International Journal on Artificial Intelligence Tools

Using A Lagrangian Heuristic For A Combinatorial Auction Problem

YUNSONG GUO
School of Computing, National University of Singapore, 3 Science Drive 2, Singapore 117543
guoyunso@gmail.com

ANDREW LIM
Department of IELM, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong
iealim@ust.hk

BRIAN RODRIGUES
School of Business, Singapore Management University, Singapore 259756
br@smu.edu.sg

JIQING TANG
Department of IELM, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong
tjq@ust.hk

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

Combinatorial auctions allow bidders to bid for items leading to more efficient allocations, but determining winners in auctions is $\mathcal{NP}$-complete. In this work, a simple yet effective Lagrangian relaxation based heuristic algorithm is presented. Extensive computational experiments using standard benchmark data (CATS) as well as newly generated more realistic test sets were conducted which showed the heuristic was able to provide optimal solutions for most test cases and is within 1% from the optimums for the rest within very short times. Experiements comparing CPLEX 8.0, the fastest current algorithm, showed the heuristic was able to provide equally godd or better solutions often requring less than 1% of the time required by CPLEX 8.0.

Keywords: combinatorial auction; Langrange relaxation; heuristic.

1. Introduction

Combinatorial auctions where bidders are allowed to bid for a collection of items can bring about better allocations but where winner determination is $\mathcal{NP}$-complete. In an auctioning context, an auctioneer requires m items to be sold and an auction is conducted among n bidders who make n bids, $\{b_1, b_2, ..., b_n\}$, where each bid b_i

covers m_i ($1 \leqslant m_i \leqslant m$, $1 \leq i \leqslant n$) distinct items, given by I_i. If bid b_i is selected, a profit p_i ($1 \leqslant i \leqslant n$) results for the auctioneer and hence a bid is a pair $b_i = (I_i, p_i)$. The combinatorial auction winner determination problem (CAWDP) is to label bids as winning or losing to maximize the total auctioneers profit with each item allocated to at most one selected bid:

$$\text{maximize} \sum_{i \in N} p_i x_i$$

$$s.t. \quad \sum_{i \in N} a_{ij} x_i \leq 1, \ j \in M \tag{1}$$

$$x_i \in \{0, 1\}, \ i \in N \tag{2}$$

where $N = \{1, \ldots, n\}$, $M = \{1, \ldots, m\}$, and $a_{ij} = 1$ if bid i covers item j, i.e., $j \in J_i$ and 0 otherwise. Here (1) ensures each row in the 0-1 matrix $[a_{ij}]_{n \times m}$ is covered by at most one column and (2) ensures that $x_i = 1$ iff bid i of is in the solution.

This problem is a well-known $\mathcal{NP}$-complete set packing problem (Ref. 8), and in fact is not even approximable to a ratio of $n^{1-\epsilon}$ in polynomial time, for a fixed $\epsilon > 0$ unless $\mathcal{ZPP} = \mathcal{NP}$ (Ref. 12). The problem has been studied by a number of researchers recently (Ref. 11, 13, 12, 1, 6, 5) and an excellent survey of combinatorial auctions which includes this problem can be found in (Ref. 3).

In recent studies, solution approaches to the CAWDP have included both exact and non-exact heuristic methods. Exact algorithms include a branch-and-bound search given by Fujishima et al. (Ref. 5), an iterative deepening $\mathcal{A}^*$ search by Sandholm (Ref. 12) and the direct application of CPLEX solver by Anderson et al (Ref. 1). It is common to use test sets from CATS (Combinatorial Auctions Test Suite) (Ref. 10). Experiments on the CATS data had shown earlier that the CPLEX 6.5 solver is efficient (Ref. 1). Recently, Sandholm et al. (Ref. 11, 13) proposed a new branch-and-bound tree-search algorithm called CABOB (Combinatorial Auction Branch on Bids), which when applied to the CATS test set with different distributions, achieved faster results than CPLEX 7.0, generally. We believe CABOB and the CPLEX solver are the current leading exact methods for the CAWDP. To cater for test sets with larger scales or more general distributions, non-exact methods have been preferred; these include using an iterative greedy heuristic (Ref. 9) and stochastic local search (Ref. 6). In this work, a heuristic based on Lagrangian relaxation with subgradient optimization is developed. Although a variation of this technique has been applied to a closely related set covering problem (Ref. 2), to the best of our knowledge, no previous work has used Lagrangian relaxation for the CAWDP.

