Solving the winner determination problem via a weighted maximum clique heuristic

Qinghua Wu[a], Jin-Kao Hao[b,*]

[a] *School of Management, Huazhong University of Science and Technology, No. 1037, Luoyu Road, Wuhan, China*
[b] *LERIA, Université d'Angers, 2 bd Lavoisier, 49045 Angers, Cedex 01, France*

Abstract

Combinatorial auctions (CAs) where bidders can bid on combinations of items is an important model in many application areas. CAs attract more and more attention in recent years due to its relevance to fast growing electronic business applications. In this paper, we study the winner determination problem (WDP) in CAs which is known to be NP-hard and thus computationally difficult in the general case. We develop a solution approach for the WDP by recasting the WDP into the maximum weight clique problem (MWCP) and solving the transformed problem with a recent heuristic dedicated to the MWCP. The computational experiments on a large range of 530 benchmark instances show that the clique-based approach for the WDP not only outperforms the current best performing WDP heuristics in the literature both in terms of solution quality and computation efficiency, but also competes very favorably with the powerful CPLEX solver.

Keywords: Winner determination; Combinatorial auctions; Tabu search; Maximum clique; Heuristics; Combinatorial optimization.

1. Introduction

Combinatorial auctions (CAs) is a type of auctions where bidders are allowed to buy entire bundles of goods (or items) in a single transaction (Cramton et al., 2006). One key issue in CAs is the winner determination problem (WDP) (Lehmann et al., 2006). Given a set of combinatorial bids, each bid being defined by a subset of items with a price, two bids are conflicting if they share at least one item. The 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. It is known that the WDP is equivalent to the maximum weight set packing problem (de Vries & Vohra, 2003), and can be reduced to the maximum

*Corresponding author.
Email addresses: wu@info.univ-angers.fr (Qinghua Wu), hao@info.univ-angers.fr (Jin-Kao Hao)

weight clique problem (MWCP). From the computational complexity point of view, the WDP belongs to the class of NP-complete problems (Rothkopf et al., 1998). From the practical point of view, the WDP finds many applications in, for instance, production management (Ray et al., 2011), intelligent transportation systems (Satunin & Babkin, 2014; de Vries & Vohra, 2003), electronic commerce (de Vries & Vohra, 2003), game theory (Fontanini & Ferreira, 2012), knowledge management (Wu, 2001), logistics services (Ignatius et al., 2011; Pla et al., 2014; de Vries & Vohra, 2003).

The WDP can be reduced to the maximum vertex weight clique problem (MWCP) (Ausiello et al., 1980). As a consequence, any solution method designed for the MWCP can be applied to solve the WDP via its clique formulation. This solution approach is extremely appealing since 1) we can solve the WDP without developing dedicated WDP algorithms, and 2) we can take full advantage of new algorithmic developments on the MWCP to better solve the WDP. Moreover, one can even apply different clique methods to enlarge the classes of the WDP instances that can be solved. As far as we know, this clique based approach for the WDP was not explored in the published literature.

The first objective of the paper is thus to investigate the strong connection between the WDP and the MWCP by carrying out an in-depth experimental assessment about the performance of this clique-based approach for the WDP. For this purpose, we adopt the recent multi-neighborhood tabu search heuristic (MN/TS) for the MWCP (Wu et al., 2012) and present extensive evaluations of this approach for the WDP both in terms of solution quality and computing efficiency. In particular, we provide computational results on three sets of well-known WDP test suites (for a total of 500+20+10=530 problem instances) which are commonly used in the literature. We show that this clique-based approach is clearly superior to the current best performing heuristics in the literature which are specially designed for the WDP. Moreover, it dominates the powerful CPLEX 12.4 solver on the realistic test suite and shows a competitive performance on the other two test suites.

The rest of the paper is organized as follows. In Section 2, we provide a review on existing algorithms for the WDP and summarize the main contributions of this work. In section 3, we present the formal definition of the WDP and the transformation of the WDP to the maximum weight clique problem. In section 4, we briefly recall the multi-neighborhood tabu search heuristic MN/TS for the MWCP. In section 5, we provide computational results and comparisons on a wide range of benchmark instances from the literature. In section 6, we offer some insights on the behavior of the clique approach. The last section is dedicated to conclusions and perspectives for future research.

2. Literature review and main contributions

The computational challenge of the WDP and its wide practical applications have motivated a variety of solution approaches, including both exact and heuristic methods.

Exact methods have the theoretical advantage of guaranteeing the optimality of the solution found, but they need a computing time which grows exponentially with the problem size in the general case. Still, highly effective exact algorithms are available in the literature for solving the WDP. Attempts to exactly solve the WDP (under the name of set packing) can be found as early as in the beginning of 1970s (Padberg, 1973). Many studies have appeared ever since. Most exact algorithms are based on the general branch-and-bound (B&B) framework. Representative examples include the combinatorial auction structural search (CASS) (Fujishima et al., 1999), the Combinatorial Auction Multi-Unit Search (CAMUS) (Leyton-Brown et al., 2000a; Leyton-Brown, 2003), the BOB algorithm (Sandholm & Suri, 2003), the CABOB algorithm (Sandholm et al., 2005), the linear programming based B&B algorithm (Nisan, 2000), and the clique-based B&B algorithm using graph coloring for bounding (Wu & Hao, 2014). Other interesting exact methods for the WDP are a branch-and-price algorithm based on a set packing formulation (Günlük et al., 2005), a branch-and-cut algorithm (Escudero et al., 2009), and a dynamic programming algorithm (Rothkopf et al., 1998). Finally, the general integer programming approach based on CPLEX was intensively studied in (Andersson et al., 2000; Guo et al., 2006; Sandholm et al., 2005), showing an excellent performance in many cases.

On the other hand, given the intrinsic intractability of the WDP, various heuristic algorithms have been devised to handle problems whose optimal solutions cannot be reached by exact approaches. For instance, Casanova (Hoos & Boutilier, 2000) is a well-known stochastic local search algorithm which explores the space of feasible allocations (non-overlapping subsets of bids) by adding at each step an unallocated bid and removing from the allocation the bids which are conflicting with the added bid. The selection rule employed by Casanova takes into consideration of both the quality and history information of the bid. Casanova is shown to be able to find high quality solutions much faster than the CASS algorithm (Fujishima et al., 1999). The WDP is also modeled as a set packing problem and solved by a simulated annealing algorithm (SAGII) with three different local move operators: an embedded branch-and-bound move, greedy local search move and exchange move (Guo et al., 2006). SAGII outperforms dramatically Casanova and the CPLEX 8.0 solver for realistic test instances. A memetic algorithm is proposed by Boughaci et al. (2009), which combines a local search component with a specific crossover operator. The local search component adds at each iteration either a random bid with a probability p or a best bid with the largest profit with probability $1 - p$, and then removes the conflicting bids from the allocation. This hybrid algorithm reaches excellent results on the tested realistic instances. Other interesting heuristics include greedy algorithms (Lau & Goh, 2002; Mito & Fujita, 2004), a tabu search algorithm (Boughaci et al., 2010), an equilibrium-based local search method (Tsung et al., 2011) and a recombination-based tabu search algorithm (Sghir et al., 2014).

From the above review, we observe that the existing (exact and heuristic) methods follow two solution strategies. The first one is to consider directly the WDP and design dedicated algorithms. This is the case for most of the reviewed

methods. The second one is to recast the WDP as another related problem P and then solved with a solution method designed for P. Examples are shown in Guo et al. (2006) and Padberg (1973) where the WDP is modeled as the set packing problem and in Andersson et al. (2000), Guo et al. (2006) and Sandholm et al. (2005) where the WDP is reformulated as an integer programming problem and solved by the general CPLEX solver.

Compared with the existing studies on the WDP, this work has the following main contributions:

First, we handle the WDP by recasting it as a weighted maximum clique problem and applying an effective clique heuristic to solve the problem. To our knowledge, this is the first study formally investigating the strong connection between the WDP and the MWCP and presenting extensive computational assessments of the clique based approach to the WDP.

Second, this study discloses that the clique based approach is well suited for the WDP, and is able to delivery very competitive and even better results than the current best performing WDP heuristics which are specially designed for the problem. This is particularly true for the cases where each bid contains a relatively long list of item.

