard [11]. Some approaches guarantee optimality but may be time-inefficient for some problem instances [19], [21]. Some approaches are time-efficient and achieve optimality by restricting the form or size of bid requests [22]. Some approaches use heuristic or approximation techniques for time efficiency but not optimality. Hoos and Boutilier [23] used stochastic local search algorithm as an approximation to WDP. Zurel and Nisan [24] also proposed an approximation which runs the linear-programming relaxation of the packing problem and then refines the solution by local improvements in the order of bids (hill-climbing). The hill-climbing concept was also adopted by Fukuta and Ito [25] to improve the performance of a simple greedy approach [16]. They also considered the use of simulated annealing technique. Other approximation approaches include dynamic programming [3] and genetic algorithm [26].

[1] One possible way of doing this is through the creation of *dummy goods* [20]. Also note that all OR bids can be converted into equivalent XOR bids [19].

In this paper, we mainly consider approximations that use a BRF to define a total order $\prec$ on $\{(\mathbf{s}_i, b_i)\}_{i=1}^n$ such that $(\mathbf{s}_j, b_j) \prec (\mathbf{s}_i, b_i)$ if $(\mathbf{s}_j, b_j)$ ranks higher than $(\mathbf{s}_i, b_i)$. Algorithm 1 shows the general framework for greedy allocations which examine all bid requests in the order defined by $\prec$ to determine whether each request can be granted.

Algorithm 1 BRF-based Greedy Allocation

$\mathcal{B} \leftarrow \{(\mathbf{s}_i, b_i)\}_{i=1}^n$
$x_i \leftarrow 0$ for all i
while $\mathcal{B} \neq \emptyset$ **do**
  Let $(\mathbf{s}_k, b_k)$ be the request that ranks first in $\mathcal{B}$
  if $\mathbf{q} - \mathbf{s}_k > \mathbf{0}$ **then**
    $\mathbf{q} \leftarrow \mathbf{q} - \mathbf{s}_k$
    $x_k \leftarrow 1$
  end if
  $\mathcal{B} \leftarrow \mathcal{B} \setminus \{(\mathbf{s}_k, b_k)\}$
end while

There have been many BRFs proposed for single-unit CA. The BRF proposed by Lehmann et al. [16] favors a request that maximizes normalized bid value defined as

$$w_s(S_i, b_i) = \frac{b_i}{|S_i|^\alpha}, \tag{3}$$

where α is a configurable parameter. Mito and Fujita [27] considered several possible BRFs inspired by the heuristics for the maximum weighted independent set (MWIS) problem [28]. Let N_i be the set of all conflicting requests for request (S_i, b_i). One such BRF sets a priority defined as

$$w_n(S_i, b_i) = \frac{b_i}{(|N_i| + 1)^\beta}, \tag{4}$$

where β is a configurable parameter. Another BRF considered by them is

$$w_\phi(S_i, b_i) = \frac{\phi(S_i, b_i)}{(\sum_{(S_j, b_j) \in N_i} b_j + 1)^\beta}, \tag{5}$$

where

$$\phi(S_i, b_i) = \frac{b_i}{(\sum_{(S_j, b_j) \in N_i} |S_i \cap S_j| + 1)^\alpha}. \tag{6}$$

Function $\phi(\cdot)$ alone could also be a BRF.

Not too many approaches have been proposed for multi-unit CA. Leyton-Brown et al. [20] proposed an optimal WDP algorithm. This algorithm uses techniques like branch-and-bound and dynamic programming, which makes it difficult to be decentralized. As an approach to allocating fine-grained spectrum resources, Jia et al. [29] generalized the BRF $w_s(\cdot)$ defined in (3) to multi-unit CAs. The proposed BRF is

$$w_m(\mathbf{s}_i, b_i) = \frac{b_i}{(\sum_{k=1}^{m} s_i^k)^\alpha}. \tag{7}$$

The same BRF has also been used for the allocation of virtual machine instances in clouds [30], [31]. The work in [32] generalized the BRF to consider scarcity of resources with

$\alpha = 0.5$. Mashayekhy et al. [5] considered the following BRF for a bid request in a unit of time.

$$w_d(\mathbf{s}_i, b_i) = \frac{b_i}{\prod_{s_i^k \neq 0} s_i^k}. \tag{8}$$

Some BRFs for single-unit CA like (4) do not consider the number of resource instances. When being used in multi-unit CA, these BRFs may perform poorly. BRFs like (5) and (6) have not yet been extended to handle multi-unit resources. A possible extension is to replace $|S_i \cap S_j|$ in (6) with some matching term like $\mathbf{s}_i \cdot \mathbf{s}_j$.

B. Pricing Scheme

The VCG mechanism generalizes the second-price scheme to ensure truthful bidding in CAs. VCG demands that each winner a_i in VCG has to pay the social opportunity cost (i.e., the reduction of the total winning bid excluding a_i's) due to the presence of a_i's request. Suppose that we have a set of request pairs $\mathcal{B} = \{(S_i, b_i)\}_{i=1}^n$. Let $\mathcal{B}_{-i} = \mathcal{B} \setminus \{(S_i, b_i)\}$ and $W(\mathcal{B}) \subseteq \mathcal{B}$ be a set of winning requests with the highest total bid given $\mathcal{B}$. Each winning request $(S_i, b_i) \in W(\mathcal{B})$ has to pay $p_i(\mathcal{B}) = \sum_{(S_j,b_j)\in W(\mathcal{B}_{-i})} b_j - \sum_{(S_k,b_k)\in W(\mathcal{B})\setminus\{(S_i,b_i)\}} b_k$. VCG payment has been used in [33]. VCG is economically efficient but computationally infeasible because computing $W(\mathcal{B})$ is NP-hard.

For BRF-based winner determination designed for single-unit CA, Lehmann et al. [16] defined *monotonicity* property for BRF, which states that the BRF gives (S_j, b_j) a rank equal to or higher than that of (S_i, b_i) if $S_j \subseteq S_i$ and $b_j \geq b_i$. BRF $w_s(\cdot)$ has the monotonicity property. BRFs $w_n(\cdot)$, $w_\phi(\cdot)$, and $\phi(\cdot)$ do not ensure monotonicity because it is possible that $S'_i \subset S_i$ but $(S'_i, b_i) \not\succ (S_i, b_i)$ as long as the set N_i remains unchanged for both S_i and S'_i.

The monotonicity property can be generalized for multi-unit CA as follows. Let $\mathbf{s}_i = (s_i^1, s_i^2, \ldots, s_i^m)$ be a_i's request vector. Let $\mathcal{S} = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n\}$. Define binary relation $\leq$ on $\mathcal{S}$ as $\mathbf{s}_i \leq \mathbf{s}_j$ if $s_i^k \leq s_j^k$ for all $k \in \{1, \ldots, m\}$. Define binary relation $<$ on $\mathcal{S}$ as $\mathbf{s}_i < \mathbf{s}_j$ if $\mathbf{s}_i \leq \mathbf{s}_j$ and $\mathbf{s}_i \neq \mathbf{s}_j$. A BRF is monotone if it gives $(\mathbf{s}_j, b_j)$ a rank equal to or higher than that of $(\mathbf{s}_i, b_i)$ whenever $\mathbf{s}_j \leq \mathbf{s}_i$ and $b_j \geq b_i$. By this definition, both $w_m(\cdot)$ and $w_d(\cdot)$ are monotone.

An allocation of resources among single-minded bidders is *exact* if each bidder a_i is allocated either its request S_i (if a_i wins) or nothing (otherwise) [16]. We also assume the exactness property for our multi-unit CA.

