A Recombination-Based Tabu Search Algorithm for the Winner Determination Problem

Ines Sghir[1,2], Jin-Kao Hao[1], Ines Ben Jaafar[2], and Khaled Ghédira[2]

[1] LERIA, Université d'Angers, 2 Bd Lavoisier, 49045 Angers Cedex 01, France
[2] SOIE, ISG, Université de Tunis, Cité Bouchoucha 2000 Le Bardo, Tunis, Tunisie

Abstract. We propose a dedicated tabu search algorithm (TSX_WDP) for the winner determination problem (WDP) in combinatorial auctions. TSX_WDP integrates two complementary neighborhoods designed respectively for intensification and diversification. To escape deep local optima, TSX_WDP employs a backbone-based recombination operator to generate new starting points for tabu search and to displace the search into unexplored promising regions. The recombination operator operates on elite solutions previously found which are recorded in an global archive. The performance of our algorithm is assessed on a set of 500 well-known WDP benchmark instances. Comparisons with five state of the art algorithms demonstrate the effectiveness of our approach.

Keywords: Winner determination problem; Tabu Search; Solution recombination; Combinatorial optimization; Heuristics.

1   Introduction

An auction involves an auctioneer wishing to maximize his/her selling revenue and a set of bidders wishing to minimize their cost according to their valuations of the items that they want to acquire. Examples of the most widely known auctions are the English auction, the Holland's auction, the Sealed envelope auction, and the Vickrey auction [12]. These auctions typically handle one item per sell.

Combinatorial auctions are multi-item auctions, which allow bids on items combinations [5, 11]. In a combinatorial auction, we are given a set of items exposed to buyers. Buyers offers different bids, each bid being defined by a subset of items with a price (bidder's valuation). Two bids are conflicting if they share at least one item. The Winners Determination Problem (WDP) is to determine a conflict-free allocation of items to bidders (the auctioneer can keep some of the items) that maximizes the auctioneer's revenue defined as the sum of the valuations of the winning bids. The WDP is known to be a NP-hard problem with a number of practical applications like e-commerce, games theory and resources allocation in multi-agents systems [21, 11].

Formally, given a set of items $M = \{1, 2, ..., m\}$ and a set of n bids $N = \{1, 2, ...n\}$. Each bid j is a tuple $< S_j, P_j >$ where S_j is a subset of items covered by bid j, and P_j, the price of bid j. Let B be a $m \times n$ binary matrix

such that $B_{ij} = 1$ if object $i \in S_j$, $B_{ij} = 0$ otherwise. Furthermore, define a decision variable x_j for each bid j such that $x_j = 1$ if bid j is a winning bid, 0 otherwise. Then, the WDP can be stated as the following binary integer optimization problem.

$$Maximize \; f(x) = \sum_{j \in N} P_j x_j \qquad (1)$$

subject to

$$\sum_{j \in N} B_{ij} x_j \leq 1, i \in M \qquad (2)$$

The objective function (1) allows to maximize auctioneer's gain calculated by the sum of prices of the winning bids while the constraints expressed by formula (2) ensure that an item appears at most in one winning bid.

The computational challenge of the WDP and its practical applications have motivated a number of solution approaches including exact methods [18] and metaheuristic methods. Representative examples of exact methods include: Branch-on-Items (BoI), Branch-on-Bids (BoB) [19], Combinatorial Auctions BoB (CABoB) [20], Combinatorial Auction Structural Search (CASS) [6] and Combinatorial Auctions Multi-unit Search (CAMUS) [15]. A dynamic programming approach is introduced in [17] while a linear programming method is investigated in [16]. An algorithm based on integer programming is shown in [1], a constraint programming approach is used to solve a particular combinatorial Vickrey auction [9]. On the other hand, several stochastic methods were proposed for the WDP. They include a local search method named Casanova [10], a hybrid algorithm combining simulated annealing with Branch-and-Bound (SAGII) [8], and more recently a tabu search method [3] and a memetic algorithm [4].

The rest of the paper is organized as follows. Section 2 describes the proposed algorithm which is based on two complementary neighborhoods and a recombination operator. Experimental results are reported in section 3 and compared with five representative algorithms for the WDP. Finally, section 4 concludes the paper.