Third, from a more technical perspective, it is well known that move operators play a key role to the performance of a local search algorithm. Most of the currently best WDP heuristics rely only on a basic 'add-and-repair' operator which first adds an unallocated bid to the current allocation and then removes the conflicting bids from the allocation. From the clique point of view, this basic move operator is quite limited and effective clique algorithms employ more complicated operators like add, swap and drop (Pullan, 2006; Pullan & Hoos, 2006; Wu & Hao, 2014). This study indirectly demonstrates the usefulness of these combined move operators for the WDP problem, promoting the idea that to design effective WDP heuristics, it would be relevant to integrate similar combined operators.

3. Winner determination problem (WDP)

The optimal winner determination problem in CAs can be defined as follows. Let $M = \{1, 2, \ldots, m\}$ be the set of m items to be sold by the auctioneer, and let $B = \{B_1, B_2, \ldots, B_n\}$ be the set of bids submitted by the buyers. Each bid can be denoted by a couple (S_i, P_i), where $S_i \subseteq M$ is a set of items and P_i is the global price of the items in S_i. Let B be a $m \times n$ binary matrix such that $B_{ij} = 1$ if object $j \in S_i$, $B_{ij} = 0$ otherwise. Furthermore, define $x_i = 1$ if the bid B_i is accepted (a winning bid), and zero otherwise (a losing bid). Then the winner determination problem (WDP) is to label the bids as winning or losing so as to maximize the auctioneer's revenue under the constraint that each item can be allocated to at most one bidder. More formally, the WDP problem can be modeled as the following integer programming formulation.

$$\max \sum_{i=1}^{n} P_i x_i \tag{1}$$

$$\text{subject to} \sum_{i=1}^{n} B_{ij} x_i \leq 1, j \in \{1, \ldots, m\} \tag{2}$$

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

The objective function (1) is to maximize the auctioneer's revenue which is equal to the sum of the prices of the winning bids. The constraint (2) establishes that no item is allocated to more than one bid. Due to the free disposal assumption, some items could be left uncovered. One notices that this formulation is widely studied in the literature (Andersson et al., 2000; Boughaci et al., 2009, 2010; Guo et al., 2006; Hoos & Boutilier, 2000; Sandholm, 2002; Sandholm & Suri, 2003; Sandholm et al., 2005).

The WDP can be conveniently transformed to the maximum (vertex) weight clique problem. To see this, we first recall some basic notations related to the MWCP.

Let $G = (V, E, \omega)$ be an undirected graph with vertex set $V = \{1, \ldots, n\}$, edge set $E \subset V \times V$, and vertex weighting function ω which associates to each vertex i a positive weight w_i. A clique C of G is a subset of V such that every two vertices are pairwise adjacent, i.e., $\forall u, v \in C, \{u, v\} \in E$. For a clique C of G, define its weight as the sum of the weights of all vertices in C, i.e., $W(C) = \sum_{i \in C} w_i$. Then the maximum weight clique problem (MWCP) is to determine a clique of maximum weight.

For a given WDP instance $B = \{B_1, B_2, \ldots, B_n\}$ with $B_i = (S_i, P_i)$ $(1 \leq i \leq n)$, we define a maximum weight clique instance $G = (V, E, \omega)$ as follows.

- For a bid $B_i \in B$, define a vertex $i \in V$ with the weight w_i set to be equal to P_i. That is, $V = \{1, 2, \ldots, n\}, \forall i \in V, w_i = P_i$.
- Define the edge matrix E by: $e_{ij} = \begin{cases} 1, & \text{if } S_i \cap S_j = \emptyset, i, j \in \{1, \ldots, n\} \\ 0, & \text{otherwise.} \end{cases}$

That is, any two vertices i and j are connected by an edge if the corresponding bids $B_i = (S_i, P_i)$ and $B_j = (S_j, P_j)$ share no common item, which indicates that these two bids can be accepted together. The edge matrix can be directly used to detect the conflict bids that share at least an item.

Now it is easy to observe that a maximum weight clique $C = \{i_1, \ldots, i_r\}$ of the graph $G = (V, E, \omega)$ corresponds to a feasible subset $\{B_{i_1}, \ldots, B_{i_r}\}$ of B with a maximum revenue, i.e., the optimal solution of the WDP, implying that any solution method for the MWCP can be used to solve the WDP.

To illustrate the transformation, we consider a WDP instance with 8 items and 7 bids (Fig. 1 (left)). The associated graph $G = (V, E, \omega)$ with respect to the WDP instance is shown in Figure 1 (right) where each bid is represented by a vertex with its weight equal to the price of the bid. Two vertices are linked by an edge if the two corresponding bids do not share any item. It is clear that the optimal solution to the MWCP defined by the graph is given by the set vertices $\{3, 4, 6\}$ which represents the set of winning bids {Bid3, Bid4, Bid6} with a maximum revenue of 70.

Generally, one observes that given a set M of items, if the candidate bids contain many (few) items, the edge density of the associated graph will have a low (high) edge density (with respect to the complete graph). As we show in Sections 5 and 6, graph density may influence the performance of an solution method, which is particularly true for the clique based approach studied in this work.

4. Multi-neighborhood tabu search for the WDP

Given a WDP instance formulated as a MWCP instance (see last section), we can solve the problem with any available algorithm designed for the MWCP. In this work, we adopt a recent heuristic algorithm called "multi-neighborhood tabu search (MN/TS)" (Wu et al., 2012), which shows excellent performances on the MWCP problem as well as on the related set packing problem. In this section, we briefly review the main ingredients of the MN/TS algorithm. For a thorough presentation, the reader is referred to Wu et al. (2012).

4.1. Search space, evaluation function, and multi-start strategy

For a given MWCP instance $G = (V, E, w)$, the search space Ω explored by the MN/TS algorithm is composed of all possible cliques of G. For a given solution $C \in \Omega$, its quality (i.e., objective value) is evaluated by its weight $W(C) = \sum_{i \in C} w_i$.

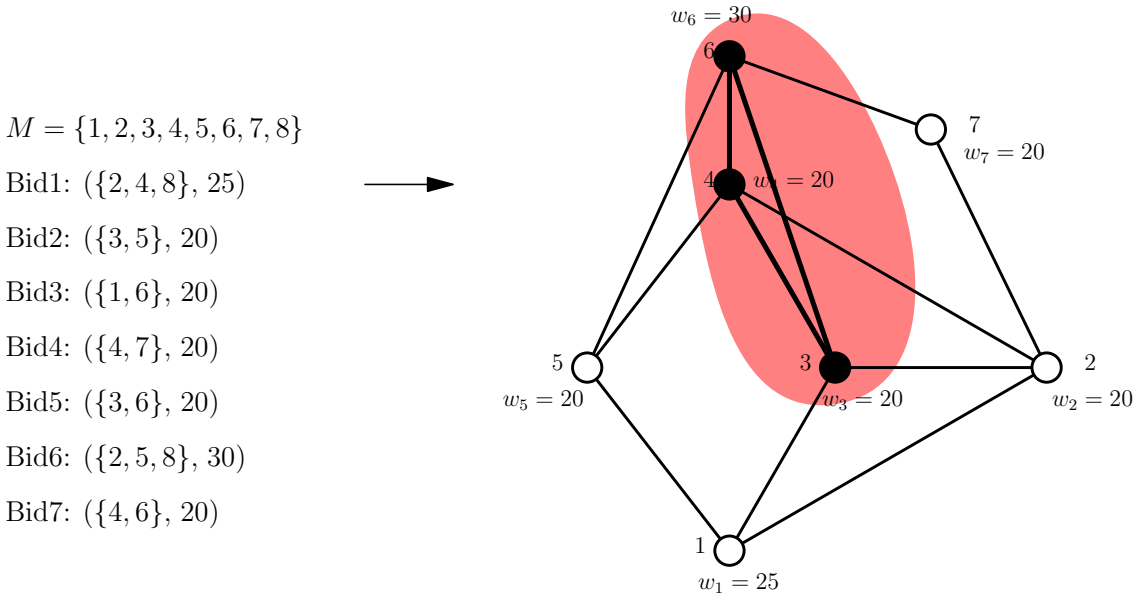

Figure 1: The original MDP instance (left) and the associated MWCP instance $G = (V, E, \omega)$ (right).

