Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2014, Article ID 845071, 10 pages
http://dx.doi.org/10.1155/2014/845071

Research Article

A Game Theoretical Approach for Solving Winner Determination Problems

Chen-Kun Tsung,[1] Hann-Jang Ho,[2] and Sing-Ling Lee[1]

[1] *Institute of Computer Science and Information Engineering, National Chung Cheng University, No. 168, Section 1, University Road, Min-Hsiung, Chia-yi 62102, Taiwan*
[2] *Department of Applied Digital Media, WuFeng University, No. 117, Section 2, Chiankuo Road, Min-Hsiung, Chia-yi 62153, Taiwan*

Correspondence should be addressed to Hann-Jang Ho; hannjangho@gmail.com

Received 14 September 2013; Revised 13 December 2013; Accepted 2 January 2014; Published 10 February 2014

Academic Editor: Manuel Ruiz-Galan

Determining the winners in combinatorial auctions to maximize the auctioneer's revenue is an NP-complete problem. Computing an optimal solution requires huge computation time in some instances. In this paper, we apply three concepts of the game theory to design an approximation algorithm: the stability of the Nash equilibrium, the self-learning of the evolutionary game, and the mistake making of the trembling hand assumption. According to our simulation results, the proposed algorithm produces near-optimal solutions in terms of the auctioneer's revenue. Moreover, reasonable computation time is another advantage of applying the proposed algorithm to the real-world services.

1. Introduction

The combinatorial auctions support the *complementarity* for bidding. The complementarity means that bidders can bid several goods with a price. The combinatorial auctions are useful in the instances with some considerations. For example, the Federal Communications Commission (FCC) used the combinatorial auction to sell electromagnetic spectra. Most bidders in the auction prefer to buy successive, rather than nonsuccessive, electromagnetic spectra. If the FCC used traditional auctions in selling a group of spectra to a single buyer, for example, the English auctions, the bidders may win some nonsuccessive spectra. Thus, the winners will drop out the auction because the expected utility is not achieved [1]. After the FCC used the combinatorial auction to sell the electromagnetic spectra, the combinatorial auctions are widely applied to solve the optimization problems, such as the study of economic performance [2] and task assignment [3].

Determining the winners in the combinatorial auctions is called as the winner determination problem (WDP) [4–9]. Solving the WDP to maximize the auctioneer's revenue is an NP-complete problem [5]. This indicates that the auctioneer requires huge time to solve some large-scale WDP instances. Currently, the approaches of solving the WDPs can be classified into three categories.

(1) The *optimal algorithms* can find the optimal solutions, but huge computation cost is required in some problem instances, such as [5–7].

(2) The *approximation algorithms* can find the winners rapidly, but they do not guarantee to find the optimal solutions for all kinds of instances, such as [8, 10, 11].

(3) The *restricted algorithms* can find the optimal solutions rapidly, but they can be used for the combinatorial auctions only with some restricted conditions, such as [6].

Although the optimal algorithms can find the solutions with maximum auctioneer's revenue, they require long CPU time for solving some instances. This approach is appropriate for small-scale problems. On the other hand, it is not suitable for the large-scale or on-line version because of its huge computation cost. The restricted algorithms are useful for the problems with special properties. This kind of approaches rapidly finds an optimal solution in a constrained problem. If the auctioneer applies the restricted algorithm to a general

problem, the solution may not reach the expected objective. Comparing to the optimal algorithms, the second approach can find the winners rapidly. Even if the optimal solution is not guaranteed, it reduces considerable computation cost to find the approximation solutions. Furthermore, the auctioneer must build the auction environment to fit the requirements of the restricted algorithm, but he does not need so by using the second approach. Therefore, a better approximation algorithm is more efficient and convenient for the auctioneers to solve WDPs.

In this paper, we formulate the WDP as a noncooperative game and apply some concepts of the game theory to design an approximation algorithm for solving the WDPs. We treat each auction good as a player who can determine the winner. In the proposed noncooperative game, all players use the auctioneer's revenue as their utility functions. Our proposed algorithm adopts the stability of Nash equilibria (NEs) that the obtained solutions are accepted by all players [12–14]. We utilize the evolutionary game to iteratively improve the solution quality and consider the hand trembling assumption to increase the solution diversity. According to our simulation results, our proposed algorithm achieves 97.81% revenue comparing to the optimal solutions. Moreover, our proposed algorithm is more efficient and convenient for the auctioneers to determine winners in combinatorial auctions.

2. Related Work

Finding the optimal solutions in the general WDPs has been proofed as an NP-complete problem [5]. The approaches of solving the general WDPs can be divided into two categories. The first one is finding the optimal winners, and the other one is the approximation algorithms. The optimal algorithms calculate the winners with maximum revenue, and the research goal is to decrease the computation cost. The approximation algorithms focus on increasing the solution quality under a reasonable running time.

Sandholm proposed the optimal algorithm, named the Combinatorial Auction Branch On Bids (CABOB), to find the winners with maximum auctioneer's revenue [7]. The Divide-and-Conquer is performed to partition the original WDP instance into some subproblems. Then, the CABOB applies the Linear Programming and the Branch-and-Bound approaches to eliminate unnecessary calculations. So, the CABOB decreases the computation cost. Although the computation efficiency is improved, the auctioneer still requires huge running time in some instances to obtain the optimal solution via adopting the CABOB. For the large-scale WDP instances, decreasing the computation cost is the major consideration of designing the optimal algorithms.

Another approach is to develop heuristic algorithms for increasing the solution quality. Avasarala et al. observed that the bid rank affects the solution quality [10]. They formulate the solutions in the bit string format that each element is a bid, and the genetic algorithm (GA) is applied to solve the WDP. When generating the initial population, each gene is given a rank as the evaluation sequence. Next, the crossover and mutation operators are applied to generate various solutions. After some iterations, the solution quality is improved. However, the parameter setting is a major problem for applying GAs to solve WDPs, such as the population size, crossover rate, and mutation rate. Thus, the auctioneers require some preprocesses to evaluate the optimal parameter settings.