2   Recombination-Based Tabu Search for the WDP

TSX_WDP uses two complimentary move operators to explore effectively the search space and a recombination operator as an additional means to escape deep local optima. In this section, we presents in detail these key components.

2.1   The solution representation

A candidate solution is represented by an allocation A (a dynamic vector). Each element of this allocation A receives the winning bid. Each bid is an object composed of the list of items and the associated prices.

2.2 The evaluation function

The objective function defined in equation (1) is used to measure the quality of a candidate solution. So if an allocation A contains k bids $\{B_1, B_2, ..., B_k\}$, ($B_i = <S_i, P_i>, 1 \leq i \leq k, k \leq n$), its quality is just equal to $f(A) = \sum_{i=1}^{k} P_i$, i.e., the sum of the valuations of the winning bids. Given two candidate solutions, the one with a higher objective value is considered to be better. This relation is used to compare neighboring solutions which are developed below.

2.3 The basic move operators and the neighborhoods

Our TSX_WDP algorithm explores the search space by using two complementary neighborhood relations which are defined by an intensification move operator and a perturbation move operator.

Intensification move The intensification move operator chooses bids among candidate bids to be inserted in the current allocation A. During one iteration of the algorithm, several bids can be selected if they improve the current allocation. To create a neighboring allocation, the following steps are followed:

- The initial candidate bids are sorted according to their utility prices;
- For each candidate bid B_x, a binary gain function is used to verify if the bid can increase the revenue of the current allocation when the bid is inserted;
- Let Q be the set of winning bids that are in conflict with the current candidate bid B_x, Let $f(Q)$ be the revenue of the set of winning bids Q, and $f(B_x)$ the price of the candidate bid B_x. The gain function returns true if $f(Q) < f(B_x)$ and returns false otherwise;
- Based on the function f, a candidate bid B_x can be added to the current allocation only if its price $f(B_x)$ is higher than the revenue of other winning bids which are conflicting with B_x in the current allocation (i.e., the gain function is true);
- The gain of B_x, when it is selected to be added in the current allocation, is calculated by: $Gain(B_x) = f(A) - f(Q) + f(B_x)$;
- When a bid B_x is inserted in the current allocation A, the bids of Q which are conflicting with B_x are removed from A;
- The steps mentioned previously are iterated until all the initial candidate bids are visited and possibly added in the current allocation A.

Perturbation move The perturbation move operator chooses randomly one candidate bid from the available ones. This move is activated only if no bid among the candidate bids can improve the current solution. In fact, the application of the intensification move can make the search to be trapped into local optima during the search process, when no more bid can be found that improves the revenue of the current allocation. Notice that this move operator can decrease temporarily the revenue of the solution, but hopefully, it helps the search to escape local optima by displacing the search to new zones of the search space. This move operator plays thus a diversification role.

2.4 Tabu list and tabu tenure management

Tabu search uses a tabu list to forbid recently visited solutions from being revisited. The TSX_WDP algorithm considers the following general prohibition rule: a bid that is chosen to be inserted in the current allocation A (by an intensification move or a perturbation move) is forbidden to be removed for the next tt iterations (called tabu tenure). tt is calculated dynamically by the function proposed in [7]: $tt = L + \lambda + f(A)$ where L is randomly chosen from the interval $[0, 9]$ and λ is empirically fixed to 0.6. Experimentations show this dynamic tabu tenure is robust and allows TSX_WDP to reach high quality solutions. Notice that we permit a move to be accepted in spite of being tabu if the move leads to a solution better than any found so far. This is called the aspiration criterion.

2.5 Elite solution archive

The proposed algorithm also relies on a solution recombination operator (see next section) which aims to blend elite solutions (high-quality local optima). This technique is based on an archive P which is built as follows. During the search, if the current best solution A^* is not improved within a fixed number p of consecutive iterations, A^* is considered as a good local optimum and is added into the archive P. At the same time, this allocation corresponds to a deep local optimum which is difficult to escape. For this purpose, we trigger a recombination operation to create a new starting point for the tabu search procedure, which is explained in the next selection.