Our MN/TS algorithm is a multi-start tabu search procedure. Each time MN/TS starts from an initial solution C, and then applies the tabu search procedure to improve C by maximizing its weight. If the weight of the current clique C is not improved for L consecutive iterations (L is called the search depth), the current round of tabu search is stopped to restart from a new initial solution. The initial solution C is constructed by first initiating C with a randomly selected seeding vertex and then repeatedly adding to C a vertex v which is absent from C and connected to all vertices of C until no such vertex v exists.

4.2. Basic move operators and neighborhoods

To explore the search space, the MN/TS algorithm jointly employs three neighborhoods which are defined by three basic move operators, denoted by ADD, $SWAP$ and $DROP$ respectively. These operators rely on the definition of two vertex subsets: PA and OM relative to a given solution C, which are defined as follows.

- PA is the set of the vertices that are excluded from the clique C and connected to *all* the vertices of C: $PA = \{v : v \in V \backslash C, \{v, i\} \in E, \forall i \in C\}$.
- OM is composed of the vertices that are excluded from the clique C and connected to *all but one* vertex of C: $OM = \{v : v \in V \backslash C, |A(v) \bigcap C| = |C| - 1\}$ where $A(v) = \{j : j \in V, \{j, v\} \in E\}$ is the set of vertices which are adjacent to v. An illustration of the relationship between a clique C and its two associated subsets PA and OM is given in Figure 2.

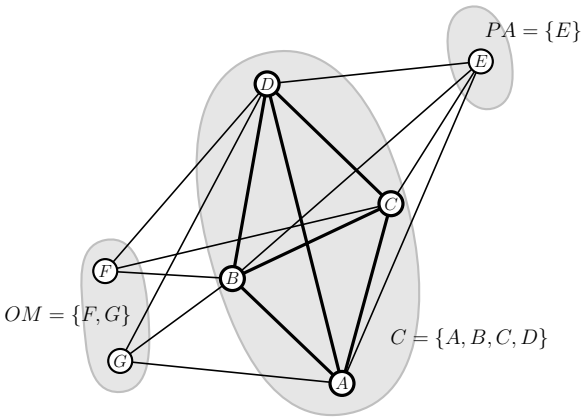

Figure 2: A clique and its two associated subsets: $C = \{1, 2, 3, 4\}$, $PA = \{5\}$ and $OM = \{6, 7\}$.

Then the add move operator ($ADD(i)$) displaces a vertex i from the set PA to the current clique C (This operator is applied only when the subset PA is not empty).

After an $ADD(i)$ move, the change in the clique weight, also known as the move gain value Δ_i, can be computed by the following expression:

$$\Delta_i = w_i \tag{4}$$

where w_i is the weight associated to the newly added vertex i. We can see that an add move always leads to an increase to the objective function value.

The swap move operator ($SWAP(i, j)$, which applies when OM is not empty) exchanges a vertex i in OM against the only vertex j of C which is not adjacent to i. For a given $SWAP(i, j)$ move, the move gain Δ_{ij} can be conveniently calculated by the following formula:

$$\Delta_{ij} = w_i - w_j \tag{5}$$

Since Δ_{ij} can be either positive or negative, a $SWAP$ move can improve or deteriorate the quality of the current solution.

Finally, the drop move operator ($DROP(i)$), consists in removing a vertex i from the current clique C. For a given dropping move, the move gain value induced by $DROP(i)$ can be easily computed by:

$$\Delta_i = -w_i \tag{6}$$

We observe that the drop operator always leads to a decrease of the clique weight.

4.3. Combined neighborhood and neighbor selection strategy

For the case of the maximum weight clique problem, we can see that a $SWAP$ move may lead to a solution which is better than any solution obtained with an ADD move. Figure 3 provides an illustrative example to show such a situation. Similarly, when no ADD move is possible, a $DROP$ move may lead to a solution better than any solution that can be obtained with a $SWAP$ move. To summarize, for the MWCP problem, there is no absolute dominance of one move operator over another move operator. The best move operator to be applied depends on the current search context and should be determined according to the search context. At each iteration of the tabu search procedure, our MN/TS algorithm jointly considers these three neighborhoods and selects the most favorable move (i.e., with the largest Δ value) among all the ADD, $SWAP$ and $DROP$ moves to generate the next solution (Ties are broken randomly).

Finally, to prevent the previously visited solutions from being revisited and to encourage the search to explore new regions of the search space, a simple prohibition rule is employed by MN/TS: vertices that leave the current clique C are forbidden to move back to C during the prohibition period (called the tabu tenure) while vertices that join the clique C can be removed from C without restriction.

5. Computational experiments

In this section, we present an extensive assessment about the capacity of the MN/TS algorithm to solve the WDP as a MWCP. For this purpose, we show experimental results obtained by MN/TS on a large collection of popular WDP benchmark instances and make comparisons with the CPLEX 12.4 slover as well as the best performing WDP heuristic algorithms published in the literature. The MN/TS algorithm is programmed in C and compiled using GNU GCC. All the experiments were carried out on a PC running Windows XP with an Intel Xeon E5440 processor (2.83 GHz and 8 GB of RAM).

5.1. Benchmarks

To evaluate the efficiency of the clique approach (with the MN/TS algorithm), we use three test suites of 530 benchmark instances. The first set of 500 instances is pre-generated problem instances, while the second and third sets (20 and 10 instances respectively) are generated randomly from generators for combinatorial auctions according to several distributions. The characteristics of these instance sets are summarized below and details can be found in Lau & Goh (2002), Leyton-Brown et al. (2000b) and Sandholm (2002).

The first set of 500 benchmark instances is provided by Lau & Goh (2002) with up to 1500 items and 1500 bids. These instances are considered in Lau & Goh (2002) to be more realistic than other instances and are divided into 5 different groups, each group having 100 instances labeled as REL-m-n, where m is the number of items and n is the number of bids. To generate these instances,

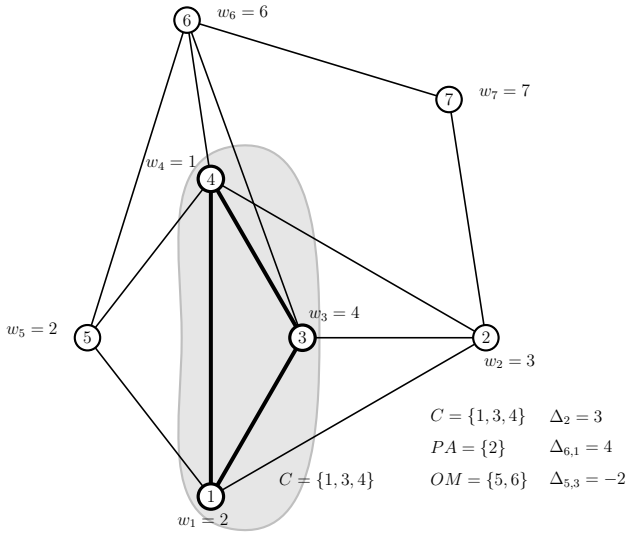

Figure 3: From clique $C = \{1, 3, 4\}$, the $SWAP$ move between vertices 6 and 1 ($SWAP(6, 1)$) leads to a solution $C_1 = \{3, 4, 6\}$, which is better than the solution $C_2 = \{1, 2, 3, 4\}$ obtained by the ADD move ($ADD(2)$).

several factors are incorporated, including a pricing factor which models a bidder's acceptable price range for each bid, a preference factor which takes into account bidder's preferences among bids, and a fairness factor which measures the fairness in distributing jobs among bidders. More details about how these instances are generated can be found in Lau & Goh (2002).