Hoos and Boutilier applied the stochastic local search to design the Casanova [8]. The Casanova evaluates the feasible solutions as many as possible. Furthermore, the Casanova uses the revenue-per-item (RPI) of each bid to determine the evaluation order and keeps track of the age for each bid. The evaluation order affects the solution quality, and many studies have proposed heuristics to determine the order [15]. Generating solutions rapidly is the advantage of the Casanova, but increasing the solution quality is the major implementation issue.

The optimal algorithms and the approximation algorithms have different advantages in solving WDPs. The advantage of one approach is the drawback of another approach. In this paper, we propose an algorithm which combines the advantages of above approaches to maximize auctioneer's revenue and minimize the computation cost. The proposed algorithm can compute the solution rapidly and improve the solution quality iteratively. Thus, our proposed algorithm is valuable for the real-world implementations.

3. Problem Model

3.1. Winner Determination Problem. The WDP consists of a bid set $B = \{B_1, B_2, \ldots, B_{|B|}\}$ and a good set $G = \{g_1, g_2, \ldots, g_{|G|}\}$. Each bid B_i includes a bundle $b_i \subseteq G$ and a bid price $p_i \in \mathfrak{R}$. The bidder i pays p_i to buy all goods claimed in b_i if i is declared as a winner. Assume that a bidder proposes only one bid. The auctioneer's objective is to compute an assignment $X = (x_1, x_2, \ldots, x_{|B|})$, $\forall x_i \in \{0, 1\}$ for all bidders to maximize the revenue. When $x_i = 1$, the bidder i is a winner, and he can buy the bundle b_i by paying p_i and can buy nothing as $x_i = 0$. Therefore, the WDP can be formulated as an integer linear program shown as follows:

$$\max \quad \sum_{i=1}^{|B|} p_i x_i \tag{1}$$

$$\text{s.t.} \quad \sum_{i \mid j \in b_i} x_i \leq 1, \quad \forall j = 1, 2, \ldots, |G| \tag{2}$$

$$x_i \in \{0, 1\}. \tag{3}$$

The WDP has one constraint shown in (2). The candidates of winning the good g_j are the bidders i whose bundles include g_j, that is, $i \mid j \in b_i$. Equation (2) shows that g_j can be assigned to at most one bidder. In other words, only one candidate of g_j can be the winner.

We use an example illustrated in Table 1 to show the WDP. There are two solutions $X_1 = (1, 0, 1)$ with revenue $5 + 3 = 8$ and $X_2 = (0, 1, 0)$ with revenue 7. The constraint drawn in (2) indicates that each good can be sold to at most one bidder, so either bidder 1 or bidder 2 can be the winner of g_2. In

Table 1: An example of a WDP with three bids and three goods.

Bundle b_i	Bid price p_i
(g_1, g_2)	\$5
(g_2, g_3)	\$7
(g_3)	\$3

Table 2: The WDP game of Table 1. The player is a good, and the strategy indicates the candidates of winning the good.

Player	Strategy
1	$\{B_1, \phi\}$
2	$\{B_1, B_2, \phi\}$
3	$\{B_2, B_3, \phi\}$

this example, X_1 maximizes the auctioneer's revenue, where selling (g_1, g_2) to bidder 1 and selling (g_3) to bidder 3.

3.2. Proposed Game Model. We formulate the WDP as a noncooperative game that each player considers the common utility function. The elements of this game $\Gamma = (N, S, u)$ are listed as follows.

(i) A set of players $N = \{n_1, n_2, \ldots, n_{|G|}\}$. We treat each good as a player in Γ.

(ii) A set of all players' strategies $S = \{S_1, S_2, \ldots, S_{|G|}\}$. For each player n_j, the strategy set is $\{\phi, i \mid j \in b_i\}$. This means that n_j can determine the winner of g_j, where g_j must appear in the bundle of each winner candidate i. Moreover, g_j is unsold for the strategy ϕ.

(iii) A utility function u. Each player's utility function is the revenue of the auctioneer in the WDP, as shown in (1).

Given a union of all players' actions $s = (s_1, s_2, \ldots, s_{|G|})$, the outcome of Γ is a winner set of the WDP. Each action s_j in s points out the winner of g_j or it is unsold for ϕ. Therefore, we can transform the solution from Γ to the WDP. Considering a WDP instance, $s = (s_1, s_2, \ldots, s_{|G|})$ and $X = (x_1, x_2, \ldots, x_{|B|})$ are the solutions of Γ and the WDP, respectively. Supposing $s_j = i$ in s, we have $x_i = 1$ in X. Thus, we have $u(s) = \sum_{i=1}^{|B|} p_i x_i$, where $x_i = 1$ means that the bundle b_i is sold, that is, $\exists j$, s.t. $s_j = i$, $\forall 1 \le i \le |G|$.

We use the example shown in Table 1 to explain Γ. Each good is a player, so the game consists of three players. The strategy is the set of the winner candidates. Both bidders 1 and 2 bid at good 2, so player 2 can claim bidder 1 or bidder 2 as the winner. Notice that each good is unnecessary to be sold, so ϕ is a feasible strategy for each player. In Table 2, we have two solutions $s' = (1, 1, 3)$ and $s'' = (\phi, 2, 2)$ with revenue 8 and 7, respectively.

It is reasonable to treat each bundle, rather than a good, as a player. We discuss the rationality of regarding a good as a player from two aspects: the computation complexity and the property of the WDP. In the real-world applications, the number of bidders is greater than the number of the goods, and we have that $|B|$ is much larger than $|G|$. Computing an NE solution in the game with more players is more complex than that with fewer players. Thus, treating each good as a player is more appropriate than formulating the bundle as a player in terms of the computation efficiency consideration. On the other hand, (2) specifies that each good can be sold to at most one bidder or unsold. Regarding a good as a player completely satisfies the meaning that claimed in (2). Treating a bundle as a player also can reach the restriction stated in (2), but the auctioneer must maintain the information of indicating which good is sold. In summary, formulating the good as a player in a game is appropriate for the computation efficiency and matches the property of the WDP.