2.6 Recombination operator

The recombination operator aims to transfer good properties of parents to their descendants. The recombination pseudo-code is given in Algorithm 1.

Algorithm 1 The recombination operator

Require: two parent solutions I_1 and I_2
Ensure: An offspring solution I_0
1: $I_0 \leftarrow \emptyset$, $D_1 \leftarrow \emptyset$, $D_2 \leftarrow \emptyset$
2: Sort the bids in each parent according to their prices
3: **while** I_1 and I_2 are not empty **do**
4:     $D_1 \leftarrow first\_element(I_1)$
5:     $D_2 \leftarrow first\_element(I_2)$
6:     if D_1 and/or D_2 are not conflicting with the bids in I_0, add D_1 and/or D_2 to I_0
7:     remove D_1 from I_1
8:     remove D_2 from I_2
9: **end while**
10: Return Child I_0

Given two parent allocations I_1 and I_2 from the elite solution archive which share the highest number of bids, the recombination operator constructs the

offspring I_0 in k steps until all the bids of the two parents are visited. Our recombination operator is inspired by the idea of backbone used in [2, 22]. In the first step, the set of bids shared by the parents are identified and directly transfered to I_0. Then the following steps are performed:

– Choose the bid with the lowest price from each parent (lines 4 and 5, Algorithm 1).
– The two selected bids are candidate bids that can be inserted in the offspring, if they are not conflicting bids. This is done by conserving the best bids with the highest revenue (lines 6 and 7, Algorithm 1).
– Remove the selected bids from their parents, even if they are not inserted in the offspring (lines 9 and 10, Algorithm 1).
– Repeat the previous steps until all the bids of the parents are examined and removed.

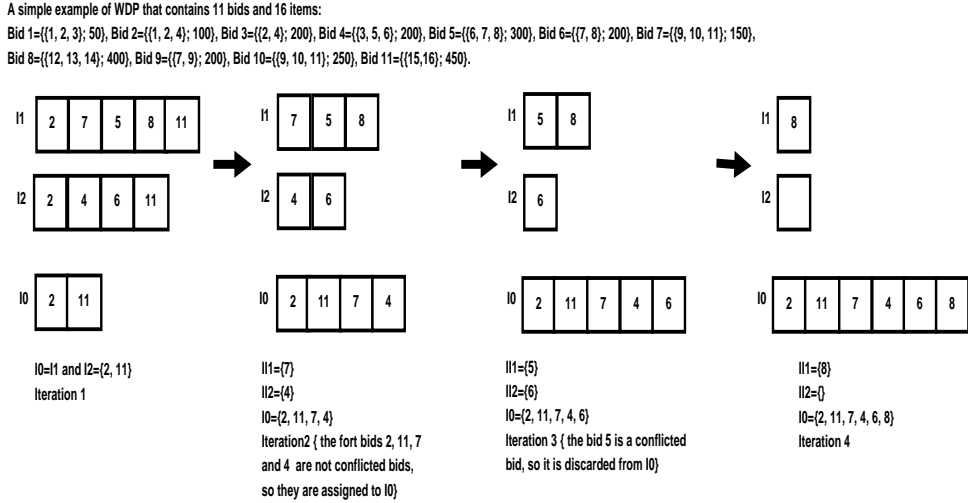An example of this recombination operation is provided in Fig. 1.

A simple example of WDP that contains 11 bids and 16 items:
Bid 1={{1, 2, 3}; 50}, Bid 2={{1, 2, 4}; 100}, Bid 3={{2, 4}; 200}, Bid 4={{3, 5, 6}; 200}, Bid 5={{6, 7, 8}; 300}, Bid 6={{7, 8}; 200}, Bid 7={{9, 10, 11}; 150}, Bid 8={{12, 13, 14}; 400}, Bid 9={{7, 9}; 200}, Bid 10={{9, 10, 11}; 250}, Bid 11={{15,16}; 450}.