The second test set of 20 instances are obtained from a generator provided by Sandholm (2002), which can be used to generate instances of different problem sizes and distributions. There are four different distributions available:

- $Random(m, n)$: The n bids are generated as follows. For each bid, first pick the number of items randomly from {1,2,...,m}. Then randomly choose that many items without replacement from {1,2,...,m}. Finally pick the price randomly from a uniform distribution on $[0, 1]$.
- $Weighted\ Random(m, n)$: As above, but pick the price between 0 and the number of items in the bid.
- $Uniform(m, n, \lambda)$: The n bids are generated as follows. For each bid, first randomly choose λ items without replacement from {1,2,...,m}, and then pick a price randomly from a uniform distribution on $[0, 1]$.
- $Decay(m, n, \alpha)$: Generate n bids as follows. For each bid, give a first item randomly selected from {1,2,...,m}. Then repeatedly add a new random item randomly selected from {1,2,...,m} (without replacement) with probability α until an item is not added or the bid includes all m items. Pick the price between 0 and the number of items in the bid. In our experiments, α is set equal to 0.75, since as indicated in Sandholm (2002), this setting leads to the hardest instances on average (at least for the algorithm in Sandholm (2002)).

The third set of problem instances are generated randomly by the program combinatorial auction test suite (CATS) generator developed by Leyton-Brown et al. (2000b). Different distributions are available in the CATS suite: paths, regions, matching, scheduling, and arbitrary. For each one of these distributions, we used the CATS default parameters.

All the instances used in our experimental evaluations will be made available at `http://www.info.univ-angers.fr/pub/hao/wdp.html`.

5.2. Parameter settings and stop condition

The main parameter required by the MN/TS algorithm is L which is the number of consecutive iterations for which the current solution cannot be further improved. Notice that given a fixed computing time, a small (large) value of L leads to more (less) frequent restarts of the tabu search procedure. In our experiment, we simply set L to 4000 as indicated in Wu et al. (2012). Given the stochastic nature of our MN/TS procedure, each instance is solved 100 times independently with different random seeds. For each run, the time limit is set to be 5 minutes.

Table 1: MN/TS versus MA and CPLEX 12.4 on some of the REL-500-1000 instances

Instance	Density	MN/TS				MA (Boughaci et al., 2009)		CPLEX	
		W	W_{avg}	hit	Time	W	Time	W	Time
in101	0.31	72724.61	72724.61	100	5.46	67101.93	129.62	67101.94	3600
in102	0.29	72518.22	72518.22	100	19.91	67797.61	132.18	70292.58	3600
in103	0.30	72129.50	72129.50	100	18.52	66350.99	133.34	69703.05	3600
in104	0.30	72709.64	72709.64	100	7.33	64618.41	135.14	71579.58	3600
in105	0.29	75646.12	75646.12	100	1.70	66376.83	153.96	68431.12	3600
in106	0.29	71258.61	71258.61	100	9.05	65481.64	140.96	66621.12	3600
in107	0.30	69713.40	69713.40	100	1.94	66245.70	146.40	69182.50	3600
in108	0.31	75813.20	75803.35	73	113.53	74588.51	161.03	74637.79	3600
in109	0.29	69475.89	69475.89	100	4.96	62492.66	144.71	65901.61	3600
in110	0.29	68295.28	68295.28	100	26.10	65171.19	149.01	67618.87	3600
in111	0.30	75133.29	75133.29	100	2.90	72969.16	157.34	72242.28	3600
in112	0.30	71342.48	71342.48	100	16.33	66671.67	151.40	70588.82	3600
in113	0.31	73365.87	73365.87	100	6.24	68901.96	165.26	70475.80	3600
in114	0.30	69224.75	69224.75	100	3.72	64190.63	160.00	66757.96	3600
in115	0.30	70221.56	70221.56	100	32.42	62052.25	148.03	66149.07	3600
in116	0.31	70032.43	70032.43	100	3.17	64849.85	162.54	69308.00	3600
in117	0.29	69982.83	69982.83	100	3.79	66466.39	152.85	69923.79	3600
in118	0.31	72160.98	72160.98	100	24.98	69239.96	159.06	72160.98	3600
in119	0.30	67038.42	67038.42	100	30.59	63968.32	153.84	64934.13	3600
in120	0.32	75514.93	75514.93	100	2.07	68587.41	166.82	74658.12	3600
Average		71715.10	71714.60		16.73	66706.15	150.17	69413.45	3600

Table 2: MN/TS versus MA and CPLEX 12.4 on some REL-1000-1000 instances

Instance	Density	MN/TS				MA (Boughaci et al., 2009)		CPLEX	
		W	W_{avg}	hit	Time	W	Time	W	Time
in201	0.15	81557.74	81557.74	100	9.45	77499.82	98.26	79466.83	3600
in202	0.15	90708.12	90708.12	100	2.47	90464.19	106.68	90537.28	3600
in203	0.16	86239.21	86239.21	100	3.88	86239.21	102.28	86239.21	3600
in204	0.16	87075.42	87075.42	100	2.67	81969.046	97.40	87075.42	3600
in205	0.15	86515.95	86515.95	100	1.73	82469.19	91.26	84016.43	3600
in206	0.15	91518.96	91518.96	100	0.75	86881.42	93.99	86888.23	3600
in207	0.16	93129.24	93129.24	100	11.23	91033.51	100.90	89085.69	3600
in208	0.15	94904.67	94904.67	100	4.24	83667.76	101.29	91782.04	3600
in209	0.15	87268.96	87268.96	100	11.25	81966.65	96.42	83166.69	3600
in210	0.15	89962.39	89962.39	100	7.64	85079.98	97.78	86940.49	3600
in211	0.15	84913.68	84913.68	100	2.96	79746.14	90.78	84028.31	3600
in212	0.16	90778.20	90778.20	100	0.51	81061.38	103.45	85390.73	3600
in213	0.16	85369.18	85369.18	100	1.88	83549.21	101.56	83501.07	3600
in214	0.15	85181.60	85181.60	100	0.63	81935.32	95.06	83554.16	3600
in215	0.17	91531.69	91531.69	100	1.50	83663.13	102.48	85965.20	3600
in216	0.16	91580.93	91580.93	100	0.45	83286.63	100.93	85656.94	3600
in217	0.13	86962.92	86962.92	100	0.60	83125.25	90.34	86962.92	3600
in218	0.16	94965.19	94965.19	100	0.68	86936.78	105.06	88300.26	3600
in219	0.15	93586.43	93586.43	100	1.15	88054.21	93.35	86006.20	3600
in220	0.17	89792.90	89792.90	100	3.84	86937.85	104.35	87883.45	3600
Average		89177.16	89177.16		3.47	84278.33	98.68	86122.37	3600

Table 3: MN/TS versus MA and CPLEX 12.4 on some of the REL-1000-500 instances

Instance	Density	MN/TS				MA (Boughaci et al., 2009)		CPLEX	
		W	W_{avg}	hit	Time	W	Time	W	Time
in401	0.14	77417.48	77417.48	100	0.16	72948.07	37.07	77417.48*	24.26
in402	0.14	76273.33	76273.33	100	0.38	71454.78	37.20	76273.33*	30.35
in403	0.15	74843.95	74843.95	100	3.02	74843.96	38.81	74843.95*	35.43
in404	0.16	78761.69	78761.69	100	0.87	78761.68	38.78	78761.69*	33.76
in405	0.16	75915.90	75915.90	100	0.39	72674.25	39.29	75915.90*	67.49
in406	0.14	72863.32	72863.32	100	0.64	71791.03	38.09	72863.32*	55.51
in407	0.17	76365.71	76365.71	100	0.26	73935.28	40.95	76365.71*	77.82
in408	0.15	77018.83	77018.83	100	0.16	72580.04	39.07	77018.83*	124.24
in409	0.13	73188.62	73188.62	100	0.38	68724.53	36.28	73188.62*	51.24
in410	0.16	73791.65	73791.65	100	0.51	71791.57	41.90	73791.65*	50.81
in411	0.15	73935.40	73935.40	100	0.58	71200.55	38.76	73935.40*	27.10
in412	0.16	75292.63	75292.63	100	0.13	75292.63	37.17	75292.63*	69.12
in413	0.16	74434.99	74434.99	100	0.32	73350.87	40.95	74434.99*	62.49
in414	0.17	77146.37	77146.37	100	0.21	77146.36	41.26	77146.37*	74.75
in415	0.14	73519.12	73519.12	100	0.24	71926.73	36.32	73519.12*	68.57
in416	0.16	73487.01	73487.01	100	0.18	72520.66	39.81	73487.01*	56.36
in417	0.15	74981.35	74981.35	100	0.51	74680.99	39.29	74981.35*	49.13
in418	0.14	71404.84	71404.84	100	1.68	71404.84	40.00	71404.84*	28.75
in419	0.15	72505.21	72505.21	100	0.10	70472.84	38.45	72505.21*	40.36
in420	0.15	75510.68	75510.68	100	0.17	71381.02	37.65	75510.68*	54.17
in421	0.16	75694.94	75694.94	100	0.12	75694.94	38.78	75694.94*	27.50
in422	0.15	77443.90	77443.90	100	0.21	72850.90	37.36	77443.90*	28.30
in423	0.13	68134.35	68134.35	100	0.17	68134.35	36.17	68134.35*	35.17
in424	0.17	77352.75	77352.75	100	1.56	73196.15	43.26	77352.75*	49.28
in425	0.17	77333.91	77333.91	100	0.15	73258.59	41.35	77333.91*	49.21
in426	0.17	76430.18	76430.18	100	0.37	74524.80	39.67	76430.18*	211.26
in427	0.15	76387.56	76387.56	100	0.12	73147.95	36.96	76387.56*	57.56
in428	0.15	77384.94	77384.94	100	0.29	76554.58	38.64	77384.94*	52.00
in429	0.15	75540.96	75540.96	100	0.12	75540.96	39.87	75540.96*	61.19
in430	0.16	79038.75	79038.75	100	0.71	76264.92	39.60	79038.75*	66.95
Average		75313.34	75313.34		0.49	73268.36	38.95	75313.34	57.33