Table 3: The classical normal-form game: the prisoner's dilemma. Each prison has two choices: confess and silent. If they both confess, each one serves for one year, and two years as they remain silent. If one of them confesses, the prison who confesses does not need to serve, and the other one get a sentence of three years.

Player 1	Player 2	
	Confesses	Silent
Confesses	$(-1, -1)$	$(0, -3)$
Silent	$(-3, 0)$	$(-2, -2)$

4. Proposed Method

4.1. Finding Nash Equilibrium in the Normal-Form Game. Before introducing our approach, we first present the idea of finding a pure NE in a normal-form game. This is the foundation of the proposed algorithm. As shown in Table 3, the classical prisoner's dilemma includes two players. Given an initial solution, we will seek an NE outcome from this solution. Supposing the initial solution (silent, silent) and each player receives two-year serving. Because each player only cares about the self-utility, Player 1 observes that performing "confess" is more beneficial than "silent" under (silent, silent). Thus, Player 1 chooses "confess" and the outcome becomes (confess, silent). Then, Player 2 is aware that the utility is decreased from -2 to -3, and the strategy "confess" is more beneficial than "silent" currently under (confess, silent). Therefore, Player 2 also moves from "silent" to "confess", and the outcome is (confess, confess). Eventually, both Player 1 and Player 2 accept the outcome (confess, confess). If any player deviates from the outcome, their utility will be decreased from -1 to -3. No unilateral deviation will occur under (confess, confess), so this solution is an NE.

4.2. Nash Equilibrium Search Algorithm. In above prisoner's dilemma, we have the following procedure of seeking an NE. Consider an initial solution $s = (s_1, s_2, \ldots, s_{|G|})$. Pick up a player n_j, and change their action from s_j to s'_j. The outcome becomes $s' = (s_1, \ldots, s'_j, \ldots, s_{|G|})$. If all players are satisfied with s', it is an NE. Otherwise, pick up another player and check the existence of the other auction with higher utility. Repeat this idea until all players simultaneously accept the outcome.

```
Result: the winner set s
(1) begin
(2)     generate a solution s at random
(3)     while not meet stop condition do
(4)         s' ← findNE(s)
(5)         if r(s') > r(s) then
(6)             update gvc
(7)             s ← s'
(8)         end
(9)         s ← longJump(s)
(10)    end
(11)    return s
(12) end
```

Algorithm 1: The main algorithm of NESA.

```
Data: s: the base solution
Result: s': the NE solution
(1) begin
(2)     for each good g_j in s do
(3)         s' ← s
(4)         remove the winner of g_j in s'
(5)         add feasible bids to s' except for g_j based on a
            given ranking function
(6)         if r(s') > r(s) then
(7)             update gvc
(8)             return findNE(s')
(9)         end
(10)    end
(11)    return s
(12) end
```

Algorithm 2: The procedure *findNE*(*s*).

We use the WDP example Γ drawn in Table 1 to explain the manner of finding an NE. Let the initial solution be $s = (\phi, 2, 2)$. The bidder 2 is the winner of g_2 and g_3, and the utility of each player is \$7. The player n_1 has two strategies: ϕ and 1. If n_1 chooses ϕ, the outcome is still s. If n_1 picks up another strategy (selling g_1 to bidder 1), bidder 2 will lose n_2 and n_3, and bidder 3 can be the new winner of g_3. The outcome moves to $s' = (1, 1, 3)$, and the utility is increased from \$7 to \$8. Performing the strategy 1 (selling the good to bidder 1) is more beneficial than the current action ϕ, so n_1 changes the action from ϕ to 1. The outcome moves from s to s'. In s', no player can gain more utility by performing the other strategy, so s' is an NE.

Based on the above principle of searching the NE, we propose the Nash equilibrium search approach (NESA) to compute the winner set in the WDP. Except for the idea of searching NEs, we also consider other two ideas of the game theory to design the NESA. We adopt the learning strategy from the evolutionary game to improve the solution quality. To increase the solution diversity, the hand trembling assumption is taken into account for simulating the mistake making behavior in the real-world. The NESA includes two major procedures. The first one, termed *findNE*(), is used to seek an equilibrium outcome. Another procedure is named by *longJump*(), and that is designed to increase the solution diversity.

The main algorithm of the NESA is shown in Algorithm 1. The auctioneer should provide the stop criterion of the NESA, for example, up to ten minutes. First, the NESA randomly generates a feasible solution s. The NESA invokes *findNE*() to calculate the equilibrium solution s' from s. We apply the revenue calculation function $r(s)$ to compute the revenue that the auctioneer can earn from s. If the NE solution s' is more beneficial then s for the auctioneer, s' survives to the next iteration. Simultaneously, we update the good victim count (gvc) information of each bid. The gvc is a count that indicates the importance of each bid in terms of the auctioneer's revenue. Supposing $r(s') > r(s)$, which means the auctioneer can gain more revenue from s', we increase the gvc of each good in s' by one. Thus, the good with higher gvc value implies that selling the good contributes more revenue for the auctioneer. In the last step, the NESA invokes the diversity increase procedure *longJump*() to produce the solution which is quite different to the NE. If the stop criterion is not satisfied, *findNE*() finds the NE from the solution produced by the *longJump*().

4.3. The Local Search. Finding the NE solutions is the objective of the local search procedure *findNE*(). The NE represents a steady status that each player will not receive more utility by the unilateral deviation [13, 14]. The player n_j chooses the winner from the candidates whose bundles include the good g_j. All players use the same utility function: $r(s)$ which is the auctioneer's revenue.

The procedure of the local search *findNE*() is illustrated in Algorithm 2. *findNE*() receives a base solution s and outputs an NE solution s'. First, *findNE*() changes the action of g_j to ϕ and then declares the possible candidates as winners to establish a new feasible solution s' by a given ranking function. If s' contributes more auctioneer's revenue than s, *findNE*() returns s'. Simultaneously, increase the gvc of each good in s' by one. If we cannot find another solution better than s, all players accept with s, and s is an NE. Therefore, *findNE*() returns the NE solution s.