Fig. 1. An example of the recombination operator

2.7 The TSX_WDP algorithm

The general TSX_WDP algorithm is formalized in Algorithm 2. The algorithm starts with an empty allocation in which no bid is chosen and tries to improve it by looking for a better solution in the current neighborhood. In each iteration, the best authorized bids are selected among the candidate bids to be included

in the current allocation. This is achieved with the intensification move (lines 7-9 of Algorithm 2). When no bid can be found to increase the revenue with the intensification move, TSX_WDP switches to the perturbation move by choosing a random bid from the candidate bids (line 11 of Algorithm 2). In both cases, the choice of the bids depends on the status of the tabu list which is updated after each move. Any conflicting bids in the current allocation, when new bids are considered, are removed (lines 13 and 14 of Algorithm 2). The search process is repeated for a fixed number $Itermax$ of iterations. During these $Itermax$ iterations, if the current best solution cannot be updated for consecutive p (fixed experimentally) moves, the best local optimum found so far is inserted into the archive P and the recombination operator is activated to generate a new starting point for a new round of the tabu search procedure (lines 20-25 of Algorithm 2).

2.8 Discussion

The proposed TSX_WDP algorithm distinguished itself from the existing heuristic approaches by several features. First, its tabu search procedure is based on two complementary move operators to generate neighboring solutions. In particular, the intensification move can add several bids (instead of a single bid like in most local search based heuristics). The tabu search procedure adopts a dynamic tabu tenure which is missing in the existing methods. Second, the recombination operator is based on the idea of backbone which proves to be quite useful for the WDP.

3 Experimentation

This section gives experimental results of the proposed algorithm which is implemented in Java. The program is run on a computer with a Core I5 2.5GHz, 8GB of RAM. To assess our TSX_WDP algorithm, we run TSX_WDP on various benchmarks of diverse sizes defined in [13]. Theses benchmarks take into account several factors like the prices, bidders preferences and object distribution on bids. They can be divided into five groups where each group contains 100 instances.

-REL 500-1000: From in101 to in200: m = 500, n = 1000
-REL 1000-1000: From in201 to in300: m = 1000, n = 1000
-REL 1000-500: From in401 to in 500: m = 1000, n = 500
-REL 1000-1500: From in501 to in 600: m = 1000, n = 1500
-REL 1500-1500: From in601 to in 700: m = 1500, n = 1500

We calibrated the parameters of the proposed algorithms by an experimental study: The maximum number of iterations ($itermax$) is fixed to 200 and the parameter responsible for the tabu tenure λ is fixed to 0.00006. Each of the 500 instance is solved 40 times independently by the TSX_WDP algorithm with different random seeds.

Algorithm 2 TSX_WDP for the Winners Determination Problem

Require: A matrix M, a parameter $Itermax$, Vector of bids B, Parameter p

Ensure: A vector of winning bids A^* and its revenue $f(A^*)$

1: $Iter \leftarrow 0$ {Iteration counter}, Initiate $tabu\_list$
2: $A^* \leftarrow A \leftarrow \varnothing$
3: $opt \leftarrow 0$ {An counter that is incremented if the current solution does not improve in two consecutive iterations; opt returns to 0, when it exceeds the value p, after activating the recombination operator}
4: initialize $tabu\_list$
5: $P \leftarrow \varnothing$ {Archive of the best local optima encountered A^*}
6: **while** $(Iter < Itermax)$ **do**
7:    Construct neighborhoods from A based on the intensification move
8:    **if** There exists an intensification move **then**
9:       Choose an overall best allowed neighbor A' according to max gain criterion and by considering M {to remove from A' any conflicting bid) {Section 2.3}
10:    **else**
11:       Apply the perturbation move {Section 2.3} by choosing a random bid from B to create a neighbor A'
12:    **end if**
13:    $A \leftarrow A'$ (Move to the selected neighboring solution A')
14:    Update $tabu\_list$ {Section 2.4} and B {delete the winner bids from B and add the looser bids in it}
15:    **if** $f(A) > f(A^*)$ **then**
16:       $A^* \leftarrow A$
17:    **else**
18:       $opt \leftarrow opt + 1$
19:    **end if**
20:    **if** $opt = p$ **then**
21:       Add A^* to the Archive P
22:       $I_1, I_2 \leftarrow$ Parent_Selection(P) {Section 2.5}
23:       $I_0 \leftarrow$ Recombination_Operator(I_1, I_2) {Section 2.6}
24:       $A \leftarrow I_0$
25:       $opt \leftarrow 0$
26:    **end if**
27:    $Iter \leftarrow Iter + 1$
28: **end while**
29: **return** $(A^*$ and $f(A^*))$