Table 4: MN/TS versus MA and CPLEX 12.4 on some of the REL-1500-1500 instances

Instance	Density	MN/TS				MA (Boughaci et al., 2009)		CPLEX	
		W	W_{avg}	hit	Time	W	Time	W	Time
in601	0.09	108800.44	108800.44	100	9.12	99044.32	110.62	105286.85	3600
in602	0.08	105611.47	105611.47	100	1.72	98164.23	114.18	99254.88	3600
in603	0.08	105121.02	105121.02	100	1.21	94126.96	110.71	101270.04	3600
in604	0.09	107733.80	107733.80	100	16.62	103568.86	110.60	105185.67	3600
in605	0.09	109840.98	109840.98	100	2.72	102404.76	122.40	103694.50	3600
in606	0.09	107113.06	107113.06	100	1.47	104346.07	107.79	107113.06	3600
in607	0.09	113180.28	113180.28	100	1.37	105869.44	113.26	103095.66	3600
in608	0.09	105266.10	105266.10	100	1.12	95671.77	109.15	99490.66	3600
in609	0.09	109472.33	109472.33	100	1.34	98566.94	111.12	100895.86	3600
in610	0.10	113716.96	113716.96	100	2.67	102468.60	120.17	113716.96	3600
in611	0.09	106666.32	106666.32	100	2.50	98974.64	107.98	106666.32	3600
in612	0.09	109796.70	109796.70	100	6.85	106056.07	122.81	109796.70	3600
in613	0.09	107980.15	107980.15	100	1.52	93289.85	120.14	99328.57	3600
in614	0.10	108364.57	108364.57	100	7.72	97510.72	122.51	100513.13	3600
in615	0.08	110508.81	110508.81	100	4.27	101770.70	108.67	104433.21	3600
in616	0.09	109740.48	109740.48	100	4.61	100169.53	109.65	108139.54	3600
in617	0.09	113302.43	113302.43	100	5.80	100653.88	114.98	105899.16	3600
in618	0.10	111385.08	111385.08	100	6.60	102378.27	120.71	105154.80	3600
in619	0.09	107571.59	107571.59	100	2.79	97306.30	115.00	98035.64	3600
in620	0.09	110937.97	110937.97	100	3.00	102951.68	115.79	101712.44	3600
Average		109105.52	109105.52		4.25	100264.67	114.41	103934.18	3600

Table 5: MN/TS versus MA and CPLEX 12.4 on some of the REL-1000-1500 instances

Instance	Density	MN/TS				MA (Boughaci et al., 2009)		CPLEX	
		W	W_{avg}	hit	Time	W	Time	W	Time
in501	0.08	88656.95	88656.95	100	1.47	79132.03	107.82	88656.95	3600
in502	0.08	86236.91	86236.91	100	1.76	80340.76	108.71	83757.54	3600
in503	0.07	87812.37	87812.37	100	19.63	83277.71	114.15	86318.17	3600
in504	0.10	85600.00	85600.00	100	4.62	81903.02	116.11	84220.22	3600
Average		87076.55	87076.55		6.87	81163.38	111.69	85738.22	3600

Table 6: Comparison of MN/TS with MA, SAGII and Casanova on 14 REL instances. The results of MA, SAGII and Casanova are from Table 7 in Boughaci et al. (2009)

Instance	MN/TS		MA (Boughaci et al., 2009)		SAGII (Guo et al., 2006)		Casanova (Hoos & Boutilier, 2000)	
	W	Time	*W*	Time	*W*	Time	*W*	Time
in202	90708.12	2.47	90464.19	106.68	86179.64	45.19	52048.73	113.55
in207	93129.24	11.23	91033.51	100.90	88513.37	44.58	51003.39	132.92
in208	94904.67	4.24	83667.76	101.29	83500.82	44.80	51340.27	111.16
in227	92613.37	5.23	83667.76	101.29	86747.23	44.58	54998.44	117.16
in501	88656.95	1.47	79132.03	107.82	85101.43	71.05	53992.12	164.94
in525	93651.06	3.83	85143.13	109.96	88086.29	67.56	58365.02	164.66
in593	94044.63	1.39	80304.07	106.89	82046.16	69.03	58527.76	171.72
in594	91334.89	2.56	86112.90	103.56	82341.22	68.11	55821.04	171.80
in600	91538.68	3.78	82555.91	95.46	83772.91	67.09	56001.82	174.25
in664	112866.86	6.84	102905.25	123.81	104346.07	90.73	65543.41	161.11
in688	112012.55	7.51	103742.53	114.81	106056.08	90.95	64962.00	166.88
in694	114477.05	7.17	108114.12	107.87	105699.93	91.09	70140.69	170.56
in699	107132.33	5.19	103762.70	120.71	103252.95	90.42	65026.40	171.70
in700	106730.67	4.23	101510.20	117.42	105462.71	91.22	62404.80	160.98
Average	98842.93	4.79	91579.71	108.46	92221.91	69.74	58583.79	153.81

5.3. Experimental results on the REL instances

In this section, we present the computational results obtained by the MN/TS algorithm on the REL instances. To assess the relative quality of our results, we compare MN/TS with the CPLEX 12.4 solver and the recent memetic algorithm (MA) (Boughaci et al., 2009) which is one of the current best performing heuristics for these instances. MA was run on a Pentium-IV processor (2.8 GHZ CPU and 1 GB RAM). According to the Standard Performance Evaluation Cooperation (www.spec.org), this computer is as fast as the computer we used for our experiments. To obtain the results of the CPLEX 12.4 solver on our computer, we use the integer programming formulation (Eq. (1)-(3)) as described in Section 3 and run CPLEX until a timeout limit of 3600 seconds is reached, or until the memory limits is reached. Previous studies show that general ILP solvers like CPLEX represent a highly effective approach for many cases of the WDP problem (see for instance Andersson et al. (2000) and Sandholm et al. (2005)) and thus can be considered as a valuable reference for our comparative study. Since there are 500 REL instances, we show first in Tables 1-6, like previous studies in the literature (Boughaci et al., 2009; Guo et al., 2006), the detailed results on some selected instances from each of the five groups, and then in Table 7 the global results for the five groups of 500 instances.

Tables 1-5 present the detailed computational results of the MN/TS algorithm in comparison with those of the reference algorithm (MA) reported in Boughaci et al. (2009) as well as those obtained with the CPLEX 12.4 solver. Each table is dedicated to one of the 5 groups of the REL instances. The first column in these tables indicates the name of the instance. Column 2 indicates the density of the transformed graph. Columns 3 to 7 show the results obtained

by the MN/TS algorithm including the maximum revenue (W) obtained by the MN/TS algorithm over the 100 independent trials, the average revenue (W_{avg}) over the 100 trials, the success rate (hit) for reaching W_{avg} and the average CPU time in seconds ($Time$) over the 100 trials on which the W value is reached. Columns 7 to 8 present MA's results, which are directly extracted from the original paper (Boughaci et al., 2009). The last two columns report the results obtained with the CPLEX 12.4 solver within a time limit of 3600 seconds. Finally, the last row "Average" indicates the summary of our algorithm's average performance with reference to the MA algorithm and the CPLEX 12.4 solver.