Because we focus on the game without any special property, *findNE*() must consider some techniques to prune unnecessary searches. In line (5) of Algorithm 2, there are some strategies for determining the ranking of adding feasible bids. For example, the auctioneer can rank the feasible bids via the criteria: (1) the bid price, (2) the RPI, or (3) the random approach. The ranking function which includes a specific search direction can compute the new solution s' quickly. However, the solution quality may be restricted. If the auctioneer does not well investigate the properties of the instance, the random approach is more appropriate to the general cases than other heuristics. Therefore, we adopt the random process for ranking the feasible bids in this paper.

4.4. The Diversity Increase Procedure. The procedure *longJump*(), shown in Algorithm 3, is used for computing another base solution of *findNE*(). Given a base solution s,

Data: s: the base solution
Result: s': the diversity increased solution
(1) **begin**
(2) $s' \leftarrow s$
(3) **foreach** bid g_b in s' **do**
(4) **if** *the random value* > gvr(g_b) **then**
(5) remove g_b in s'
(6) **end**
(7) **end**
(8) add feasible bids to s' except for g_b with trembling at random based on a given ranking function
(9) **if** $r(s') > r(s)$ **then**
(10) update gvc
(11) **return** s'
(12) **else if** *trembling* **then**
(13) **return** s'
(14) **else**
(15) **return** s
(16) **end**
(17) **end**

Algorithm 3: The procedure *longJump*(s).

first, the good g_b is removed according to the probability gvr(g_b) that is called as the good victim rate (gvr). The gvr of good g_b is the gvc ratio that is normalized by the maximum gvc over all winners in the solution s:

$$\text{gvr}(g_b) = \frac{\text{gvc}_{g_b}}{\max_{\forall g_j \in s} \text{gvc}_{g_j}}. \quad (4)$$

Higher gvr(g_b) indicates that the good g_b contributes more auctioneer's revenue than other goods. Removing g_b from the solution is not beneficial for the auctioneer in expectation. So, the remove probability of g_b is inversely proportional to gvr(g_b). The remove process is shown from step 3 to step 7.

In step 8, we randomly declare some winners to s' to establish a new feasible solution which is identical to that we considered in line (5) of Algorithm 2. When choosing a winner, the hand trembling assumption is considered to increase the solution diversity [16]. The hand trembling assumption means that all players may make a mistake even if the mistake probability is very small. In other words, there is a small probability that we declare the other bidder as a winner.

If s' contributes more revenue than s, there is an NE solution with higher revenue that we have not found. *longJump*() updates the gvc of each element in s' and outputs s'. Otherwise, s is returned. Based on the hand trembling assumption, s' with less auctioneer's revenue still can be accepted with a small probability.

5. Simulation Result

We consider three metrics to evaluate the NESA: the revenue performance, the anytime performance, and the optimal solution comparison. The simulation results are compared with the general GA and the Casanova. The GA is outstanding in solving optimization problems, such as optimizing the complex network system [17] and solving the WDP [10]. The Casanova converges rapidly, so we combine *findNE*() with the Casanova, termed as the Hybrid, in the anytime performance evaluation. Because the NESA outperforms the GA and the Casanova, we only compare *findNE*() with the optimal solution. The objective of the optimal solution comparison is to measure the effects of different initial solutions.

Table 4: The average number of goods in each bundle.

	Decay	Random	Uniform
Large-scale model	2.01	100.9453	5
Small-scale model	1.9946	50.4188	5

Table 5: The parameter settings of the genetic algorithm.

Parameters	Settings
Population size	100
Crossover rate	0.9
Mutation rate	0.025

5.1. Simulation Environment. By referring to [4, 8, 10], we consider three bid models, and each instance includes m goods and n bids.

(1) *Decay*($m, n, 0.75$): (1) Randomly assign a good, (2) repeatedly add a good with the probability 0.75, which is the same as [4, 8], until a good is not added or the bundle includes all goods, (3) pick up a price between 0 and the number of goods in this bundle.

(2) *Random*(m, n): (1) Determine the number of goods from $\{1, 2, 3, \ldots, m\}$ at random, (2) randomly choose enough goods without replacement, (3) pick up a price from the uniform distribution on $[0, 1]$.

(3) *Uniform*($m, n, 5$): (1) Choose five goods from $\{1, 2, 3, \ldots, m\}$ without replacement, (2) pick up a random price from a uniform distribution on $[0, 1]$.

We consider the large-scale model with 2000 bids and 200 goods and the small-scale model with 1000 bids and 100 goods. Table 4 shows the average number of goods in a bundle. We generate five instances for each bid model and use the average values over ten runs for each algorithm.

Table 5 shows the parameters of the GA. The population size is not as large as that applied in [10]. During our parameter tuning phase, the GA averagely requires seven seconds when the population size is set as 100. The computation time requires more than 100 seconds for the population size 1000. To increase the solution diversity and decrease the computation cost, we use the population size 100. For the Casanova, all parameters are the same as that proposed in [8], and they are shown in Table 6. The NESA includes only one parameter, the trembling probability, and it is set as 5%. We observed that various trembling probabilities do not affect the solution qualities dramatically.

5.2. Revenue Performance. Given 200 seconds, the revenue performance in the Decay, Random and Uniform bid models

Table 6: The parameter settings of the Casanova.

Parameters	Settings
wp	0.15
np	0.5
Restart	1000

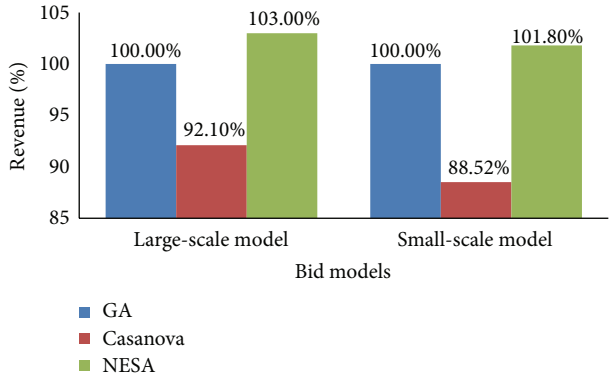

Figure 1: The revenue performance in the Decay model.

is shown in Figures 1, 2, and 3, respectively. In each figure, the results of the large-scale model with 200 goods and 2000 bids are arranged in the left, and the small-scale model results are in the right. The auctioneer's revenue is presented in the percentage, and the base line is the GA.