3.1 Experimental results

In Table 1, we present the computational results of the TSX_WDP algorithm on the five groups of benchmarks. Given that there are 500 instances, we show only some results of each group, like in some recent papers [4]. For each presented instance, the following computational statistics are indicated: the *maximum revenue* obtained by the TSX_WDP algorithm over the 40 independent trials (Rbest), the *average revenue* over the 40 trials (Ravg), the *worst revenue* over the 40 trials (Rworst) and the average CPU time in seconds (AvgTime). As one can observe, the values of Ravg are very close to the values of Rbest in most of cases and these two values are even equal for certain instances (for example for in101, in102, in205 etc.). This table shows the proposed algorithm can consistently reach high quality solutions for the tested problems.

Table 1. Results obtained by TSX_WDP for WDP benchmarks

Instances	Rbest	Ravg	Rworst	AvgTime	Instance	Rbest	Ravg	Rworst	AvgTime
in101	69585.298	69585.298	69585.298	88	in201	81557.742	80383.277	79331.63	56
in102	72518.222	72518.222	72518.222	76	in202	89289.573	86815.261	81291.193	52
in103	69730.618	69475.485	65903.632	75	in203	86239.213	83941.410	77220.427	54
in104	71327.641	70765.941	65948.396	78	in204	84879.397	84374.869	76822.810	55
in105	73351.044	71570.624	68899.994	93	in205	83748.837	83748.837	83748.837	57
in401	77417.482	77191.182	70628.481	12	in501	83738.040	83506.552	82605.443	107
in402	76273.336	76153.051	74469.073	10	in502	83297.340	82546.590	76751.565	82
in403	74843.958	74356.247	69989.28	10	in503	83718.749	82017.955	78112.719	81
in404	78761.690	78597.224	77939.364	10	in504	83944.901	82772.535	77217.558	76
in405	75915.900	75640.510	74899.125	10	in505	83071.930	81876.413	78909.275	66
in601	107246.248	102862.848	96840.461	117	in602	99668.269	97854.579	91452.904	78
in603	98577.454	96567.287	95219.36	75	in604	101713.602	100786.326	99395.413	78

3.2 Comparative results for the WDP

In order to further show the effectiveness of the TSX_WDP algorithm, we present a comparative study with five state of the art algorithms from the literature: Casanova [10], SAGII [8], SLS [3], TS [3], MA [4].

In Table 2, we show the general comparative results for each group. In this table, rows μ correspond to the average of best objective value of the 100 instances in each group. Rows *time* represent the average time to reach the best solution. $\delta(\%)$ is the deviation of the TSX_WDP algorithm with respect to each reference algorithm. The deviations are calculated respectively as follows: $\mu_{TSX\_WDP} - \mu_{algo\_X})/\mu_{TSX\_WDP}$ where *algo_X* is one of the five reference algorithms. Since the compared algorithms are implemented in different languages and run on different plateforms, the comparison is focused on solution

quality that can be reached by each algorithm. The computing time is provided only for indicative purposes. The results of the reference algorithms are extracted from the corresponding papers except the results of Casanova which are from [8].

Table 2. Comparative results of TSX_WDP with Casanova, MA, SLS, TS, SAGII on the WDP benchmarks: μ is the average of the best objective value of the 100 instances in each group. *time* is the average time to reach the best solution.