From Tables 1-5, one observes that, for 86 of the 94 selected instances where the MA algorithm reports its results, the MN/TS algorithm outperforms MA both in terms of solution quality and computing time. For the remaining 8 instances on which the two algorithms attain the same revenue, MN/TS needs much less computing time than MA to reach its objective values. Even the average values (W_{avg}) of MN/TS are better than the best results obtained by MA. We can conclude that the MN/TS algorithm dominates the reference MA algorithm both in terms of solution quality and computing time.

Concerning the results obtained by the CPLEX 12.4 solver within a time limit of 3600 seconds, we can observe from Tables 1-5 that for 30 (REL-1000-500, Table 3) of the 94 REL instances, the CPLEX 12.4 solver is able to find optimal solutions (marked with a $*$ symbol when proven to be optimal by CPLEX). For 55 of the 94 instances, MN/TS is able to reach a larger revenue than the CPLEX 12.4 solver. For the remaining 39 instances, both the MN/TS algorithm and the CPLEX 12.4 solver attain the same revenue, however, MN/TS needs much less computing time than the CPLEX 12.4 solver for each of these instances. We can see that MN/TS performs much better than the CPLEX 12.4 solver on the REL benchmark instances.

For all these instances, we observe that the results of MN/TS are very stable. Except for one instance, MN/TS attains its best objective value for each of the 100 runs (see column hit).

To augment the above comparison, Table 6 compares the results of the MN/TS algorithm with those obtained by the MA algorithm as well as two additional effective heuristic approaches in the literature: SAGII (Guo et al., 2006) and Casanova (Hoos & Boutilier, 2000). The results for these 3 reference algorithms are extracted from Table 7 in Boughaci et al. (2009). From Table 6, it is observed that the MN/TS algorithm outperforms these 3 reference algorithms both in terms of solution quality and computing time. For each of these 14 instances, the MN/TS algorithm attains a much larger objective value within significantly shorter computing times compared with each of these 3 reference algorithms.

To further confirm the effectiveness of the MN/TS algorithm, we summarize in Table 7 the averaged results obtained by the MN/TS algorithm as well as the three reference algorithms for the 500 instances of the five groups. In Table 7, the column μ_W corresponds to the arithmetic average revenue of the 100 instances in each group and the column μ_{Time} gives the average time in second. The results of Table 7 clearly show a better performance of MN/TS compared to the

Table 7: Comparison of MN/TS with MA, SAGII and Casanova on the five groups of 500 REL instances. The results of the reference algorithms are from Boughaci et al. (2009)

Instance	ins	MN/TS		MA (Boughaci et al., 2009)		SAGII (Guo et al., 2006)		Casanova (Hoos & Boutilier, 2000)	
		μ_W	μ_{Time}	μ_W	μ_{Time}	μ_W	μ_{Time}	μ_W	μ_{Time}
REL-500-1000	100	71470.93	12.28	67520.23	477.22	64922.02	38.06	37053.78	119.46
REL-1000-500	100	75540.68	0.38	74149.49	101.12	73922.10	24.46	51248.79	57.74
REL-1000-1000	100	89158.98	3.12	84926.39	281.63	83728.34	45.37	51990.91	111.42
REL-1000-1500	100	89552.18	6.39	80805.98	297.35	82651.49	68.82	56406.74	168.24
REL-1500-1500	100	108627.17	2.64	102234.80	321.27	101739.64	91.78	65661.03	165.92
Average		86869.98	4.96	81927.37	295.71	81392.78	53.69	52472.25	124.55

competing methods. For each of the 5 groups of instances, the MN/TS algorithm achieves a much larger average revenue when compared with each of the three reference algorithms. In particular, MN/TS produces significantly better results than MA and SAGII which are currently among the most successful algorithms in the literature for the REL instances while the historical Casanova method is largely dominated by the other algorithms. This experiment demonstrates thus the effectiveness of the MN/TS algorithm for the whole set of REL instances.

5.4. Experimental results on the Sandholm benchmark sets

To further assess the performance of the clique approach for the WDP, we test the MN/TS algorithm on 20 instances which covers all of the benchmark distributions presented in Sandholm (2002) and compare our results with those obtained by the CPLEX 12.4 solver. For each of these distributions (random, weighted random uniform and decay), five benchmark instances are generated by fixing the number of bids equal to 2000 and varying the number of items from 100 to 500. For the uniform distribution, each bid consists of 10 items. Notice that some heuristic approaches like Casanova also reported their results on instances produced by the Sandholm's generator, however, due to the randomness of the generator, it is impossible to exactly replicate their test instances. As a consequence, it is difficult to objectively compare results with those approaches. For this study, CPLEX is again used as our reference approach.

Table 8 presents the detailed computational results of the MN/TS algorithm in comparison with those of the CPLEX 12.4 solver with a time limit of 3600 seconds. The last column δ indicates the deviation of the MN/TS algorithm with respect to CPLEX. The deviation is calculated as follows: $(W - CPLEX)/CPLEX$. The averaged results over the 20 instances are presented in the last row. From Table 8, we observe that for all these instances except for the 5 uniform distribution instances, CPLEX is able to obtain optimal solutions within 3600 seconds. For the 5 random distribution instances, MN/TS is able to reach the optimal solutions and it is faster than CPLEX. For the 5 weighted random distribution instances, MN/TS can also successfully achieve the optimal solutions with similar computing times. For the 5 uniform distribution instances where CPLEX fails to obtain an optimal solution within 3600

Table 8: MN/TS versus CPLEX on the Sandholm benchmark set

Instance	Density	MN/TS				CPLEX		δ
		W	W_{avg}	hit	Time	W	Time	
Random2000_100	0.03	18.16	18.16	100	0.17	18.16*	0.91	0.00%
Random2000_200	0.02	15.94	15.94	100	0.14	15.94*	2.66	0.00%
Random2000_300	0.02	13.09	13.09	100	0.13	13.09*	4.61	0.00%
Random2000_400	0.01	14.10	14.10	100	0.62	14.10*	5.45	0.00%
Random2000_500	0.01	12.63	12.63	100	0.98	12.63*	7.23	0.00%
Wrandom2000_100	0.04	43.52	43.52	100	7.02	43.52*	0.70	0.00%
Wrandom2000_200	0.02	41.87	41.87	100	8.58	41.87*	3.30	0.00%
Wrandom2000_300	0.02	42.40	42.40	100	0.27	42.40*	4.09	0.00%
Wrandom2000_400	0.01	46.03	46.03	100	0.05	46.03*	5.47	0.00%
Wrandom2000_500	0.01	37.69	37.69	100	0.83	37.69*	9.09	0.00%
Uniform2000_100_10	0.33	6.85	6.85	100	19.17	6.53	3600.00	4.86%
Uniform2000_200_10	0.60	12.24	12.24	100	14.69	11.82	3600.00	3.55%
Uniform2000_300_10	0.70	17.20	17.20	100	41.04	15.61	3600.00	10.18%
Uniform2000_400_10	0.77	22.02	22.02	100	79.70	19.18	3600.00	14.79%
Uniform2000_500_10	0.81	26.56	26.56	100	134.88	24.94	3600.00	6.48%
Decay2000_100	0.78	86.37	86.25	15	217.96	86.37*	0.28	0.00%
Decay2000_200	0.89	159.18	158.40	3	220.01	159.67*	0.39	-0.31%
Decay2000_300	0.93	220.66	216.52	1	226.23	226.82*	0.70	-2.46%
Decay2000_400	0.95	266.76	261.67	1	256.66	277.01*	0.72	-3.70%
Decay2000_500	0.96	316.11	311.43	1	189.36	340.81*	1.23	-7.25%
Average		70.97	70.23		70.92	72.70	902.34	1.30%

seconds, MN/TS outperforms CPLEX by producing much better results in solution quality (revenue). Inversely, on the decay distribution whose graphs have a very high density (≥ 0.89), MN/TS shows a worse performance than CPLEX both in terms of solution quality and computing time, since for 4 of the 5 decay distribution instances, MN/TS is unable to obtain an optimal solution with deviations to the optimal solutions from 0.31% to 7.25%.