For a BRF-based winner determination for single-unit CA with both the exactness and monotonicity properties, it is proved [16] that there is a critical value c_i for each b_i such that a_i gets S_i if $b_i > c_i$ and a_i gets nothing if $b_i < c_i$.

For example, assume that (S_i, b_i) is a winning request and (S_j, b_j) is the request that has the highest rank in the set of requests that do not win because of the presence of (S_i, b_i). For the BRF defined in (3), (S_i, b_i) is a winning request because $w_s(S_i, b_i) > w_s(S_j, b_j)$, which implies that

$$b_i > b_j \frac{|S_i|^\alpha}{|S_j|^\alpha}. \tag{9}$$

On the other hand, (S_i, b_i) would not be a winning request if $w_s(S_i, b_i) < w_s(S_j, b_j)$ or, equivalently, if

$$b_i < b_j \frac{|S_i|^\alpha}{|S_j|^\alpha}. \tag{10}$$

Therefore, $c_i = b_j \times |S_i|^\alpha / |S_j|^\alpha$ is the critical value for b_i.[2]

It is not difficult to see that a critical value also exists for each bidder in a multi-unit CA with the same setting.

In a *sealed-bid* auction, agents simultaneously submit sealed bids to an auctioneer, who then determines the set of winners and their associated payments. Agents in this case have no information about other bids. If a BRF-based winner determination is used for which the criticality property holds, then the following pricing scheme ensures truthful bidding in a sealed-bid CA [16].

$$p_i = \begin{cases} c_i, & \text{if } x_i = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

C. Decentralized Mechanisms

In the literature, decentralized auctions sometimes refer to the idea of decoupling the WDP into subproblems each solved by a local agent. Each agent solves its locally constrained utility maximization problem, but an auctioneer is still needed to collect and update bidding information for coordinating an iterative auction [34]. Parkes [35] proposed an ascending-price auction protocol which executes in rounds. At the start of every round, the auctioneer announces a provisional allocation of bundles to bidders as well as new prices for another set of bundles. Brewer [17] proposed an auction protocol where participants may place higher bids on known bundles or propose alternative allocations to earn bonus.

Esteva and Padget [36] proposed a decentralized approach to WDP based on leader election protocol running on a ring overlay network. Their approach targets at single-item auction so there is only one winner. Lewis et al. [37] proposed a decentralized adaptive pricing scheme for sellers to determine best selling prices in a posted-price model, where sellers announce prices and buyers decide the quantities to purchase from each seller. They considered a single type of quantitatively-divisible resource.

Choi et al. [8] proposed two decentralized auctions for the assignment of different tasks to a fleet of robotic agents. One of these auctions assigns each agent multiple tasks. The auction consists of two phases. In the first phase, each agent independently and incrementally constructs its own bundle of tasks and place their bids on tasks based on the rewards (i.e., utilities) they may receive from the bundle. By exchanging information with all other participants, a bidder knows the most up-to-date standing bid on each task and may add an additional task into its bundle with a higher standing bid for this task. Because bidders construct their bundles independently, they may have conflicting collections of tasks. The second phase resolves possible bundle conflicts by applying a set of arbitration rules.

[2]Though not explicitly stated, critical values should also exist for other monotone BRFs with exact allocations [27].

As a result, some agent needs to release all the tasks contained in her bundle, and the algorithm returns to the first phrase.

Our approaches differ from the approaches by Choi et al. [8] in the following points. First, their approaches assume that bidders are intrinsically truthful, while ours does not. Second, bidders in their approaches dynamically form their bundles and bids, while bidders in our approaches are single-minded. Third, our winner-determination approaches guarantee a conflict-free consensus, so no extra conflict-resolution phase is needed to re-examine the assignment result. Fourth, every task in [8] should be assigned, but this is not a requirement in our approaches. Finally, our approaches consider multi-unit CAs, not only single-unit CAs.

III. The Proposed Approach

Designing a decentralized multi-unit CA faces several challenges. The first one is how each bidder locally determines whether the bidder herself is a winner according to the given BRF. It is not trivial because bidders competing for a common resource may be winners at the same time. The second challenge is to ensure that bidder interactions do not cause instability of the whole auction scheme, i.e., all bidders reach to a consensus on the set of final winners despite the fact that bidder may quit bidding or revise its bid request as a reaction to a change of another bid request. The third challenge is to make each bidder independently figure out how much it should pay for the auction.

A. Dynamic Multi-unit CA Game

In the proposed framework, bidder independently sets up bid request and then notifies all competitors of that setting. The setting may cause the competitors to make their own moves. Because notifications take arbitrary time and there is no synchronization scheme to coordinate bidder's moves, bidders make moves one after another in a non-deterministic manner. We thus model multi-unit CA as a dynamic game. In contrast, bidders in sealed-bid CAs place their bids without bidding information of any others, rendering it a static (one-shot) game.

We assume a monotone BRF for winner determination, which defines a total order $\prec$ on $\mathcal{B} = \{(\mathbf{s}_i, b_i)\}_{i=1}^n$. We also assume exact allocation and single-minded bidders. Therefore, a single-minded a_i has no incentive to submit $\mathbf{s}'_i < \mathbf{s}_i$ if a_i desires $\mathbf{s}_i$ because $v_i(\mathbf{s}'_i) = 0$. On the other hand, if a_i submits $\mathbf{s}'_i$ such that $\mathbf{s}_i < \mathbf{s}'_i$, the probability of winning its request is lowered (due to the monotonicity property) without increasing its payoff. Therefore, agent a_i has no incentive to manipulate $\mathbf{s}_i$ and does not change $\mathbf{s}_i$ during the auction: a_i submits $(\mathbf{s}_i, b_i)$ for the first bid request and b_i only for all subsequent bid requests. In open ascending-price auctions and other decentralized auctions [17], [35], [38], agents can only *raise* their bids. We take the same assumption.

Besides setting up bid requests, bidders also need to declare whether they themselves are winners and notify their competitors of the declarations. The notification is needed because bid requests alone do not suffice for the competitors to make their decisions. For example, suppose that a_6 in Fig. 1 can be a winner only if a_5 is not but whether a_5 can win may depend on whether a_4 wins. Bidder a_6 alone cannot deduce a_5's win or loss just by comparing their own bid requests; a_5 should notify a_6 of her winning declaration.

We use x_i to denote a_i's declaration, where $x_i = 1$ if a_i declares a win and $x_i = 0$ otherwise. In the proposed game, x_i is part of a_i's strategy and interpreted as a_i's willingness to win and pay. It is therefore possible that a_i changes x_i from 0 to 1 or from 1 to 0 without changing b_i. Our approach thus allows bidders to withdraw their current bids. However, bidders in any other auction have no freedom to configure x_i's because bidders are implicitly assumed to be always willing to win with their current bids.

It is a concern whether x_i's are *correct* regarding the resource capacity constraint and the BRF. For this, we extend the definition of critical value for single-unit CA in [16] to multi-unit CAs as follows.