The Casanova produces less revenue than the GA in all our simulations, but the difference is not too much. The maximum gap is 11.48% comparing to the GA, and it takes place in the small-scale Decay bid model. In addition, the revenue gap between the Casanova and the NESA is 13.05% at most. The Casanova performs worse than the GA in terms of the auctioneer's revenue, and we have two observations. First, the search direction of the Casanova is invariant. So, it is difficult to improve the solution quality without increasing the solution diversity. For example, the NESA considers *longJump*() to increase the solution diversity, and the GA has the mutation operator. Another issue is the parameter setting. The Casanova has three parameters. During the parameter training phase, we did not find the optimal settings for all bid models. We applied the same settings as that appeared in [8] to our simulations. Thus, the Casanova does not perform well in all simulations. In summary, even though the Casanova provides the solutions worse than the GA and the NESA, the gap is not too much. After overcoming the above issues, the Casanova could increase the solution quality in expectation.

For the overall performance, the auctioneer gains more revenue by using the NESA than other algorithms in all our simulations. The extra revenue that the NESA can reach (comparing to the GA) is proportional to the number of goods in a bundle. If each bidder prefers to buy larger number of auction goods, the auction can obtain more revenue via using the NESA. For example, the NESA in the Random bid model performs better than in the Decay bid model.

On the other hand, in the large-scale instances, the NESA obtains the solutions with more auctioneer's revenue than in

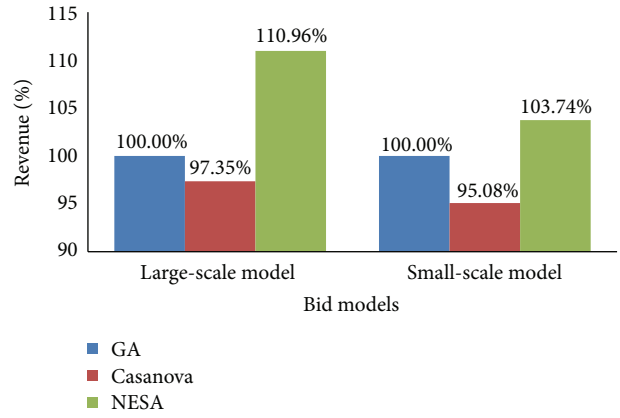

Figure 2: The revenue performance in the Random model.

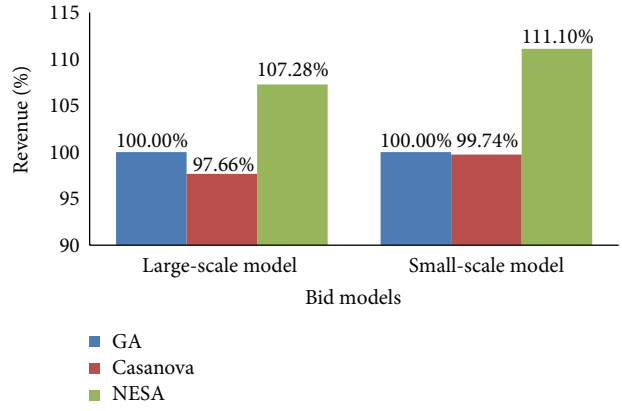

Figure 3: The revenue performance in the Uniform model.

the small-scale instances. Although the NESA in the large-scale instances provides higher solution quality than in the small-scale instances, the improvement is not proportional to the problem scale. For example, the auctioneer in the large-scale Decay model only receives 1.8% extra revenue than in the small-scale Decay model, but the problem scale is increased two times. So, we conjecture that the computation time is not sufficient for the NESA to compute the solution with maximum revenue. We will discuss this observation in the next section.

In summary, the NESA is an outstanding approach in terms of the solution quality for solving the WDP. In particular, in the large-scale instances and the large number of goods in a bundle, the NESA can provide the solutions with more auctioneer's revenue.

5.3. Anytime Performance. We randomly select an instance of each bid model and use the average values of ten runs for each algorithm. The revenue is tracked in every second. To capture the explicit convergence information, we only considered the large-scale models. Except for the GA, the Casanova, and the NESA, we combine the *findNE*() with the Casanova, labeled by the Hybrid. Figures 4, 5, and 6 show the anytime performance in various bid models.

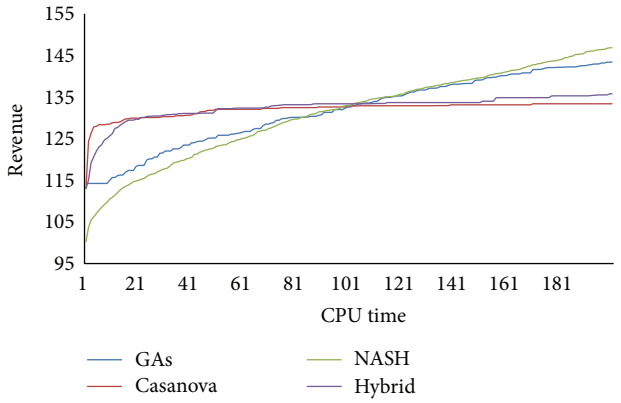

Figure 4: The anytime performance in the large-scale Decay model.