CATS is used to construct test set of various distributions of items covered by each bid. However, recent research has pointed to CATS being undemanding for exact methods such as CPLEX because of the existence of dominating bids, i.e., bids that require few items but provide high profit (Ref. 1, 11), which has been confirmed

in our experimental observations. Such bids are not realistic since bid profit is commonly tied in with the costs of items. In addition to using CATS, therefore, new test sets were designed with the feature of having bid profits proportional prices of the individual items it covers. For these, experimental results show that CPLEX 8.0 fails to provide optimal solutions even for limited problem sizes.

2. A Lagrangian relaxation based heuristic

Lagrangian relaxation is used to fix good upper bounds for solutions after which a heuristic can be applied to find solutions. The first component of the algorithm acquires Lagrange multipliers and uses a deterministic algorithm to find an upper bound for each set of multipliers. These upper bounds are relaxed (and the likely infeasible) solutions and are adjusted for feasible solutions in the second component of the algorithm. Finally, to improve solutions, a refinement component searches neighborhoods of feasible solutions.

Let a Lagrangian multiplier be given by $\mathbf{u} = (u_1, u_2, ..., u_m)$ where $u_j \geq 0$ for all j and define the problem CAWDP($\mathbf{u}$) by:

$$maximize \sum_{i \in N} p_i x_i + \sum_{j \in M} [u_j (1 - \sum_{i \in N} a_{ij} x_i)] \tag{3}$$

$$s.t. \;\; x_i \in \{0, 1\}, \; i \in N \tag{4}$$

We now show that CAWDP($\mathbf{u}$) is a relaxation of CAWDP.

PROPOSITION 1: CAWDP($\mathbf{u}$) *is a relaxation* CAWDP *for any multiplier* $\mathbf{u}$.

Proof. It is clear that feasible solutions of CAWDP are feasible solutions of CAWDP($\mathbf{u}$). Next, the value of the maximum in CAWDP($\mathbf{u}$) is always greater than or equal to the maximum value in CAWDP since, for a feasible solution x of CAWDP, $\sum_{i \in N} p_i x_i \leq \sum_{i \in N} p_i x_i + \sum_{j \in M} [u_j (1 - \sum_{i \in N} a_{ij} x_i)]$ since $u_j \geq 0$ for all $j \in M$ and $1 - \sum_{i \in N} a_{ij} x_i \geq 0$ for all $j \in M$. □

By Proposition 1, any solution of CAWDP($\mathbf{u}$) is an upper bound for CAWDP for any $\mathbf{u}$. But Proposition 1 alone is insufficient to derive a good upper bound for the CAWDP since the quality of the bound will depend on $\mathbf{u}$ itself. We know, however, that given any $\mathbf{u}$, the optimal solution of CAWDP($\mathbf{u}$) can be found in linear time by virtue of the following proposition.

PROPOSITION 2 : *For a given* $\mathbf{u}$, *an optimal solution of* CAWDP($\mathbf{u}$) *can be found in* $O(mn)$ *time.*

Proof. By the definition of CAWDP($\mathbf{u}$),

$$max\{ \sum_{i \in N} p_i x_i + \sum_{j \in M} [u_j (1 - \sum_{i \in N} a_{ij} x_i)] \}$$

$$= max\{\sum_{i\in N} p_i x_i + \sum_{j\in M} u_j - \sum_{j\in M}(u_j \sum_{i\in N} a_{ij}x_i)\}$$

$$= max\{\sum_{i\in N} p_i x_i + \sum_{j\in M} u_j - \sum_{i\in N}\sum_{j\in M} u_j a_{ij} x_i\}$$

$$= max\{\sum_{j\in M} u_j + \sum_{i\in N} x_i(p_i - \sum_{j\in M} a_{ij}u_j)\}$$