Definition 1 (Critical Value): Given $\mathcal{B} = \{(\mathbf{s}_j, b_j)\}_{j=1}^n$ and $\{x_j | j \neq i\}$, the critical value of b_i is c_i if

$$\sum_{(\mathbf{s}_j, b_j) \prec (\mathbf{s}_i, b_i)} \left(x_j \cdot s_j^k\right) \leq q_k - s_i^k \text{ for all } s_i^k \neq 0 \tag{12}$$

when $b_i > c_i$, and

$$\sum_{(\mathbf{s}_j, b_j) \prec (\mathbf{s}_i, b_i)} \left(x_j \cdot s_j^k\right) > q_k - s_i^k \text{ for some } s_i^k \neq 0 \tag{13}$$

when $b_i < c_i$. If (12) holds when $b_i \geq 0$, we define $c_i = 0$. Intuitively, a_i wins only if $b_i \geq c_i$ and does not only if $b_i \leq c_i$. Therefore, $x_i = 1$ is *correct* if $b_i \geq c_i$ and $x_i = 0$ is correct if $b_i \leq c_i$.[3]

Let p_i be the price that a_i has to pay at the end of the auction. The utility of a_i is defined to be a_i's payoff in the auction:

$$u_i = x_i \left(v_i(\mathbf{s}_i) - p_i\right). \tag{14}$$

Intuitively, a_i's utility is her valuation on $\mathbf{s}_i$ minus her payment p_i if a_i declares a win, and zero otherwise.

We assume critical-value-based payment (i.e., $p_i = c_i$). If a_i does not declare a win, a_i gets zero utility by (14). If a_i declares a win (i.e. $x_i = 1$), a_i's utility is $u_i = v_i(\mathbf{s}_i) - c_i$. It appears that a_i can declare x_i without setting up a matching b_i. However, a_i has no incentive to declare $x_i = 0$ if a_i in fact wins (i.e., $b_i > c_i$). On the other hand, if a_i declares $x_i = 1$ but $b_i < c_i$, it is implied that $\sum_{(\mathbf{s}_j, b_j) \prec (\mathbf{s}_i, b_i)} \left(x_j \cdot s_j^k\right) > q_k - s_i^k$ for some $s_i^k \neq 0$. Any agent a_j with $s_j^k > 0$ can detect this. We assume that there is some penalty imposed on bidders who get caught cheating on x_i, so a_i has no incentive to declare $x_i = 1$ if a_i in fact loses (i.e., $b_i < c_i$). However, a_i could set up x_i and b_i at the same time to win an auction. It can be deduced that a_i's best response (the setting of b_i and x_i) depends on the relationship between $v_i = v_i(\mathbf{s}_i)$ and c_i:

$$BR_i = \begin{cases} x_i = 0, & \text{if } c_i > v_i, \\ x_i = 1 \text{ and } b_i > c_i, & \text{if } c_i < v_i. \end{cases} \tag{15}$$

[3] x_i can never be incorrect when $b_i = c_i$.

B. Decentralized BRF-based Winner Determination

We now describe the details of the proposed decentralized winner determination scheme. This scheme works for both first-price auction and auctions with critical-value-based payment.

Each agent a_i in the scheme is free to set up b_i and x_i. Different agents may have different ideas about how to place their initial bids, so we assume that b_i is an arbitrary value in $[0, v_i]$ initially. The initial value of x_i is not important, so it could be either 0 or 1. We assume that each agent a_i broadcasts $(\mathbf{s}_i, b_i, x_i)$ to all other agents in the beginning of the scheme so each agent a_i has knowledge of N_i, $\mathcal{B}$, and $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ initially.

Our approach demands that each agent a_i maintains a local copy of b_j and x_j for each $a_j \in N_i$. When a_i receives a new update of b_j and x_j from another agent a_j, it executes Algorithm 2 as a response. a_i first updates its knowledge about b_j and x_j, and checks whether a_i can win with its current bid by identifying a_i's *key predecessor*.

Algorithm 2 Best Response of Agent a_i

```
1:  On initiation or receiving update(b'_j, x'_j) from a_j ∈ N_i
2:      b_j = b'_j; x_j = x'_j
3:      k ← key_predecessor(i, B, x)
4:      if k = i then
5:          b'_i ← b_i; x'_i ← 1
6:      else                                   ▷ (s_k, b_k) ≺ (s_i, b_i)
7:          c_i ← min_b{(s_i, b) ≺ (s_k, b_k)}
8:          if c_i + ε < v_i then ▷ ε: minimum allowable increment
9:              b'_i ← b ∈ [c_i + ε, v_i]; x'_i ← 1
10:         else                               ▷ b_i ≤ c_i and c_i ≥ v_i
11:             b'_i ← b_i; x'_i ← 0
12:         end if
13:     end if
14:     if x'_i ≠ x_i or b'_i ≠ b_i then
15:         x_i ← x'_i; b_i ← b'_i
16:         send update(b_i, x_i) to each a_j ∈ N_i
17:     end if
18: end
```

Definition 2 (Key Predecessor): If a_i can win with its current bid, a_i's key predecessor is a_i itself. Otherwise, a_i's key predecessor is a_k if $(\mathbf{s}_k, b_k)$ is the request that ranks the lowest among all winning requests whose absence alone would make $(\mathbf{s}_i, b_i)$ granted.

If a_i's key predecessor is $a_k \neq a_i$, we have $(\mathbf{s}_k, b_k) \prec (\mathbf{s}_i, b_i)$ and a_k must a neighboring node of a_i in the conflict graph. Algorithm 3 details how to identify the key predecessor for a_i. If a_i cannot win with its current bid, a_i finds out its critical value c_i, i.e., the minimal value of b_i that allows $(\mathbf{s}_i, b_i)$ to outrank $(\mathbf{s}_k, b_k)$. Assuming $\mathbf{q} = (3, 2, 2, 2, 2)$ and the BRF defined in (7) with $\alpha = 1$, Table I shows an example of key predecessors and critical values. The key predecessor of a_3 is a_2 because $(\mathbf{s}_3, b_3)$ would be granted if $(\mathbf{s}_2, b_2)$ were not present. To let $(\mathbf{s}_3, b_3)$ outrank $(\mathbf{s}_2, b_2)$, b_3 should be greater than

$$c_3 = \frac{b_2 (\sum_{k=1}^{m} s_3^k)^\alpha}{(\sum_{k=1}^{m} s_2^k)^\alpha} = \frac{70 \times 4}{3} = 93.33. \tag{16}$$

Therefore, a_3's critical value c_3 is 93.33. The key predecessor of a_5 is a_1 because only the absence of a_1's request alone can make a_5's request granted. Neither a_2 nor a_4 is a_5's key predecessor.

Algorithm 3 Procedure *key_predecessor*$(i, \mathcal{B}, \mathbf{x})$

```
N_i ← {a_j | (a_i, a_j) is an edge in the conflict graph}
C ← {(s_j, b_j) | a_j ∈ N_i ∧ x_j = 1 ∧ (s_j, b_j) ≺ (s_i, b_i)}
total_unit[k] ← 0 for all k ∈ {1, ..., m}
while C ≠ ∅ do
    Let (s_j, b_j) be the request that has the highest rank in C
    for all k ∈ {1, ..., m} do
        total_unit[k] ← total_unit[k] + s_j^k
        if s_i^k > 0 ∧ s_i^k + total_unit[k] > q_k then
            return j
        end if
    end for
    C ← C \ {(s_j, b_j)}
end while
return i
```

Agent a_i can win its request by setting b_i to a value equal to or greater than $c_i + \epsilon$, where ϵ is the minimum allowable increment on bid. However, whether it is worthy for a_i to win depends on the relationship between $c_i + \epsilon$ and v_i. If $c_i + \epsilon < v_i$, a_i can win and get positive utility by changing b_i to some value in the range $[c_i + \epsilon, v_i]$[4]. Otherwise, a_i has no incentive to change b_i because winning the request only gives her a negative payoff (as $u_i = v_i - p_i < 0$). If a_i ever changes b_i or x_i, a_i notifies all a_i's competitors of the update.

C. Stabilization, Correctness, and Consistency

In this subsection, we analyze whether the proposed protocol stabilizes, and, if it does, whether the outcome is correct and consistent.