5.5. Experimental results on the CATS instances

In this section, we test the MN/TS algorithm on 10 instances that covers all of the benchmark distributions available in the CATS test suite: arbitrary, matching, paths, regions and scheduling. For each distribution, two instances are generated by fixing the number of bids equal to 2000 and setting the number of items equal to 40 and 100 respectively. The CATS distributions seems to be easy to CPLEX since it can solve instances from this test suite with thousands of bids and hundreds of items (Sandholm et al., 2005).

In Table 9, we summarize the computational results obtained by MN/TS as well as those of CPLEX. It can be seen that CPLEX performs extremely well on this test suite. For each of these 10 instances, CPLEX is able to reach the optimal solution within 60 seconds. For 8 of these 10 instances, MN/TS is able to achieve the optimal solutions but it is much slower than CPLEX. For the two remaining instances whose graphs have a very high density (≥ 0.86), MN/TS

Table 9: MN/TS versus CPLEX on the CATS instances

Instance	Density	MN/TS				CPLEX		δ
		W	W_{avg}	hit	Time	W	Time	
Arbitrary2000_40	0.17	4046.01	4046.01	100	2.16	4046.01*	0.50	0.00%
Arbitrary2000_100	0.85	8977.21	8841.35	2	215.36	8977.21*	51.98	0.00%
Matching2000_40	0.85	129.80	129.80	100	0.15	129.80*	0.03	0.00%
Matching2000_100	0.96	508.32	507.38	4	189.26	515.82*	0.03	-1.45%
Paths2000_40	0.83	24.87	24.69	5	198.38	24.87*	0.03	0.00%
Paths2000_100	0.86	36.41	36.16	1	225.39	36.77*	0.09	-0.96%
Regions2000_40	0.23	4558.89	4558.89	100	4.63	4558.89*	0.22	0.00%
Regions2000_100	0.57	9401.70	9400.49	91	110.56	9401.70*	0.70	0.00%
Scheduling2000_40	0.72	84.62	84.62	100	0.13	84.62*	0.03	0.00%
Scheduling2000_100	0.84	204.51	204.48	93	89.52	204.51*	0.05	0.00%
Avg		2797.23	2783.38		103.55	2798.02	5.36	-0.24%

fails to reach an optimal solution with a deviation to the optimal values smaller than 1.45%. We conclude that CPLEX is a better approach than MN/TS on the CATS instances.

6. Discussions

As shown in the last section, MN/TS shows excellent performances on most of the tested WDP benchmark instances. This is particularly the case for the REL and Sandholm test suites. However, for some cases from the Sandholm and CATS test suites, our approach does not perform well and fails to attain the optimal solutions which can be easily reached by the CPLEX 12.4 solver. Even if there is no formal arguments to fully explain these results, it is intuitively understood that the performance of a given algorithm depends on the properties of the underlying graphs. In our case, it seems that the density of the underlying MWCP graph impacts on the behavior of the MN/TS algorithm, graphs with high densities being more difficult for MN/TS. For instance, the REL instances where the densities of the transformed graphs are generally low (from 0.07 to 0.35) are quite easy for MN/TS. This is also true for the random distribution and weighted random distribution instances from the Sandholm test suite whose densities range from 0.01 to 0.04. Inversely, the performance of MN/TS clearly decreases on graphs with a very high density. This is the case for the five decay distribution instances (densities $\geq$ 0.78) and two CATS instances (densities $\geq$ 0.86).

To understand why the clique approach does not work well for dense graphs, we highlight the fact that the number of cliques in the search space grows exponentially with the graph density. As such, a dense graph contains a high number of cliques, making it more difficult for an algorithm to locate the optimal solution. As an illustration, Figure 4 shows the evolution of the number of maximal cliques contained in random graphs of size 40 with densities ranging from 0.1 to 0.8. The number of maximal cliques in a graph is obtained with the

algorithm of Loukakis & Tsouros (1981) by enumerating all maximal cliques in the graph. Note that a maximal clique is a clique not contained in any other clique and corresponds to a leaf node in the search tree. From Figure 4, we observe that as the graph density increases, the number of cliques contained in the graph grows drastically, rendering the task of reaching the maximum clique extremely difficult. This observation is also confirmed on the unweighted maximum clique problem. Indeed, even the most effective heuristic algorithms such as DLS (Pullan & Hoos, 2006) and PLS (Pullan, 2006) have trouble to attain the optimal solutions for some large and dense graphs with a density larger than 0.8.

7. Conclusion and perspectives

The winner determination problem (WDP) in combinatorial auctions is an important and difficult combinatorial optimization problem with a number of highly relevant practical applications. In this paper, we have investigated for the first time the approach of solving the WDP by recasting the problem into the maximum weight clique problem and solving it with an existing clique algorithm (MS/TS).

We have evaluated the merit and limit of this clique-based approach for the WDP via a large experimental assessment with three well-known test suites (REL, Sandholm, CATS) of the literature. These test suites represent a total

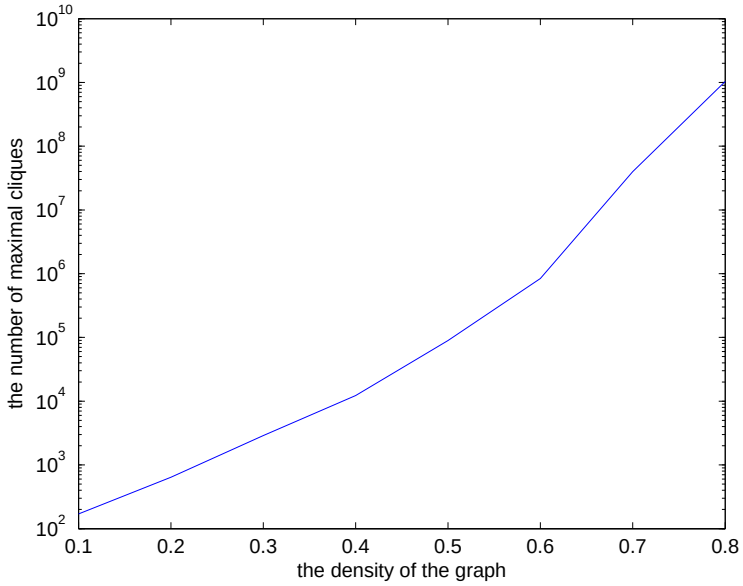

Figure 4: The number of maximal cliques grows with the graph density.

of 530 test cases and cover divers problem characteristics. We have compared the results of the clique-based approach with those reported by several best performing specific WDP heuristics (MA, SAGII, Casanova) as well as the powerful (and general) CPLEX 12.4 solver. According to the computational results, we can draw the following conclusions.

Despite its simplicity, the clique-based approach with the MS/TS algorithm shows excellent performances on a large portion of the tested benchmark instances. For the REL instances which are difficult for the CPLEX 12.4 solver and other existing heuristics, the clique approach performs extremely well and improves the previous best known results for a large number of cases, making MN/TS the best choice for this set of instances. For the Sandholm test suite, the clique approach is preferable to CPLEX when the graph density is not too high (< 0.80) or when the graph has a random, weighted random and uniform distributions. For the decay distribution, CPLEX is a better choice. For the CATS test suite which is known to be easy to CPLEX, CPLEX dominates the clique approach even if MS/TS manages to find good solutions in some cases.

This study confirms that no single method can dominate all the other approaches on all test problems. In this sense, the clique-based approach presented in the paper offers an effective and complementary alternative for practical solving of the WDP problem.

Based on this work, we advance some research perspectives with the purpose of designing more effective WDP methods.

First, this study focuses on the MS/TS heuristic to solve the problem. Given that different MWCP heuristics employ different search strategies, it would be interesting to investigate other MWCP heuristics to see whether one can enlarge the classes of the WDP that can be effectively solved. Going one step further, it is even possible to consider a clique approach using multiple MWCP heuristics to cope with different WDP instances.