From this it follows that, for a fixed $\mathbf{u}$, since $\sum_{j\in M} u_j$ is a constant, and $p_i - \sum_{j\in M} a_{ij}u_j$ can be determined for any $i \in N$, a solution can be constructed by examining $p_i - \sum_{j\in M} a_{ij}u_j$ letting x_i be 1 if $p_i - \sum_{j\in M} a_{ij}u_j \geq 0$ and 0 otherwise. This procedure requires $O(mn)$ time. □

From Proposition 2, it is clear that with a good Lagrangian multiplier, a tight upper bound for CAWDP can easily be found. This leads us to using subgradient optimization.

2.1. *Using subgradient optimization for the Lagrange multiplier*

In the algorithm, an initial vector of multipliers is used and then iteratively updated using subgradient optimization. In implementation, this vector is set to be the profit of a bid that covers the item divided by the number of items covered by that bid and then averaged. In order to update $\mathbf{u}^k$, the vector at the $k^{th\cdot}$ iteration, to get $\mathbf{u}^{k+1}$, we introduce a subgradient vector $\mathbf{g}$ and use $\mathbf{u}^{k+1} = \mathbf{u}^k + \mathbf{g} \times F$ with F being a subgradient constant. To decide the subgradient vector $\mathbf{g}$ (Ref. 4), let $g_j(\mathbf{u}) = 1 - \sum_{\{i\ :\ a_{ij}=1\}} x_i(\mathbf{u})$ for each $j \in M$ where $x_i(\mathbf{u})$ are components of a solution $\mathbf{x}(\mathbf{u})$ of CAWDP($\mathbf{u}$). In the k^{th} iteration, the Lagrangian multiplier $\mathbf{u}^k$ is derived, and $\mathbf{x}^k$, the optimal solution for CAWDP($\mathbf{u}^k$) obtained. As this solution can be infeasible, it is adjusted to ensure feasibility using a random heuristic in which bids selected in $\mathbf{x}^k$ are first sorted in decreasing order of a reduced profit $c_i := p_i - \sum_{\{j\ :\ a_{ij}=1\}} u_j^k$, for each $i \in N$, and then considered one by one where each bid is given a probability 0.9 of selection and allowed to be selected only if it does not contain any item in common with previously selected bids. This heuristic is run 200 times for each $\mathbf{x}^k$ to generate 200 solutions.

In addition, a deterministic method is used to obtain one additional feasible solution. Let Π be the set of items covered by more than one bid in $\mathbf{x}^k$, i.e., $j \in \Pi$ if there exists at least two bids $b_i \in \mathbf{x}^k$, $b_p \in \mathbf{x}^k$, with $a_{ij} = 1$ and $a_{pj} = 1$. Include all bids selected in $\mathbf{x}^k$, and then exclude one bid at a time with the smallest $\frac{c_i}{|\Pi \cap I_i|}$ value until Π becomes empty. The solution is then feasible and a single feasible solution is obtained from each $\mathbf{x}^k$. From these solutions, the best 50 are kept for refinement in the next component of the heuristic.

The outline of the heuristic is given in Algorithm 1.

Algorithm 1 (Subgradient Optimization To Improve Lagrange Multipliers)

(1) Set $u_j^0 \longleftarrow \frac{\sum_{\{i\ :\ a_{ij}=1\}} \frac{c_i}{|\{k\ :\ a_{ik}=1\}|}}{|\{i\ :\ a_{ij}=1\}|}$, $j \in M$, $F \longleftarrow 1$, $iter \longleftarrow 0$
(2) While $F > 0.01$, $iter \longleftarrow iter + 1$; calculate $\mathbf{x}(\mathbf{u})$ and run random heuristic 200 times to get a feasible solution; run the deterministic procedure for an addtional solution and update best 50 solutions. Set $g_j \longleftarrow 1 - \sum_{\{i\ :\ a_{ij}=1\}} x_i(\mathbf{u})$ and $\mathbf{u} \longleftarrow \mathbf{u} + \mathbf{g} \times F$
(3) If after a fixed number of iterations, the best 50 solution do not improve, set $F = F/2$

2.2. *Refinement of solutions*

Once the 50 best solutions are found as described, a greedy local search is applied to explore the neighborhood of the solutions to improve solution quality where a neighborhood exchange operator used here is found in Lau and Goh (Ref. 9). Experiments showed that this refinement increased solution quality by up to 3%. The refinement algorithm is described in Algorithm 2.