Test Set	100 instances	REL-500-1000	REL-1000-500	REL-1000-1000	REL-1000-1500	REL-1500-1500
TSX_WDP	Time	74.19	9.45	48.98	75.92	90.61
	μ	**69647.975**	**75274.184**	**86786.159**	**85577.806**	**103178.732**
Casanova	Time	119.46	57.74	111.42	168.24	165.92
	μ	37053.78	51248.79	51990.91	56406.74	65661.03
$\delta_{TSX/Casanova}(\%)$		46.79	31.91	40.09	34.08	36.36
TS	Time	91,07	25.84	104,30	223,37	175.68
	μ	65286.94	71985.34	81633.63	77931.41	97824.64
$\delta_{TSX/TS}(\%)$		6.26	4.36	5.93	8.93	5.18
SLS	Time	22.35	5.91	14.19	14.97	16.47
	μ	64216.14	72206.07	82120.31	79065.08	98877.07
$\delta_{TSX/SLS}(\%)$		7.79	4.07	5.37	7.61	4.16
MA	Time	56.64	14.98	33.05	24.51	28.22
	μ	65740.25	73604.62	83304.20	79644.64	99957.96
$\delta_{TSX/AM}(\%)$		5.61	2.21	4.01	6.93	3.12
SAGII	Time	38.06	24.46	45.37	68.82	91.78
	μ	64922.02	73922.10	83728.34	82651.49	101739.64
$\delta_{TSX/SAGII}(\%)$		6.78	1.79	3.52	3.41	1.39

Table 2 discloses that TSX_WDP gives an improvement between 31% and 47% in solution quality compared to Casanova. TSX_WDP finds better solutions with shorter times than Casanova. TSX_WDP shows good performances compared to SLS. The improvement is between 4% and 8%. The results of TSX_WDP are better than TS in quality and in time (with an improvement rate between 4% and 9%). TSX_WDP outperforms MA. The deviation is between 2% and 7%. Finally, TSX_WDP produces better results than SAGII which is currently the most successful algorithm for the WDP and is based on sophisticated Branch-and-Bound and preprocessing tools (The deviation is between 1% and 7%). Thus, we can conclude that TSX_WDP discovers new best results for the five groups of benchmarks.

To further illustrate the results of Table 2, we consider the comparative curves of Fig. 2. The X-axis of the curves represent the 5 groups of the WDP benchmarks and their Y-axis are the gain of each group (μ). These curves confirm that TSX_WDP competes favorably with each of the reference algorithms for each group of instances.
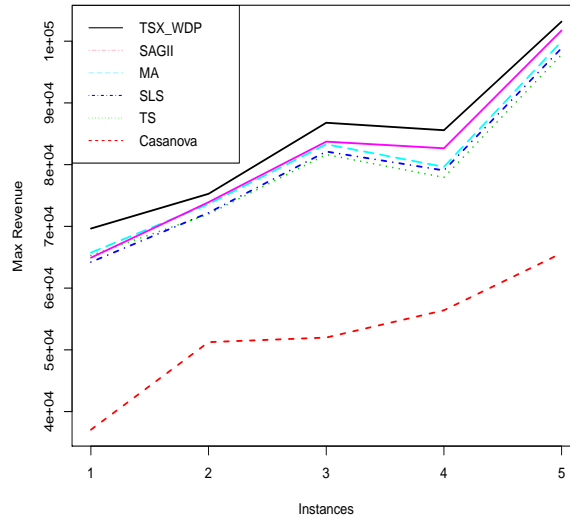
Fig. 2. A comparison of the solution quality between TSX_WDP, Casanova, TS, SLS, MA and SAGII

4 Conclusion

In this work, we have presented a tabu search algorithm for the winner determination problem based on a two different neighborhood structures and a recombination operator. The algorithm uses the intensification move to improve progressively the quality of the current solution. When the solution cannot be further improved, the TSX_WDP algorithm switches to a perturbation move by choosing a random bid. In both cases, a tabu list is used to prevent the search from revisiting the previous examined solutions. To escape deep local optima, the proposed algorithm employs a backbone-based recombination operator which relies on an elite solution archive which is built and updated during the search. This recombination operator generates new starting points for tabu search with the aim of leading the algorithm into new promising search areas. The proposed TSX_WDP algorithm is evaluated on a set of 500 benckmark instances. The comparative study with five reference algorithms shows the proposed algorithm is able to reach solution of very high quality.

Acknowledgment

2013) and LigeRO projects (2009-2013) from the Region of Pays de la Loire, France.