The protocol may potentially not stabilize because every time a_i changes b_i or x_i, the rank of its bid request in terms of $\prec$ and thus the key predecessors of other bidders may change. That may cause another bidder's reaction and change the set of (declared) winners. For convergence, we consider first how each agent's knowledge about bids and winner declarations evolves with time. Let $\mathbf{b}_i^t = (b_1, b_2, \ldots, b_n)$ and $\mathbf{x}_i^t = (x_1, x_2, \ldots, x_n)$ denote a_i's knowledge of all b_j's and x_j's, respectively, after the t-th execution of Line 15 of Algorithm 2 by a_i. Let $\mathbf{b}_i^0$ and $\mathbf{x}_i^0$ be a_i's initial knowledge. Note that $(\mathbf{b}_i^t, \mathbf{x}_i^t) \neq (\mathbf{b}_i^{t+1}, \mathbf{x}_i^{t+1})$ for all $t \geq 0$. A sequence $\xi_i^t = (\mathbf{b}_i^0, \mathbf{x}_i^0)$, $(\mathbf{b}_i^1, \mathbf{x}_i^1)$, $(\mathbf{b}_i^2, \mathbf{x}_i^2)$, $\ldots$, $(\mathbf{b}_i^t, \mathbf{x}_i^t)$ is a *transition path* of agent a_i that is of length t.

Line 9 in Algorithm 2 is the only place for a_i to change its bid, which is conditioned on the relationship between c_i and v_i (Line 8). Clearly, agents can only raise their bids and no agent places a bid higher than its valuation.

Theorem 1: Any transition path of any agent is finite.

Proof: Consider any agent a_i and one of its transition paths $\xi_i = (\mathbf{b}_i^0, \mathbf{x}_i^0)$, $(\mathbf{b}_i^1, \mathbf{x}_i^1)$, $(\mathbf{b}_i^2, \mathbf{x}_i^2)$, $\ldots$. Since agents

[4]For first-price payment, the optimal setting will be $b_i = c_i + \epsilon$ because $p_i = b_i$. For critical-value-based payment, setting b_i to any value not less than $c_i + \epsilon$ gives a_i the same payment $p_i = c_i$.

TABLE I
KEY PREDECESSOR AND CRITICAL VALUE EXAMPLE WITH $\mathbf{q} = (3, 2, 2, 2, 2)$

Bidder (a_i)	Demand vector ($\mathbf{s}_i$)	Bid (b_i)	BRF ($b_i / \sum_k s_i^k$)	x_i	Key predecessor	Critical value (c_i)
a_1	$(1, 0, 1, 0, 0)$	50	25.00	1	a_1	-
a_2	$(0, 0, 0, 2, 1)$	70	23.33	1	a_2	-
a_3	$(0, 1, 0, 1, 2)$	93	23.25	0	a_2	93.33
a_4	$(2, 1, 1, 0, 0)$	90	22.50	1	a_4	-
a_5	$(1, 0, 2, 1, 0)$	63	15.75	0	a_1	100

can only raise their bids and no agent places a bid higher than its valuation, there exists some integer t_i such that $\mathbf{b}_i^t = (b_1, b_2, \ldots, b_n)$ no longer changes when $t \geq t_i$. Although the values of t_i may be different for different a_i's, eventually $\mathbf{b}_i^t$ will stabilize for all a_i's. After that, the rank of bid requests is finalized. Without loss of generality, assume that $(\mathbf{s}_j, b_j) \prec (\mathbf{s}_{j+1}, b_{j+1})$ for all $1 \leq j < n$. We already know that no bidder a_j has the incentive to cheat on x_j. It follows that the value of x_1 will eventually stabilize. Because of this, the value of x_2 will eventually stabilize, and so on. Therefore, ξ_i must be finite. ■

When the protocol stabilizes, the outcome (i.e., the collective settings of b_i's and x_i's) is *correct* if it meets the capacity constraint and conforms to the BRF-based winner-determination rule defined below.

Definition 3 (BRF-based Winner Determination Rule): Let $\prec$ be a total order defined by a BRF on bid requests $\mathcal{B} = \{(\mathbf{s}_i, b_i(\mathbf{s}_i))\}_{i=1}^n$. An outcome of the CA $X = (x_1, x_2, \ldots, x_n)$ conforms to the BRF-based winner determination rule if for every x_i, where $1 \leq i \leq n$, $x_i = 1$ only if

$$s_i^k + \sum_{(\mathbf{s}_j, b_j) \prec (\mathbf{s}_i, b_i)} \left(x_j \cdot s_j^k\right) \leq q_k \text{ for all } s_i^k \neq 0. \tag{17}$$

More than one outcomes may conform to the BRF-based winner determination rule. For example, if two bidders can both win regardless of their relative ranks, then we can yield another confirming outcome by switching their ranking places in a known confirming outcome.

Lemma 1: Let a_j be a_i's key predecessor returned by Algorithm 3. If $j \neq i$ and a_i increases b_i to some value that makes $(\mathbf{s}_i, b_i)$ outrank $(\mathbf{s}_j, b_j)$, we have

$$\sum_{(\mathbf{s}_l, b_l) \prec (\mathbf{s}_i, b_i)} \left(x_l \cdot s_l^k\right) + s_i^k \leq q_k \text{ for all } s_i^k \neq 0. \tag{18}$$

Proof: Since $j \neq i$, Algorithm 3 returns its value in Line 9. This implies that $x_j = 1$, $(\mathbf{s}_j, b_j) \prec (\mathbf{s}_i, b_i)$,

$$\sum_{(\mathbf{s}_l, b_l) \prec (\mathbf{s}_j, b_j)} \left(x_l \cdot s_l^k\right) + s_i^k \leq q_k \text{ for all } s_i^k \neq 0,$$

and

$$\sum_{(\mathbf{s}_l, b_l) \prec (\mathbf{s}_j, b_j)} \left(x_l \cdot s_l^k\right) + x_j \cdot s_j^k + s_i^k > q_k \text{ for some } s_i^k \neq 0.$$

Therefore, if a_i increases b_i to some value to make $(\mathbf{s}_i, b_i)$ outrank $(\mathbf{s}_j, b_j)$, then (18) holds. ■

Theorem 2: The outcome of Algorithm 2 conforms to the BRF-based winner determination rule. That is, for every x_i, where $1 \leq i \leq n$, $x_i = 1$ only if (17) holds.

Proof: There are only two conditions for a_i to set x_i to 1 in Algorithm 2: either a_i's key predecessor is a_i or a_i's key predecessor is some $a_l \neq a_i$ yet a_i has increased b_i to some value that makes $(\mathbf{s}_i, b_i)$ outrank $(\mathbf{s}_l, b_l)$. For the former case, Algorithm 3 ensures that that

$$\sum_{(\mathbf{s}_j, b_j) \prec (\mathbf{s}_i, b_i)} \left(x_j \cdot s_j^k\right) + s_i^k \leq q_k \text{ for all } s_i^k \neq 0. \tag{19}$$

For the latter case, we also have (19) by Lemma 1. We thus have the proof. ■

Theorem 3: Algorithm 2 meets the resource capacity constraint specified in (2), i.e.,

$$\sum_{i=1}^n \left(x_i \cdot s_i^k\right) \leq q_k \text{ for all } k = 1, \ldots, m.$$