Algorithm 2 (A Greedy Refinement Algorithm)

(1) Set $B \longleftarrow$ bids within the 50 solutions obtained
(2) While $|B| \leq$ min (1500, n), select a bid b_i not in B with maximum profit and set $B \longleftarrow B \cup b_i$
(3) For each of the 50 solutions β_i, $1 \leq i \leq 50$, while for $b_j \in B$ with profit p_j larger than the sum of weights of bids in β_i which conflict with b_j
(4) Return the best among the 50 solutions

The set of bids B is a candidate set we choose from to add to the 50 solutions. The constraint of maximum 1500 bids in B is used to limit the times required. The greedy method tries to add new bids from the candidate set which after removing all conflicting bids will result in an increase in total profit. An outer "while" loop is terminable since the profit of each β_i increases monotonically.

3. Computational Experiments

Experiments to compare the heuristic (denoted by LH) with CPLEX 8.0 solver were conducted. According to Sandholm et al. (Ref. 11), the current leading exact algorithm, the branch-and-bound method CABOB outperforms CPLEX 7.0 moderately for most test sets from CATS. However, for 7 out of 9 distributions used, both CPLEX 7.0 and CABOB generated optimal results within 10 seconds for the worst test case of each distribution. For other the other distributions, CABOB ran in 20 seconds for the worst test set while CPLEX 7.0 only required less than 10 seconds. In only for 1 out of the 9 distributions, namely a components distribution, did CABOB produce the optimal solution within 1 second whereas CPLEX 7.0 required more than 800 seconds. As a result, we believe the CPLEX remains a competitive tool for the CAWDP using CATS test sets. Comparison were also

based on CPLEX 8.0, which is in general 40% faster than CPLEX 7.0 for integer programming problems according to ILOG specifications (Ref. 7).

All experiments were conducted on a 2.8GHz Pentium(R) IV machine with 1GB Memory and LH was implemented in ANSI C++ and compiled with GNU GCC 3.2 compiler. In order to conduct fair comparisons, parameters of CPLEX were tuned for CAWDP before the experiments. Because of the various characteristics of the benchmarks, there is no general parameter settings which work best for all instances. After careful considerations, parameters for CPLEX 8.0 were set as follows: 1.) set *mip* strategy to emphasize feasibility; 2.) set *mip* clique cut generation strategy to 2 (aggressive); 3.) explore the "up" branch first in the enumeration tree. Default values were used for all the other parameters. We found that the above settings worked especially well for the difficult benchmarks.

3.1. *Comparison using CATS test sets*

Experimental results using CATS and an additional test set which is harder than CATS for the CAWDP are provided here. A total of 8 different distributions from CATS, namely the exponential, random, uniform, binomial, decay, scheduling, matching and paths distributions, were used. Test cases of the first 5 distributions were exactly the same as used in Anderson et al. 2000, and the last 3 distributions were generated directly from CATS. Here, the usual naming convention in describing the results is followed. For example, $exp-p-q$, is the test set with exponential distribution with p items and q bids.

The results are presented in Table 1. Here, $\mu_{Density}$ is the average density of the 10 test cases in each group, where the density of a test case is defined by $\frac{\sum_{i \in N} \sum_{j \in M} a_{ij}}{n \times m}$ which is an indicator of how many items one bid covers. μ_{CPLEX} and μ_{LH} is the average results obtained by CPLEX 8.0 and LH, respectively, for each category and t_{CPLEX} and t_{LH} are the average time spent in seconds, respectively. Here, δ denotes the difference between LH's results and the optimal solutions found from CPLEX.