References

1. Andersson, A., Tenhunen, M., Ygge, F.: Integer programming for combinatorial auction winner determination. In: Proceedings of the 4th International Conference on Multi-agent Systems. IEEE Computer Society Press, pp 39–46, New York (2000)
2. Benlic, U., Hao, J.K.: A multilevel memetic approach for improving graph k-partitions. IEEE Transactions on Evolutionary Computation 15(5): 624–472, 2011.
3. Boughaci, D., Benhamou, B., Drias, H.: A memetic algorithm for the optimal winner determination problem. Soft Computing, 13(8-9): 905-917 (2009)
4. Boughaci, D., Benhamou, B., Drias, H.: Local Search Methods for the optimal winner determination problem in combinatorial auctions. Journal of Mathematical Modelling and Algorithms, 9(2): 165–180 (2010)
5. Cramton, P., Shoham, Y., and Steinberg, R.: Combinatorial auctions. MIT Press, Cambridge (2006)
6. Fujishima, Y., Leyton-Brown, K., Shoham, Y.: Taming the computational complexity of combinatorial auctions: optimal and approximate approaches. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence, pp 48–53, Sweden (1999)
7. Galinier, P., Hao, J.K.: Hybrid evolutionary algorithms for graph coloring. Journal of Combinatorial Optimization, 3(4): 379–397 (1999)
8. Guo, Y., Lim, A., Rodrigues, B., Zhu, Y.: Heuristics for a bidding problem. Computers & Operations Research, 33(8): 2179–2188 (2006)
9. Holland, A. , O'sullivan, B.: Towards fast Vickrey pricing using constraint programming. Artificial Intelligence Review, 21(3-4): 335–352 (2004)
10. Hoos, HH., Boutilier, C.: Solving combinatorial auctions using stochastic local search. In: Proceedings of the 17th National Conference on Artificial Intelligence, pp 22–29, USA (2000)
11. Abrache, J., Crainic, T.G, Gendreau, M., Rekik, M.: Combinatorial auctions. Annals of Operations Research, 153(1): 131–164 (2007)
12. Klemperer, P.,: Auctions: Theory and Practice, Princeton, N.J.: Princeton University Press, (2004)
13. Lau, H.C., Goh, Y.G.: An intelligent brokering system to support multi-agent web-based 4th-party logistics. In: Proceedings of the 14th International Conference on Tools with Artificial Intelligence, pp. 54-61, Washington, DC (2002)
14. Lehmann, D., Rudolf, M., Sandholm, T.: The Winner Determination Problem. In: Cramton et al. (ed) Combinatorial auctions. MIT Press, Cambridge (2006)
15. Leyton-Brown, K., Shoham, Y., Tennenholtz, M.: An algorithm for multi-unit combinatorial auctions. In: Proceedings of the 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence. AAAI Press/MIT Press, pp 56-61, Austin, Texas (2000)
16. Nisan, N.: Bidding and allocation in combinatorial auctions. In: Proceedings of the 2nd ACM Conference on Electronic Commerce. ACM Press, Minneapolis, MN, pp 1-12, Minneapolis (2000)
17. Rothkopf, MH., Pekee, A., Ronald, M.: Computationally manageable combinatorial auctions. Management Science, 44(8): 1131-1147 (1998)

18. Sandholm, T.: Optimal winner determination algorithms. In: Cramton et al. (ed) Combinatorial auctions. MIT Press, Cambridge (2006)
19. Sandholm, T., Suri, S.: Improved optimal algorithm for combinatorial auctions and generalizations. In: Proceedings of the 17th National Conference on Artificial Intelligence, pp 90-97, USA (2000)
20. Sandholm, T., Suri, S., Gilpin, A., Levine, D.: CABoB: a fast optimal algorithm for combinatorial auctions. In: Proceedings of the International Joint Conferences on Artificial Intelligence, pp 1102-1108, Seattle, WA (2001)
21. Vries, S., Vohra, R.: Combinatorial auctions a survey. INFORMS Journal on Computing, 15: 284-309 (2003)
22. Wang, Y., Lu, Z., Glover, F., Hao, J.K.: Backbone guided tabu search for solving the UBQP problem. Journal of Heuristics, 19(4): 679-695 (2013