Proof: For each k, $1 \leq k \leq m$, let $(\mathbf{s}_i, b_i)$ be the bid request that is of the lowest rank in $\mathcal{B}$ such that $x_i = 1$ and $s_i^k \neq 0$. This implies for all request $(\mathbf{s}_j, b_j)$ that ranks lower than $(\mathbf{s}_i, b_i)$, we have either $s_j^k = 0$ or $x_j = 0$. Therefore,

$$\sum_{j=1}^n \left(x_j \cdot s_j^k\right) = s_i^k + \sum_{(\mathbf{s}_j, b_j) \prec (\mathbf{s}_i, b_i)} \left(x_j \cdot s_j^k\right).$$

Because $x_i = 1$, we know that (17) holds by Theorem 2. We thus have the proof. ■

Even though Algorithm 2 converges and is correct, the outcome may deviate from that obtained by Algorithm 1 with the same BRF. Let $\mathbf{b} = (b_1, b_2, \ldots, b_n)$ and $\mathbf{b}' = (b_1', b_2', \ldots, b_n')$ be two vectors that represent the final bids found by Algorithms 2 and 1, respectively. Because $0 \leq b_i \leq v_i$ for all $b_i \in \mathbf{b}$ and $b_i' = v_i$ for all $b_i' \in \mathbf{b}'$, we have $\mathbf{b} \leq \mathbf{b}'$. Consequently, the ranks of bid requests in these two algorithms can be different. Table II shows an example where $\mathbf{b}' = \{9, 13, 10\}$. Here we show a particular transition sequence when running Algorithm 2, where a_2 stops raising b_2 and turns x_2 to 0 as soon as it realizes that $c_2 = 14 > v_2$ due to the setting of $b_1 = 7$. The knowledge of $x_2 = 0$ in turn causes a_3 to stop at $b_3 = 3$ with the declaration of $x_3 = 1$. As a result, the a_3's request is ranked higher than a_1's in Algorithm 1 but lower than it in Algorithms 2.

Despite of the difference in ranks, we shall prove that Algorithm 2 is *consistent* in the sense that it identifies the same set of winners as Algorithm 1 using the same BRF.

TABLE II
AN EXAMPLE WITH $\mathbf{q} = (1, 1)$ AND BRF $=(b_i / \sum_k s_i^k)$

Bidder (a_i)	Demand vector ($\mathbf{s}_i$)	Valuation (v_i)	b'_i	b_i
a_1	$(1, 0)$	9	9	7
a_2	$(1, 1)$	13	13	8
a_3	$(0, 1)$	10	10	3

Theorem 4: Given $\mathbf{b}$ and $\mathbf{b}'$ as defined above, let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{x}' = (x'_1, x'_2, \ldots, x'_n)$ be two vectors that represent the auction results of Algorithms 2 and 1, respectively, where $x_i = 1$ (resp. $x'_i = 1$) means $(\mathbf{s}_i, b_i)$ (resp. $(\mathbf{s}_i, b'_i)$) is granted by Algorithm 2 (resp. Algorithm 1). If these two algorithms use the same BRF, then $x_i = x'_i$ for all i, $1 \le i \le n$.

Proof: Without loss of generality, we assume that $\mathcal{B}' = \{(\mathbf{s}_i, b'_i)\}_{i=1}^n$ is sorted in an ascending order of ranks. That is, $(\mathbf{s}_i, b'_i = v_i)$ is ranked i-th in $\mathcal{B}'$. However, $(\mathbf{s}_i, b_i)$ is not necessarily ranked as the i-th request in $\mathcal{B} = \{(\mathbf{s}_i, b_i)\}_{i=1}^n$. Let $\mathbf{c} = (c_1, c_2, \ldots, c_n)$ and $\mathbf{c}' = (c'_1, c'_2, \ldots, c'_n)$ be two vectors that represent the critical values for $(\mathbf{b}, \mathbf{x})$ and $(\mathbf{b}', \mathbf{x}')$, respectively. We shall prove the theorem by induction on i.

Basis Step: Consider the case of $i = 1$. $(\mathbf{s}_1, v_1)$ outranks all other requests in $\mathcal{B}'$. If $x'_1 = 0$, which implies $s_1^l > q_l$ for some l, $1 \le l \le m$, then x_1 must be 0 as well. If $x'_1 = 1$, then $s_1^l \le q_l$ for all l, $1 \le l \le m$, and $b'_1 = v_1 > c'_1$. The fact that $b_j \le b'_j = v_j$ for all $j \ne 1$ implies that $c_1 < c'_1$. By Algorithms 2, $x_1 = 1$ because $(\mathbf{s}_1, b_1)$ can outrank all other requests in $\mathcal{B}$ when b_1 is in $[c_1, v_1]$[5].

Inductive Step: Assume that $x_j = x'_j$ for all $1 \le j \le k$, where k is an arbitrary number in $[1, n-1]$. Consider the case of $k' = k + 1$. If $x'_{k'} = 1$, then a necessary condition is

$$\sum_{j=1}^{k'} (x'_j \cdot s_j^l) \le q_l \text{ for all } l, 1 \le l \le m. \tag{20}$$

Because $(\mathbf{s}_{k'}, v_{k'})$ ranks k'-th in $\mathcal{B}'$ and $b_j \le b'_j = v_j$ for all $j > k'$, $(\mathbf{s}_{k'}, b_{k'})$ in $\mathcal{B}$ cannot rank lower than any request in $\{(\mathbf{s}_j, b_j)\}_{j=k'+1}^n$ when $b_{k'}$ is set to a value in $[c_{k'}, v_{k'}]$. That means $(\mathbf{s}_{k'}, b_{k'})$ can rank k'-th or higher in $\mathcal{B}$ when running Algorithms 2, and all requests in $\mathcal{B}$ that outrank $(\mathbf{s}_{k'}, b_{k'})$ under that condition, if any, are those that outrank $(\mathbf{s}_{k'}, b'_{k'})$ in $\mathcal{B}'$. By the inductive hypothesis and (20), we have $x_{k'} = 1$[6]. On the other hand, if $x'_{k'} = 0$, then (20) does not hold. If it is because $s_{k'}^l > q_l$ for some l, $1 \le l \le m$, then $x_{k'}$ must be 0 as well. Otherwise, there must exist $\bar{l} \in \{1, \ldots, m\}$ such that

$$s_{k'}^{\bar{l}} + \sum_{j=1}^{k} (x'_j \cdot s_j^{\bar{l}}) > q_{\bar{l}}. \tag{21}$$

Consider the set $\mathcal{B}_{k'} = \{(\mathbf{s}_j, b_j) | s_j^{\bar{l}} > 0 \land x'_j = 1 \land j < k'\}$. By the inductive hypothesis, we have $x_j = x'_j$ for any request $(\mathbf{s}_j, b_j) \in \mathcal{B}_{k'}$. Therefore, (21) is equivalent to

$$s_{k'}^{\bar{l}} + \sum_{(\mathbf{s}_j, b_j) \in \mathcal{B}_{k'}} s_j^{\bar{l}} > q_{\bar{l}}. \tag{22}$$

If all requests in $\mathcal{B}_{k'}$ still outrank $(\mathbf{s}_{k'}, b_{k'})$ in $\mathcal{B}$, $x_{k'}$ must be 0 by (22). Otherwise, let $\mathcal{C}$ denote the set of all requests in $\mathcal{B}_{k'}$ that rank lower than $(\mathbf{s}_{k'}, b_{k'})$ in $\mathcal{B}$ and let $(\mathbf{s}_j, b_j) \in \mathcal{C}$ be the one that ranks the lowest in $\mathcal{C}$. Let $\mathcal{O}_j = \{(\mathbf{s}_p, b_p) | (\mathbf{s}_p, b_p) \in \mathcal{B}_{k'} \land (\mathbf{s}_p, b_p) \prec (\mathbf{s}_j, b_j)\}$ be the set of all requests in $\mathcal{B}_{k'}$ that outrank $(\mathbf{s}_j, b_j)$ in $\mathcal{B}$. Because $x_j = 1$ and $x_p = 1$ for every $(\mathbf{s}_p, b_p) \in \mathcal{O}_j$ by the inductive hypothesis, we have

$$s_j^{\bar{l}} + x_{k'} \times s_{k'}^{\bar{l}} + \sum_{(\mathbf{s}_p, b_p) \in \mathcal{O}_j} s_p^{\bar{l}} \le q_{\bar{l}}. \tag{23}$$