Table 1. Experimental results for CATS benchmarks

Test set	# instances	$\mu_{Density}$	μ_{CPLEX}	t_{CPLEX}	μ_{LH}	t_{LH}	δ
EXP-30-3000	10	0.09	44723.8	0.27	44723.8	0.81	0
RND-400-2000	10	0.50	16143.8	5.14	16143.8	14.62	0
UNI-100-500	10	0.03	129050.0	31.31	128254	2.3954	0.6%
BIN-150-1500	10	0.20	94792.7	234.00	94792.7	7.91	0
DEC-200-10000	10	0.02	196266.0	34.81	194957.0	54.01	0.7%
SCH-400-2200	10	0.02	54.256	0.27	54.256	25.91	0
MAT-600-2000	10	0.01	997.419	0.26	997.218	12.63	0.02%
PAT-600-2000	10	0.01	61.898	0.46	61.852	19.28	0.07%

From this table it can be seen that, for all CATS test sets, LH provides high quality solutions. For the exponential, random, binomial and scheduling distributions, LH obtains optimal results while for the other 4 distributions the results are all within 1% of the optimal. With respect to time efficiency, for most test sets, except the uniform, random and binomial distributions, CPLEX is very efficient as it provides optimal results within several seconds. On the other hand, LH takes more time than CPLEX for these test sets but still produces high quality results in tens of seconds. However, LH records times from 0.8 to 54 seconds with a smaller variation than CPLEX which runs from 0.2 to 230 seconds.

3.2. *Comparison using PBP test sets*

For further comparisons of the LH algorithm, different test sets from CATS were used. Some CATS generated test cases appeared to be easily solved, since the LP relaxed solutions of CATS problems were very close, if not equal, to the actual solutions. In realistic auction problems, the price for a bid is frequently related to the quantity and quality of the items bid for. While real values of items can change due to the market fluctuations, these are generally proportional to the number of bids which cover it. With this in mind, a new proportional bid price (PBP) CAWDP test set was generated using Algorithm 3 outlined below.

Algorithm 3 (Price Proportional Bids Generation)

(1) Specify n, m, and a probability density for matrix $[a_{ij}]$
(2) Generate $[a_{ij}]$. For all bids i from 1 to n and for all items j from 1 to m, $a_{ij} \longleftarrow 1$ with the specified density
(3) Generate prices for items. For all items 1 to m, price of item $j \longleftarrow$ number of bids which cover j from $[a_{ij}]$ multiplied by p where p is a random number in (0.9, 1.1)
(4) Generate prices for bids. For bids i from 1 to n,price of bid $i \longleftarrow$ sum of prices of items it covers multiplied by p as in (3)

Test cases from PBP of different sizes were generated. Since CPLEX 8.0 could not give the optimal results for many of these test cases within reasonable times, we set the time limit for CPLEX to be 1800 seconds. The results are presented in Table 2. Here, δ_1 is the ratio of LH's result over CPLEX's, and δ_2 is the ratio of LH's time spent over CPLEX's.

Out of the 15 test sets in Table 2, LH performed equally to CPLEX 8.0 for 11 sets and was better in 3 sets by between 2% to 11%. Only one solutions is 1% worse off than CPLEX. LH obtained optimal solutions for which CPLEX required the maximum time. Notably, LH's time efficiency is significantly better than CPLEX, often requiring less than 1% the time required by CPLEX.