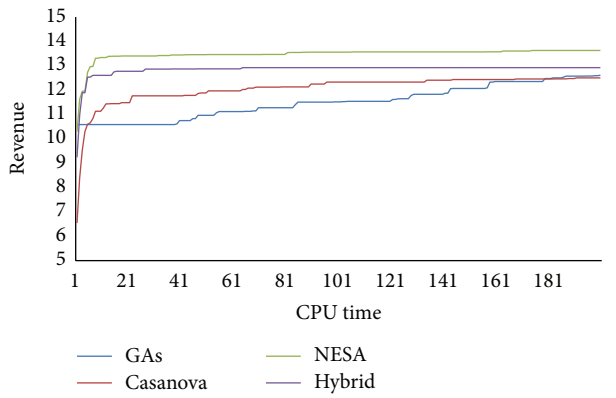

Figure 5: The anytime performance in the large-scale Random model.

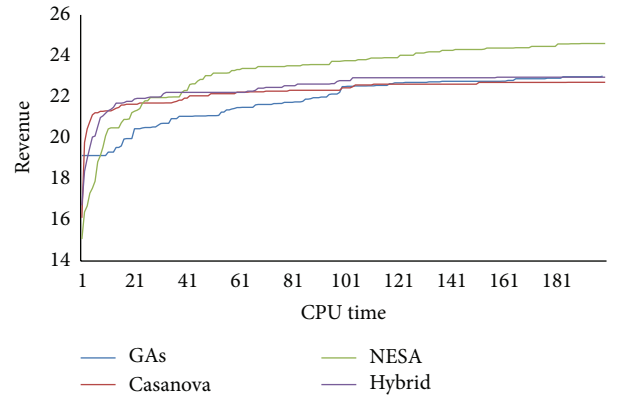

Figure 6: The anytime performance in the large-scale Uniform model.

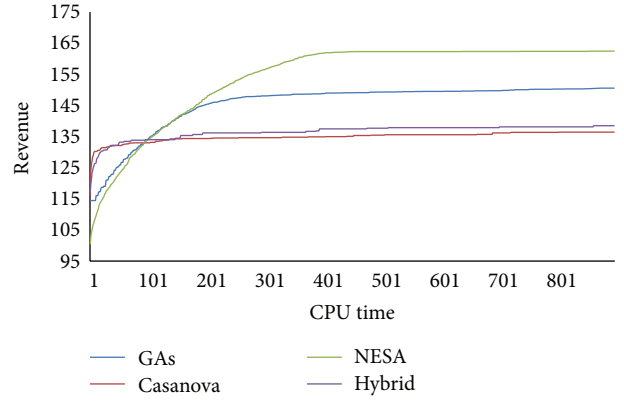

Figure 7: The convergence between all algorithms in the large-scale Decay model.

Because the GA searches widely in each iteration, the curve of the anytime performance is not raised very fast. The GA may obtain better results given enough time for the complete search. To improve the running time of the GA, the auctioneer can take into account the initial solution with high quality. Avasarala used a Casanova solution to seed the initial population of the GA [10]. Their experiments showed that the anytime performance could be improved but not too much. So, seeding the initial population may improve the anytime performance of the GA, but how to guarantee the quality of the obtained solution is another problem.

The Casanova converges rapidly in all bid models. In the early stage, for example, first 100 seconds in Figure 4, the Casanova outputs more revenue than other algorithms. However, more running time does not result in higher solution quality. For the aspect of the search amount, the Casanova only estimates one solution in an iteration. Moreover, the search direction is fixed because the RPI is applied to rank each bid. So, the Casanova rapidly meets the solution with maximum revenue. We tried to use the *findNE*() to search the NE from the solutions obtained by the Casanova, but the revenue is improved slightly. In Figures 4, 5, and 6, the curves of the Casanova and the Hybrid are very close. This simulation turns out that requiring low computation time is the major advantage of the Casanova, but the search direction restricts the solution quality. In the instances that the auctioneer must obtain the solutions in a short time, the Casanova is the best approach.

The NESA considers single solution in each iteration, but it requires more running time than the Casanova to finish an iteration. Each solution stays in the *findNE*() for a long time to check whether it is an NE solution or not. Computing the NE solution is the bottleneck of the NESA. From the results shown in Figures 4, 5, and 6, the NESA requires more time to converge in the instances that each bundle includes fewer goods, for example, the Decay bid model. Therefore, the computation efficiency of the NESA is inversely proportional to the number of goods in a bundle.

In Figure 4, the solution quality of the NESA is still increased, so the NESA requires more running to meet the solution with maximum revenue. To capture more details about the convergence information of all algorithms, we evaluate the anytime performance in the large-scale Decay model. Moreover, we increase the running time from 200 to 1200 seconds. The results are shown in Figure 7. After 900 seconds, all algorithms do not significantly improve the solution qualities, so we only drew the results within first 900 seconds. The curves of all algorithms shown in Figure 7 are clearer than those shown in Figure 4 in terms of reaching

the solutions with maximum auctioneer's revenue. Increasing the running time does not improve the auctioneer's revenue very much for the GA, the Casanova, and the Hybrid. The revenue is improved 5.689%, 2.70%, and 2.41% approximately for the GA, the Casanova, and the Hybrid, respectively. The NESA requires 421 seconds to obtain the solution with maximum revenue. Although the NESA spends more running time to meet the best solution than other algorithms, it produces more 6.63% revenue than the GA after 900 seconds. Moreover, this simulation also shows that the running time required by the NESA is acceptable for reaching the best solution in the worst case. Therefore, the NESA is appropriate to be implemented in the real-world services even if the large-scale WDP problems are considered.

5.4. Optimal Solution Comparison. According to above simulations, the NESA outperforms the GA and the Casanova in terms of the auctioneer's revenue and requires reasonable running time. In this section, we compare the auctioneer's revenue obtained by the NESA with the optimal solution. The WDP is formulated as an integer linear program, so we use the CPLEX (http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/) to compute the optimal solutions. We only consider the small-scale problems with 100 goods and 1000 bids because the computation resource required by the CPLEX in the large-scale problem exceeds that equipped in our simulation environment.