Because $\mathcal{O}_j \cup \{(\mathbf{s}_j, b_j)\} = \mathcal{B}_{k'}$, (23) can be rewritten as

$$x_{k'} \times s_{k'}^{\bar{l}} + \sum_{(\mathbf{s}_p, b_p) \in \mathcal{B}_{k'}} s_p^{\bar{l}} \le q_{\bar{l}}. \tag{24}$$

The only way to make both (22) and (24) true is $x_{k'} = 0$. ■

D. Critical-Value-Based Payment

In the proposed approach, each winner a_i independently figures out how much she should pay for the auction. The problem is trivial for first-price auctions ($p_i = b_i$). For the critical-value-based payment, the problem is to identify a_i's *key successor*.

Definition 4 (Key Successor): Let a_i be a winner. Bidder $a_k \ne a_i$ is a_i's key successor if $(\mathbf{s}_k, b_k)$ is the request that has the highest rank among all non-winning requests which would be granted if $(\mathbf{s}_i, b_i)$ were not present. If there is no such request, a_i's key successor is defined to be a_i itself.

If a_i's key successor is $a_k \ne a_i$, a_i's payment p_i is the minimal value of b_i that makes the rank of $(\mathbf{s}_i, b_i)$ equal to or higher than that of $(\mathbf{s}_k, b_k)$. If a_i's key successor is a_i itself, then $p_i = 0$.

Let us revisit the example shown in Table I. Table III shows the key successor and payment, respectively, for each winner. The key successor for a_2 is a_3 because $(\mathbf{s}_3, b_3)$ would be granted if $(\mathbf{s}_2, b_2)$ were not present. For $(\mathbf{s}_2, b_2)$ to outrank $(\mathbf{s}_3, b_3)$, b_2 should be greater than

$$c_2 = \frac{b_3 (\sum_{k=1}^m s_2^k)^\alpha}{(\sum_{k=1}^m s_3^k)^\alpha} = \frac{93 \times 3}{4} = 69.75. \tag{25}$$

Therefore, a_2's payment p_2 is 69.75. The key successor for a_1 is a_1 itself because neither a_3's nor a_5's request would be granted if a_1's request were not present. Therefore, a_1's payment is 0.

The relationship between key predecessor and key successor is not symmetric: a_1 is a_5's key predecessor in Table I but a_5 is not a_1's key successor here.

Algorithm 4 details how to identify the key successor for a_i.

[5]Note this does not mean that $(\mathbf{s}_1, b_1)$ has to rank first in $\mathcal{B}$ to win the auction.

[6]This does not mean that $(\mathbf{s}_{k'}, b_{k'})$ has to rank k'-th or higher in $\mathcal{B}$ to win the auction.

TABLE III
KEY SUCCESSOR AND PAYMENT EXAMPLE WITH $\mathbf{q} = (3, 2, 2, 2, 2)$

Bidder (a_i)	Bid (b_i)	Demand vector ($\mathbf{s}_i$)	BRF ($b_i / \sum_k s_i^k$)	Result (x_i)	Key successor	Payment (p_i)
a_1	50	$(1, 0, 1, 0, 0)$	25.00	1	a_1	0
a_2	70	$(0, 0, 0, 2, 1)$	23.33	1	a_3	69.75
a_3	93	$(0, 1, 0, 1, 2)$	23.25	0	-	0
a_4	90	$(2, 1, 1, 0, 0)$	22.50	1	a_4	0
a_5	63	$(1, 0, 2, 1, 0)$	15.75	0	-	0

Algorithm 4 Procedure *key_successor*$(i, \mathcal{B}, \mathbf{x})$

$\mathcal{C} \leftarrow \{(\mathbf{s}_j, b_j) | (\mathbf{s}_j, b_j) \prec (\mathbf{s}_i, b_i)\}$
total_unit$[k] \leftarrow 0$ for all $k \in \{1, \ldots, m\}$
for all $(\mathbf{s}_j, b_j) \in \mathcal{C}$ such that $x_j = 1$ **do**
  for all $k \in \{1, \ldots, m\}$ **do**
    total_unit$[k] \leftarrow$ *total_unit*$[k] + s_j^k$
  end for
end for
$\mathcal{D} \leftarrow \{(\mathbf{s}_j, b_j) | (\mathbf{s}_i, b_i) \prec (\mathbf{s}_j, b_j)\}$
while $\mathcal{D} \neq \emptyset$ **do**
  Let $(\mathbf{s}_j, b_j)$ be the request that has the highest rank in $\mathcal{D}$
  if $x_j = 1$ **then**
    total_unit$[k] =$ *total_unit*$[k] + s_j^k$ for all k
  else ▷ a conflicting, non-winning request
    if *total_unit*$[k] + s_j^k \leq q_k$ for all $k, s_j^k > 0$ **then**
      return j
    end if
  end if
  $\mathcal{D} \leftarrow \mathcal{D} \setminus \{(\mathbf{s}_j, b_j)\}$
end while
return i

IV. SIMULATION RESULTS

We conducted simulations to evaluate the performance of the proposed scheme in terms of total winning bid, total payment, and the time to convergence. We assume n bidders and m types of items. For supply side, the number of identical instances for any resource type r_j, q_j, is distributed over the set of integers in the range $[1, q_{\max}]$, where $q_{\max}$ is a fixed number. For demand side, the probability that any agent a_i requests any resource type r_j is a tunable parameter p_s. If a_i does request r_j, s_i^j is distributed over the set of integers in the range $[1, q_j]$. Let $v_{i,j}$ be a_i's valuation on one instance of resource type r_j. The set $\{v_{i,j}\}_{i=1}^n$ follows a Gaussian distribution truncated at 0 and 50 with mean μ_j, where μ_j is a random variable uniformly distributed over the range $[10, 20]$. Agent a_i's evaluation on $\mathbf{s}_i$ is set to

$$v_i(\mathbf{s}_i) = \sum_{j=1}^{m} (s_i^j \times v_{i,j}).$$

As mentioned, the proposed approach does not demand particular initial value of each x_i. We thus tested three possible settings for the initial values of x_i's: all 1's, all 0's, and randomly selected 1's and 0's. For each possible setting, we generated 100 test data and performed 10 trials for each data. Each result is an average over these 1000 trials.

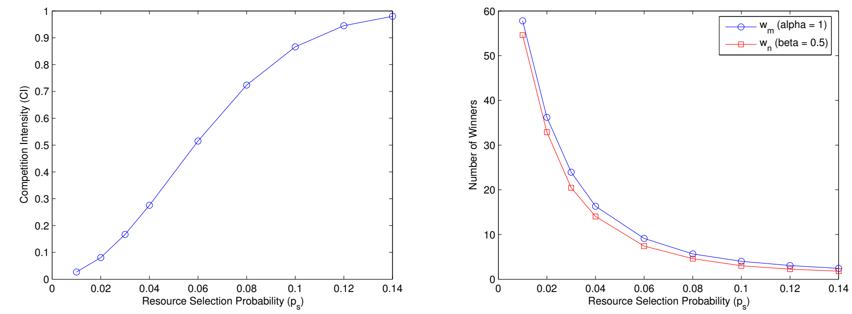
A factor that affects the performance metrics is *competition intensity* (CI), the ratio of the total number of edges in the conflict graph to the maximum (i.e., $n(n-1)/2$). We fixed m, n, and $q_{\max}$, and varied the value of p_s from 0.01 to 0.14 to adjust CI. Fig. 2a shows how CI changes with increasing p_s.