Second, as discussed in Section 2, the MS/TS algorithm used in this study as well as most effective clique algorithms rely on several move operators (e.g., add, swap, drop) while the current WDP heuristics are most often based only on a basic 'add-and-repair' operator. Given that MS/TS achieves good performances on the tested WDP benchmark instances, it can be expected that dedicated WDP heuristics can improve their performances if they use more sophisticated move operators to explore the search space.

Third, as shown in Section 3, the WDP can be formulated as an integer programming problem. As such, its LP relaxation can be solved easily by any LP solver. The resulting LP optimal solution could be a source of useful information for a dedicated WDP heuristic. For instance, one can fix some variables (bids) according to the LP optimum and exclude them from the search space examined by the WDP heuristic. Such a strategy helps the heuristic to intensify its search and proves to be useful for a number of binary LP problems. It is worthy of investigating this strategy within the context of the WDP.

Finally, the clique-based approach shows excellent performances for the WDP problem when compared with the current best performing WDP heuristics. Another interesting study would be to extend this clique approach to deal

with more generalized combinatorial auction problems, such as the multi-unit combinatorial auctions (Leyton-Brown et al., 2000a) and iterative combinatorial auctions (Parkes & Ungar, 2000).

To conclude, the WDP is a generic model which finds more and more applications including those mentioned in this paper. Advances in solution methods for the WDP will help to find satisfying solutions to many practical problems. Meanwhile, studies of challenging real-world applications will encourage the development of more effective methods for the general WDP. Given the growing interest in the problem and their applications, it can be expected that research in these domains will become even more intense and fruitful in the forthcoming years.

Acknowledgment

We are grateful to the reviewers for their helpful comments and suggestions which helped us improve the paper. This work was partially supported by the Region of "Pays de la Loire" (France) within the Radapop (2009-2012) and LigeRO Projects (2009-2013).

References

A. Andersson, M. Tenhunen, F. Ygge, Integer programming for combinatorial auction winner determination. In Proceedings of the 4th International Conference on Multi-agent Systems. IEEE Computer Society Press, New York, pp. 39–46, 2000.

G. Ausiello, A. D'Atri, M. Protasi, Structure preserving reductions among convex optimization problems. Journal of Computer and System Science 21: 136-153, 1980.

D. Boughaci, B. Benhamou, H. Drias, A memetic algorithm for the optimal winner determination problem. Soft Computing, 13(8–9), 905–917, 2009.

D. Boughaci, B. Benhamou, H. Drias, Local Search Methods for the optimal winner determination problem in combinatorial auctions. Journal of Mathematical Modelling and Algorithms, 9(2): 165–180, 2010.

P. Cramton, Y. Shoham, R. Steinberg, Combinatorial Auctions. MIT Press, 2006.

L.F. Escudero, M. Landete, A. Marín, A branch-and-cut algorithm for the winner determination problem. Decision Support Systems, 46(3): 649–659, 2009.

Y. Guo, A. Lim, B. Rodrigues, Y. Zhu, Heuristics for a bidding problem. Computers & Operations Research, 33(8): 2179–2188, 2006.

O. Günlük, L. Lászlo, S. de Vries, A branch-and-price algorithm and new test problems for spectrum auctions. Management Science, 51(3): 391–406, 2005.

W. Fontanini, P.A.V. Ferreira, A game-theoretic approach for the web services scheduling problem. Expert Systems with Applications, 41(10):4743–4751, 2014.

Y. Fujishima, K. Leyton-Brown, Y. Shoham, Taming the computational complexity of combinatorial auctions: optimal and approximate approaches. In Proceedings of the 6th International Joint Conference on Artificial Intelligence, pp, 548–553, 1999.

H.H. Hoos, C. Boutilier, Solving combinatorial auctions using stochastic local search. In Proceedings of the 17th National Conference on Artificial Intelligence, pp. 22–29, 2000.

J. Ignatius, Y.J. Lai, S.M.H. Motlagh, M.M. Sepehri, A. Mustafa, Combinatorial auction under fuzzy environment,Expert Systems with Applications, 38(9): 11482–11488, 2011.

H.C Lau, Y.G. Goh, 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. 154–161, 2002.

D. Lehmann, M. Rudolf, T. Sandholm, The winner determination problem. In Cramton et al. (Ed) Combinatorial Auctions. MIT Press, Cambridge, 2006.

K. Leyton-Brown, Y. Shoham, M. Tennenholz, An algorithm for multi-unit combinatorial auctions. In Proceedings of the 7th International Conference on Artificial Intelligence, pp. 56–61, 2000.

K. Leyton-Brown, M. Pearson, Y. Shoham, Towards a universal test suite for combinatorial auction algorithms. In Proceedings of the 2nd ACM Conference on Electronic Commerce, ACM Press, pp 66–76, October, 2000.

K. Leyton-Brown, Resource allocation in competitive multiagent systems. Doctoral Dissertation, Standford University, 2003.

E. Loukakis, C. Tsouros, A depth first search algorithm to generate the family of maximal independent sets of a graph lexicographically, Computing, 27(4):349–366, 1981.

M. Mito, S. Fujita, On heuristics for solving winner determination problem in combinatorial auctions. Journal of Heuristics, 10(5): 507–523, 2004.

N. Nisan, Bidding and allocation in combinatorial auctions. In Proceedings of the 2nd ACM Conference on Electronic Commerce. ACM Press, pp 1-12, October, 2000.

D.C. Parkes, L.H. Ungar, Iterative Combinatorial Auctions: Theory and Practice, In, Proc. 17th National Conference on Artificial Intelligence (AAAI-00), pp. 74–81, 2000.

M.W. Padberg, On the facial structure of set packing polyhedra. Mathematical Programming, 5(1): 199–215, 1973.

A. Pla, B. López, J. Murillo, N. Maudet, Multi-attribute auctions with different types of attributes: Enacting properties in multi-attribute auctions, Expert Systems with Applications, 41(10): 4829–4843, 2014.

W. Pullan, Phased local search for the maximum clique problem. Journal of Combinatorial Optimization, 12(3): 303–323, 2006.

W. Pullan, H.H. Hoos, Dynamic local search for the maximum clique problem. Journal of Artificial Intelligence Research, 25: 159–185, 2006.

A.K. Ray, M. Jenamani, P.K.J. Mohapatra, Supplier behavior modeling and winner determination using parallel MDP. Expert Systems with Applications, 38(5): 4689–4697, 2011.

M.H. Rothkopf, A. Pekeč, R.M. Harstad, Computationally manageable combinatorial auctions. Management Science, 44(8): 1131–1147, 1998.

T. Sandholm, Algorithm for optimal winner determination in combinatorial auctions. Artificial Intelligence, 135(1–2): 1–54, 2002.

T. Sandholm, S. Suri, BOB: Improved winner determination in combinatorial auctions and generalizations. Artificial Intelligence, 145(1–2): 33–58, 2003.

T. Sandholm, S. Suri, A. Gilpin, D. Levine, CABOB: A fast optimal algorithm for winner determination in combinatorial auctions. Management Science, 51(3): 374–390, 2005.

S. Satunin, E. Babkin, A multi-agent approach to Intelligent Transportation Systems modeling with combinatorial auctions. Expert Systems with Applications, 41(15): 6622–6633, 2014.

I. Sghir, J.K. Hao, I. Ben Jaafar, K. Ghédira, A recombination-based tabu search algorithm for the winner determination problem. In P. Legrand et al. (Eds): AE 2013, Lecture Notes in Computer Science 8752: 157–169, 2014, Springer Verlag.

C. Tsung, H. Ho, S. Lee, An Equilibrium-based Approach for Determining Winners in Combinatorial Auctions. In: Proceedings of the 9th IEEE International Symposium on Parallel and Distributed Processing with Applications, pp.47–51, 2011.

D.J. Wu, Software agents for knowledge management: coordination in multi-agent supply chains and auctions. Expert Systems with Applications, 20(1): 51–64, 2001.

Q. Wu, J.K. Hao, F. Glover, Multi-neighborhood tabu search for the maximum weight clique problem. Annals of Operations Research, 196(1): 611–634, 2012.

Q. Wu, J.K. Hao, A clique-based exact method for optimal winner determination in combinatorial auctions. Technical Report, 2014.

S. de Vries, R.V. Vohra, Combinatorial auctions: a survey. INFORMS Journal on Computing, 15 (3) : 284–309, 2003.