Auctioneer's revenue obtained from the *findNE*() may depend on the input solution quality. An initial solution with higher revenue may lead to a better NE solution. Except for the random approach, we use an RPI-based approach to improve the initial solutions. For each bid b_i, the RPI-based approach computes the RPI values as follows:

$$\mathrm{RPI}_{b_i} = \frac{p_i}{|b_i|}. \tag{5}$$

We first rank each bid b_i as the decreasing order of RPI_{b_i}. Then, adding the bids to build a feasible solution based on the rank result. We consider this solution as the input of the *findNE*(). So, three algorithms participate in the optimal solution comparison. We take 200 rounds for each algorithm and keep track of the variance of the solution quality in each round. The results of the optimal solution comparison are shown in Figures 8, 9, and 10, where the base line is the optimal solution.

The *findNE*() with the RPI-based approach, which is drawn in the red line, produces the stable results which are very close to the optimal solutions. In Figures 8, 9, and 10, we observe that there are some break-out points, and those results have more revenue than others. After tracking the log, each break-out point results from the successful trembling, but not all trembles improve the solution quality. Therefore, the trembling hand assumption is useful for seeking better solutions.

The qualities of the solutions obtained by the *findNE*() with random input solutions illustrated by the green line are unstable. Except for the Random model, the *findNE*() with random input solutions performs worse than the *findNE*() with the RPI-based approach and the optimal solutions. Although the random input solution provides the unrestricted search direction, the quality of the NE solution is only outstanding in few instances of the Random model. So, seeding the input solution will result in stable solutions and high solution qualities.

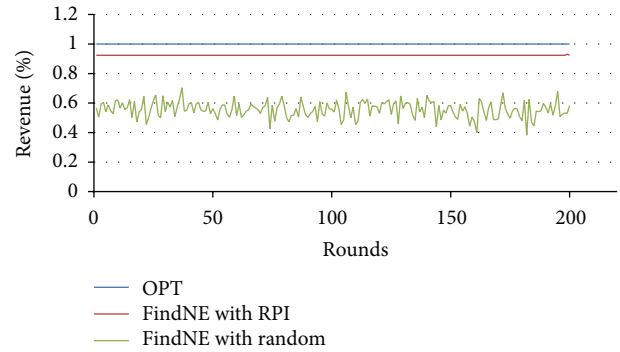

Figure 8: The optimal solution comparison in the large-scale Decay model.

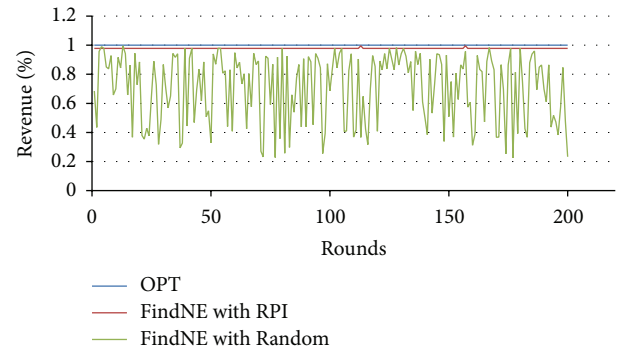

Figure 9: The optimal solution comparison in the large-scale Random model.

6. Discussion and Remark

Computing the NE with maximum utility for each player has been proved as a PPAD-complete problem [13, 18]. In a game, if the number of the union of strategy sets for all players is grown exponentially, reaching the NE requires huge computation time. Theoretically, finding a steady solution via the NESA may require huge computation time.

Solving the WDP is an NP-complete problem [5]. This means that in general cases no algorithm can compute the optimal winner set in the polynomial time unless P = NP. The search space is the critical issue to evaluate the computation complexity of the NESA. The computation complexity is inversely proportional to the degree of intersections between bundles. The worst case takes place when each bundle includes exactly two goods. Declaring a bidder as a winner means that many bidders cannot win their bundles simultaneously. In our simulations, we consider three bid models with different average numbers of goods in each bundle to prove this discussion. The simulation results turn

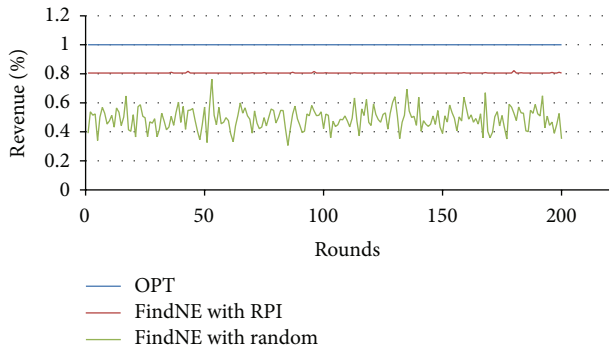

Figure 10: The optimal solution comparison in the large-scale Uniform model.

out that the computation complexity of the NESA is inversely proportional to the number of goods in a bundle. So, this discussion is correct.

The procedures in line (5) of Algorithm 2 and line (8) of Algorithm 3 deal with declaring some bidders as winners. The selection strategy of determining winners affects the solution quality. In general, the convergence speed and the auction's revenue are the trade-off. As shown in Section 5.3, the Casanova (the search with the fixed direction) rapidly finds a solution, but the auctioneer's revenue is not the major consideration. On the other hand, the GA (the comprehensive search) requires more computation time than the Casanova to generate a solution, but the auctioneer can gain more revenue. Different strategies of selecting winners can be applied to the NESA based on the consideration of the auctioneer. In this paper, we apply the idea of searching the NE to propose the framework for solving the WDP. The auctioneer can adjust the search direction when using the NESA to solve the WDP. For example, the search direction which includes the specific goal, for example, the RPI ranking function, is better in the larger-scale cases based on the computation efficiency consideration. In Section 5.4, using the input solutions with RPI approach is more stable and provides more revenue than the random input solutions.

In this paper, we propose a game model to represent the WDP and the NESA to find the NE solutions. The properties of the NESA can be discussed in the aspects of the game model and the solution quality. When selling several goods to multiple buyers, most sellers usually estimate the revenue improvement of selling a good to another buyer. We use this impression to formulate the game model of conceiving each good as a player. Each player (the auction good) can determine the winner and evaluate the revenue improvement of changing the winner. The translation from the WDP to the proposed game model does not violate the real-world behaviors, so the auctioneers would accept the proposed game. On the other hand, the NESA provides the solutions that the revenue is very close to the optimal solution. Furthermore, the running is reasonable. In summary, the NESA is appropriate to be implemented in the real-world service to compute the WDPs.