We used two BRFs for our simulations: w_m with $\alpha = 1$ and w_n with $\beta = 0.5$. When CI increases, the numbers of winners identified with both BRFs decrease, as Fig. 2b shows. Here w_m slightly outperformed w_n due to its consideration of the number of instances in the function definition. For a specific BRF, the centralized greedy approach and the proposed decentralized approach yielded exactly the same set of winners.

Figure 3 shows how total winning bid changes with respect to p_s. For both BRFs, the centralized approach outperformed the decentralized counterpart, and the performance of the decentralized approach was not affected by the initial setting of x_i's. Although there were more winners with $p_s = 0.01$ than with $p_s \geq 0.02$ in both BRFs, the total winning bid with $p_s = 0.01$ was not always higher than that with $p_s \geq 0.02$. The reason is that although there were more winners with $p_s = 0.01$ than with $p_s \geq 0.02$, each winner with $p_s = 0.01$ generally requested fewer resource instances than that with $p_s \geq 0.02$. Because each agent's bid was roughly in proportional to the number of requested instances, we obtained the highest total winning bid with $p_s = 0.02$ or $p_s = 0.03$. When p_s increased further, the total winning bid decreased because the number of winners decreased significantly as Fig. 2b indicates.

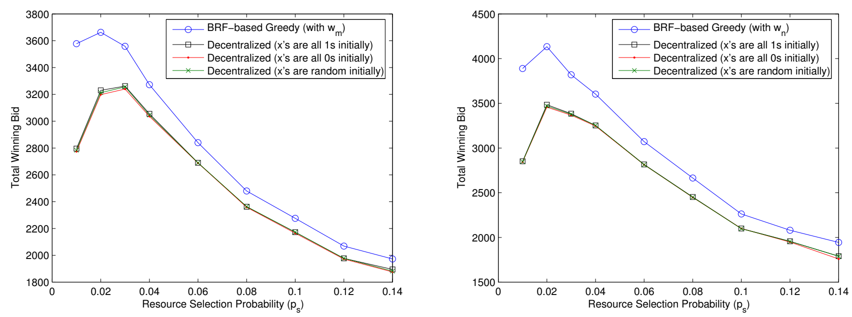
Figure 4 show how the total payment changes with increasing CI. Here all approaches exhibit behaviors similar to the result of total winning bid. However, the total payment is always lower then the total winning bid under any circumstance. The performance gap between the centralized and the proposed decentralized approach becomes smaller when CI is smaller. This trend is also similar to that exhibited in Fig. 3.

We studied the convergence time of the decentralized approach by measuring the total number of moves taken by all bidders before reaching the final result. We did not count the initial setting and broadcast of bid requests; only changes of bid requests (sendings of updates) count. Fig. 5 shows the results, which clearly depend on the initial settings of x_i's. Generally speaking, the all-1s initial setting demanded fewer moves than the all-0s initial setting when there were more winners (i.e., when p_s is small). On the other hand, the all-0s initial setting outperformed the all-0s initial setting when there were few winners (i.e., when p_s is large). The random setting generally lies between these two extremes, and would be the best choice if we do not know the CI value beforehand.



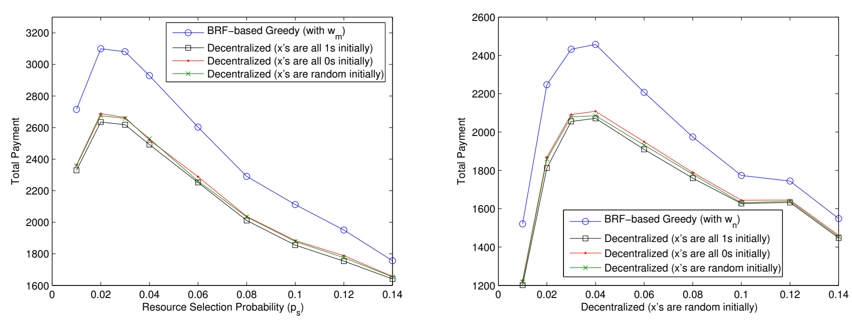
(a) (b)

Fig. 2. Results with $q_{\max} = 5$, $n = 100$, and $m = 200$. (a) Competition intensity (CI) versus p_s. (b) Number of winners versus p_s.



(a) (b)

Fig. 3. Average total winning bid with (a) w_m ($\alpha = 1$) and (b) w_n ($\beta = 0.5$). ($q_{\max} = 5$, $n = 100$, and $m = 200$)



(a) (b)

Fig. 4. Average payment with (a) w_m ($\alpha = 1$) and (b) w_n ($\beta = 0.5$). ($q_{\max} = 5$, $n = 100$, and $m = 200$)

Regardless of the initial setting of x_i's, on average each agent took fewer than two moves before stability.

V. Conclusions

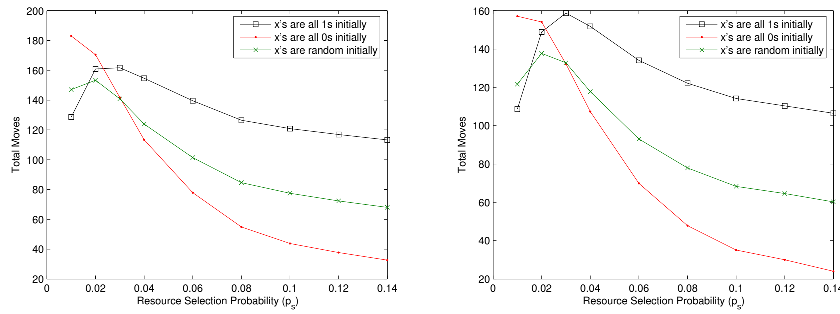
We have proposed a decentralized BRF-based winner determination and payment scheme. It allows bidders to locally determine their bid and willingness to win by identifying their key predecessors. By exchanging that information with other competitors, other bidders can take moves so as to reach a consensus. Winners determines their payments by identifying their respective key successors. We have proved that the proposed approach eventually stabilizes and is correct in the sense that the result meets the capacity constraint and conforms to the BRF-based winner-determination rule. We also proved that the proposed approach is consistent with the centralized counterpart using the same BRF because both approaches identify the same set of winners. Simulation results confirms the correctness and consistency of the proposed approach at the cost of lower total winning bid and payment.

References

[1] C. Xu, L. Song, Z. Han, Q. Zhao, X. Wang, X. Cheng, and B. Jiao, "Efficiency resource allocation for device-to-device underlay communication systems: A reverse iterative combinatorial auction based approach," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 9, pp. 348–358, Sep. 2013.

[2] F. Wu, T. Zhang, C. Qiao, and G. Chen, "A strategy-proof auction mechanism for adaptive-width channel allocation in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 10, pp. 2678–2689, Oct. 2016.

[3] L. Mashayekhy, M. M. Nejad, and D. Grosu, "A PTAS mechanism for provisioning and allocation of heterogeneous cloud resources," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 9, pp. 2386–2399, Sep. 2015.