Table 2. Experimental results for PBP test sets

Test set	*Density*	*LH*	t_{LH}	*CPLEX* 8.0	t_{CPLEX}	δ_1	δ_2
PBP-100-200	0.05	901.42⋆	0.984	901.42	5.797	1.00	0.170
PBP-100-200	0.10	1428.17⋆	0.844	1428.17	11.407	1.00	0.074
PBP-100-200	0.15	1818.26⋆	0.625	1818.26	4.75	1.00	0.132
PBP-200-200	0.03	946.62⋆	1.547	946.62	23.594	1.00	0.066
PBP-200-200	0.05	1296.03⋆	1.281	1296.03	97.360	1.00	0.013
PBP-200-200	0.10	1916.20⋆	0.921	1916.2	11.172	1.00	0.082
PBP-200-1500	0.10	18118.9	7.953	17486.4	*tle*	1.04	0.0044
PBP-200-1500	0.15	21937.5⋆	6.781	21937.5	972.703	1.00	0.0026
PBP-200-1500	0.20	25583.9⋆	8.453	25583.9	1060.92	1.00	0.013
PBP-500-2000	0.03	17598.5	21.41	17312.2	*tle*	1.02	0.0083
PBP-500-2000	0.10	33883.2	12.75	32284.8	*tle*	1.05	0.0107
PBP-500-2000	0.15	47070	15.91	42329.9	*tle*	1.11	0.0876
PBP-500-5000	0.05	58210.2	27.31	58741.2	*tle*	0.99	0.0152
PBP-500-5000	0.15	121007.0	42.67	121007.0	*tle*	1.00	0.0152
PBP-500-5000	0.20	148706.0	38.17	148706.0	*tle*	1.00	0.0212

Note:
[1] ⋆ indicates optimal solution found
[2] *tle* indicates CPLEX was unable to obtain the optimal solution within 1800 seconds and the best result at that time is reported.

4. Conclusion

In this paper, a winner determination combinatorial auction problem was studied for which a new Lagrangian relaxation based heuristic was developed. The heuristic performed well when compared with the current leading exact best method CPLEX 8.0 on various CATS benchmarks achieving optimal solutions for most test cases and within 1% of optimal solutions for the remainder. When applied to the more realistic PBP test sets, the heuristic provided equally good or better results than CPLEX always requiring mostly less than 1% time required by CPLEX 8.0.

References

1. Andersson, A. Tenhunen, M., and Ygge, F. (2000) Integer programming for combinatorial auction winner determination, In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pp. 39–46
2. Caprara, A., Fischetti, M. and Toth, P. (1999), A heuristic method for the set covering problem, *Operations Research* 47(5) pp. 730 – 743
3. DeVries, S. and Vohra, R. (2003) Combinatorial auctions: A survey, *INFORMS Journal of Computing* 15(1), pp. 284 - 309
4. Fisher, M.L. (1981) The Lagrangian relaxation method for solving integer programming problems, *Management Science* 27(1) pp. 1–18
5. Fujishima, Y., Leyton-Brown, K. and Shoham, Y. (1999), Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches, In *Sixteenth International Joint Conference on Artificial Intelligence*, pp. 548–553
6. Hoos, H.H. and Boutilier, C. (2000), Solving combinatorial auctions using stochastic local search, In Proceedings of the Seventeenth National Conference on Artificial Intelligence, pp. 22–29

7. ILOG Inc. (2002) CPLEX 8.0 User's Manual
8. Karp, Richard M., (1972), Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, Editors, *Complexity of Computer Computations*, Plenum Press, New York
9. Lau, H.C. and Goh, Y.G. (2002), An intelligent brokering system to support multi-agent web-based 4th-party logistics, In Proceedings of the *Fourteenth International Conference on Tools with Artificial Intelligence*, pp. 10–11
10. Leyton-Brown, K., Pearson, M. and Shoham, Y. (2000), Towards a universal test suite for combinatorial auction algorithms, In *ACM Conference on Electronic Commerce*, pp. 66–76, 2000.
11. Sandholm, T., Suri, S., Gilpin, A. and Levine, D. (2001), CABOB: A fast optimal algorithm for combinatorial auctions, In *Proceedings of the International Joint Conference on Artificial Intelligence* (IJCAI), Seattle, WA, pp. 1102–1108
12. Sandholm, T. (2002) Algorithm for optimal winner determination in combinatorial auctions, *Artificial Intelligence* 135(1-2) pp. 1–54
13. Sandholm, T, Suri, S., Gilpin, A., and Levine D., (2004), CABOB: A fast optimal algorithm in combinatorial auctions, to appear in *Management Science*