7. Conclusion

We propose the NESA to solve the WDP in the combinatorial auctions. The NESA utilizes the stability of the NE, the self-learning of the evolutionary game, and the mistake making of the trembling hand assumption to seek for the solutions with high auctioneer's revenue. The auctioneers can earn the revenue which is very close to the optimal solution via adopting the NESA. Moreover, the NESA requires only rational running time.

In our simulations, we consider the bid prices with some specific distributions to evaluate the performance of the NESA. According to the real-world data analysis [19], most bidders submit the prices with their private considerations. We only consider three major bid models to evaluate the performance of the NESA in this paper. However, this may be not sufficient to understand the performance of the NESA in the real-world services. For example, the bid profile may not be satisfied with a well-defined distribution. To apply the NESA in the real-world services, taking into account the relationship between the auction item and the bid profile is our next research issue. Upon understanding the properties of the received bid profile, the auctioneer can use some techniques to improve the search efficiency and the solution quality.

In the future, we will first study the data mining approaches, such as the mining association rules [20] and feature extraction [21], to explore the relationship between the auction items and the bidding behaviors. Then, we will evaluate the instances that are generated via simulating the real-world bidding behaviors. We focus on pruning the unnecessary searches of the NESA to reduce the computation complexity, but the solution quality should not be decreased. The NESA will provide a more efficient way to compute the solutions with high auctioneer's revenue in expectation.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported in part by Taiwan NSC under Grant no. NSC 102-2221-E-274-004 and NSC 102-2221-E-194-054-. The authors would like to thank reviewers for their insightful comments which helped to significantly improve the paper.

References

[1] G. H. Williams and T. R. Anderson, "Incentives, information, and winner's curse in construction industry bidding," in *Proceedings of the Portland International Conference on Management and Technology (PICMET '97)*, pp. 386–390, Portland, Ore, USA, July 1997.

[2] G. Adomavicius, S. P. Curley, A. Gupta, and P. Sanyal, "Impact of information feedback in continuous combinatorial auctions: an experimental study of economic performance," *MIS Quarterly*, vol. 37, no. 1, pp. 55–76, 2013.

[3] M. H. F. B. M. Fauadi, S. H. Yahaya, and T. Murata, "Intelligent combinatorial auctions of decentralized task assignment for AGV with multiple loading capacity," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 8, no. 4, pp. 371–379, 2013.

[4] T. Sandholm, S. Suri, A. Gilpin, and D. Levine, "CABOB: a fast optimal algorithm for combinatorial auctions," in *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI '01)*, pp. 1102–1108, Seattle, Wash, USA, 2001.

[5] Y. Fujishima, K. Leyton-Brown, and Y. Shoham, "Taming the computational complexity of combinatorial auctions: optimal and approximate approaches," in *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI '99)*, pp. 548–553, Stockholm, Sweden, 1999.

[6] T. Sandholm and S. Suri, "BOB: improved winner determination in combinatorial auctions and generalizations," *Artificial Intelligence*, vol. 145, no. 1-2, pp. 33–58, 2003.

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

[8] H. H. Hoos and C. Boutilier, "Solving combinatorial auctions using stochastic local search," in *Proceedings of the 17th National Conference on Artificial Intelligence*, pp. 22–29, Austin, Tex, USA, 2000.

[9] V. L. Smith, P. Cramton, Y. Shoham, and R. Steinberg, *Combinatorial Auctions*, MIT Press, Cambridge, Mass, USA, 2010.

[10] V. Avasarala, H. Polavarapu, and T. Mullen, "An approximate algorithm for resource allocation using combinatorial auctions," in *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT '06)*, pp. 571–578, Hong Kong, China, December 2006.

[11] Y. Sakurai, M. Yokoo, and K. Kamei, "An efficient approximate algorithm for winner determination in combinatorial auctions," in *Proceedings of the 2nd ACM Conference on Electronic Commerce (EC '00)*, pp. 30–37, Minneapolis, Minn, USA, 2000.

[12] J. F. Nash Jr., "Equilibrium points in *n*-person games," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 36, no. 1, pp. 48–49, 1950.

[13] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic Game Theory*, Cambridge University Press, Cambridge, UK, 2007.

[14] M. J. Osborne, *An Introduction to Game Theory*, Oxford University Press, Oxford, UK, 2004.

[15] M. Mito and S. Fujtta, "On heuristics for solving winner determination problem in combinatorial auctions," *Journal of Heuristics*, vol. 10, no. 5, pp. 507–523, 2004.

[16] R. Selten, "Reexamination of the perfectness concept for equilibrium points in extensive games," *International Journal of Game Theory*, vol. 4, no. 1, pp. 25–55, 1975.

[17] X.-B. Hu, M. Wang, and M. S. Leeson, "Ripple-spreading network model optimization by genetic algorithm," *Mathematical Problems in Engineering*, vol. 2013, Article ID 176206, 15 pages, 2013.

[18] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou, "The complexity of computing a Nash equilibrium," *SIAM Journal on Computing*, vol. 39, no. 1, pp. 195–259, 2009.

[19] S. Yuan, J. Wang, and X. Zhao, "Real-time bidding for online advertising: measurement and analysis," in *Proceedings of the 7th International Workshop on Data Mining for Online Advertising (ADKDD '13)*, Chicago, Ill, USA, 2013.

[20] C. R. Valencio, F. T. Oyama, P. S. Neto et al., "MR-Radix: a multi-relational data mining algorithm," *Human-Centric Computing and Information Sciences*, vol. 2, article 4, 2012.

[21] K. Sarkar, M. Nasipuri, and S. Ghose, "Machine learning based keyphrase extraction: comparing decision trees, naïve Bayes, and artificial neural networks," *Journal of Information Processing Systems*, vol. 8, no. 4, pp. 693–712, 2012.