(a) (b)

Fig. 5. Average number of moves with (a) w_m ($\alpha = 1$) and (b) w_n ($\beta = 0.5$). ($q_{\max} = 5$, $n = 100$, and $m = 200$)

[4] H. Zhang, H. Jiang, B. Li, F. Liu, A. V. Vasilakos, and J. Liu, "A framework for truthful online auctions in cloud computing with heterogeneous user demands," *IEEE Trans. Comput.*, vol. 65, no. 3, pp. 805–818, Mar. 2016.

[5] L. Mashayekhy, M. M. Nejad, D. Grosu, and A. V. Vasilakos, "An online mechanism for resource allocation and pricing in clouds," *IEEE Trans. Comput.*, vol. 65, no. 4, pp. 1172–1184, Apr. 2016.

[6] A.-L. Jin, W. Song, P. Wang, D. Niyato, and P. Ju, "Auction mechanisms toward efficient resource sharing for cloudlets in mobile cloud computing," *IEEE Trans. on Service Computing*, vol. 9, no. 6, pp. 895–909, 2016.

[7] K. Li, C. Liu, K. Li, and A. Y. Zomaya, "A framework of price bidding configurations for resource usage in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, no. 8, pp. 2168–2181, Aug. 2016.

[8] H.-L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Trans. on Robotics*, vol. 25, no. 4, pp. 912–926, Aug. 2009.

[9] K. Zhang, E. G. Collins, and D. Shi, "Centralized and distributed task allocation in multi-robot teams via a stochastic clustering auction," *ACM Trans. on Autonomous and Adaptive Systems*, vol. 7, no. 2, Jul. 2012.

[10] J. Murillo, V. M. noz, B. López, and D. Busquets, "A fair mechanism for recurrent multi-unit auctions," in *Lecture Notes in Artificial Intelligence 5244*, R. Bergmann, G. Lindemann, S. Kirn, and M. Pechoucek, Eds. Springer-Verlag, 2008, pp. 147–158.

[11] S. de Vries and R. V. Vohra, "Combinatorial auctions: A survey," *INFORMS Journal on Computing*, vol. 15, no. 3, pp. 284–309, 2003.

[12] P. Cramton, Y. Shoham, and R. Steinberg, *Combinatorial Auctions*. MIT Press, 2006.

[13] W. Vickrey, "Counterspeculation, auctions and competitive sealed tenders," *The Journal of Finance*, vol. 16, no. 1, pp. 8–37, Mar. 1961.

[14] E. Clarke, "Multipart pricing of public goods," *Public Choice*, vol. 11, no. 1, pp. 17–33, 1971.

[15] T. Groves, "Incentives in teams," *Econometrica*, vol. 41, no. 4, pp. 617–631, Jul. 1973.

[16] D. Lehmann, L. I. O'Callaghan, and Y. Shoham, "Truth revelation in approximately efficient combinatorial auctions," *J. Assoc. Comput. Mach.*, vol. 49, no. 5, pp. 577–602, Sep. 2002.

[17] P. J. Brewer, "Decentralized computation procurement and computational robustness in a smart market," *Economic Theory*, vol. 13, no. 1, pp. 41–92, Jan. 1999.

[18] D. Hausheer and B. Stiller, "PeerMart: The technology for a distributed auction-based market for peer-to-peer services," in *Proc. IEEE ICC*, May 2005.

[19] T. Sandholm, "Algorithm for optimal winner determination in combinatorial auctions," *Artificial Intelligence*, vol. 135, no. 1-2, pp. 1–54, Feb. 2002.

[20] K. Leyton-Brown, Y. Shoham, and M. Tennenholtz, "An algorithm for multi-unit combinatorial auctions," in *Proc. AAAI/IAAI*, Jul. 2000, pp. 56–61.

[21] T. Sandholm, S. Suri, A. Gilpin, and D. Levine, "CABOB: A fast optimal algorithm for winner determination in combinatorial auctions," *Management Science*, vol. 51, no. 3, Mar. 2005.

[22] B. Lehmann, D. Lehmann, and N. Nisan, "Combinatorial auctions with decreasing marginal utilities," *Games and Economic Behavior*, no. 2, pp. 270–296, May 2006.

[23] H. H. Hoos and C. Boutilier, "Solving combinatorial auctions using stochastic local search," in *Proc. AAAI/IAAI*, Jul. 2000, pp. 22–29.

[24] E. Zurel and N. Nisan, "An efficient approximate allocation algorithm for combinatorial auctions," in *Proc. of the Third ACM Conf. on Electronic Commerce*, Tampa, Florida, USA, Oct. 2001, pp. 125–136.

[25] N. Fukuta and T. Ito, "Fine-grained efficient resource allocation using approximated combinatorial auctions: A parallel greedy winner approximation for large-scale problems," *Web Intelligence and Agent Systems: An International Journal*, vol. 7, no. 1, pp. 43–63, 2009.

[26] V. Avasarala, H. Polavarapu, and T. Mullen, "An approximate algorithm for resource allocation using combinatorial auctions," in *Proc. IEEE/WIC/ACM Int'l Conf. on Intelligent Agent Technology*, Dec. 2006, pp. 571–578.

[27] M. Mito and S. Fujita, "On heuristics for solving winner determination problem in combinatorial auctions," *Journal of Heuristics*, vol. 51, no. 5, pp. 507–523, 2004.

[28] S. Sakai, M. Togasaki, and K. Yamazaki, "A note on greedy algorithms for the maximum weighted independent set problem," *Discrete Applied Mathematics*, vol. 126, pp. 313–322, 2003.

[29] J. Jia, Q. Zhang, Q. Zhang, and M. Liu, "Revenue generation for truthful spectrum auction in dynamic spectrum access," in *IEEE MobiHoc'09*, New Orleans, Louisiana, USA, May 2009.

[30] S. Zaman and D. Grosu, "Combinatorial auction-based allocation of virtual machine instances in clouds," *J. Parallel Distrib. Comput.*, vol. 73, no. 4, pp. 495–508, Apr. 2013.

[31] ——, "A combinatorial auction-based mechanism for dynamic VM provisioning and allocation in clouds," *IEEE Trans. on Cloud Computing*, vol. 1, no. 2, pp. 129–141, 2013.

[32] M. M. Nejad, L. Mashayekhy, and D. Grosu, "Truthful greedy mechanisms for dynamic virtual machine provisioning and allocation in clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 2, pp. 594–603, Feb. 2015.

[33] Y. M. Teo and M. Mihailescu, "A strategy-proof pricing scheme for multiple resource type allocations," in *Proc. 38th Int'l Conf. on Parallel Processing*, Vienna, Austria, Sep. 2009, pp. 172–179.

[34] E. Kutanoglu and S. D. Wu, "On combinatorial auction and lagrangean relaxation for distributed resource scheduling," *IIE Transactions*, vol. 31, no. 9, pp. 813–826, 1999.

[35] D. C. Parkes, "iBundle: An efficient ascending price bundle auction," in *Proc. 1st ACM Conf. on Electronic Commerce*, Nov. 1999, pp. 148–157.

[36] M. Esteva and J. Padget, "Auctions without auctioneers: Distributed auction protocols," in *Lecture Notes in Artificial Intelligence 1788*, A. Moukas, F. Ygge, and C. Sierra, Eds. Springer-Verlag, 2000, pp. 220–238.

[37] P. R. Lewis, P. Marrow, and X. Yao, "Resource allocation in decentralised computational systems: an evolutionary market-based approach," *Autonomous Agents and Multi-Agent Systems*, vol. 21, no. 2, pp. 143–171, 2010.

[38] M. P. Wellman, W. E. Walsh, P. R. Wurman, and J. K. MacKie-Mason, "Auction protocols for decentralized scheduling," *Games and Economic Behavior*, vol. 35, pp. 271–303, 2001.