AN EFFIEICNT WINNER DETERMINATION ALGOLITHM FOR COMBINATORIAL ASCENDING AUCIONS

Chihiro Ono Satoshi Nishiyama Hiroki Horiuchi

KDDI R&D Laboratories, Inc.

{ono, tomo, hiroki}@kddilabs.jp

Abstract

In this paper we study combinatorial auctions where bidders can quote for a combination of the objects being sold. In a previous article we have proposed a combinatorial ascending auction where the bidders can place a bit at an arbitrary timing via the Internet. For combinatorial auctions, computational complexity increases exponentially as the number of possible combination increases. Although some algorithms for reducing the complexity have been proposed, they are only suitable for the one-shot auctions where bidders submit bids only once simultaneously. Thus, we can improve the performance by making use of the previous valuation for doing next valuation. In this paper we propose a method to reduce computational burden for combinatorial ascending auctions and verify the effectiveness of the algorithm through the evaluation.

1. Introduction

Auctions are thought to be an efficient way of allocating items and have been popular in portals and e-marketplaces nowadays, such as ebay[1] and yahoo[2]. In particular, combinatorial auctions, where bidders can quote for a combination of the items have been studied in many places[3]. However, when we try to apply existing mechanisms for combinatorial auction to portal sites for anonymous users, these mechanisms are insufficient in terms of restrictions such as one-shot sequence or activity rules, where users must bid actively from the beginning of the auction to the end.

In order to be effectively used on the internet, in a previous article, we have proposed a Combinatorial Ascending Auction (CAA) where the bidders can place a bid at an arbitrary timing on the Internet[4].

For combinatorial auctions, computational complexity increases exponentially as the number of possible combination increases. Although some algorithms for reducing the complexity have been proposed, they are only suited for one-shot auctions where bidders submit bids only once simultaneously[5][6][7]

Thus, we can improve the performance by making use of previous valuation for doing next valuation. In this paper we propose a method to reduce computational burden for combinatorial ascending auction.

Section 2, describes the overview of the existing combinatorial auctions and

combinatorial ascending auction. Section 3 discusses about the complexity issues. Then in Section 4, proposed algorithm with an actual example is described. Section 5 shows the evaluation and Section 6 concludes the paper.

2. Overview of Combinatorial Auctions and CAA

2.1 Combinatorial Auctions

Let $I = \{i_1, i_2, ..., i_n\}$ be a collection of goods. Let $|I| = n$ be the number of the goods. Let $G = \{g_1, g_2, ...\}$ be a set of combination of elements of *I*. Then $|G| = 2^n - 1$, where the empty set is excluded.

The auction system consists of the seller, the bidders, and the auctioneer who determines an allocation for the seller. First, the seller sends the collection of heterogeneous goods *I* to the auctioneer, and the auction begins. Then the bidders send the price to an element of *G,* specifically $b_j=\{g_j, p_j(g_j)\}$, where $p_j(g_j)$ is the bid price in b_j. The auctioneer valuates the bid and sends the result to the seller. When valuating the bids, the auctioneer chooses the allocation *A* which maximizes the revenue. Let $A=\{a_1, a_2,,,\}$ be an allocation where a_i be an combination of elements. This is a problem of choosing the combination of goods which does not overlap each other. Let $S \in G$. Then this problem is expressed as:

$$\max \sum_{[i,S] \in X} p_i(S)$$

such that $X = \{[i,S] \mid S \cap S' = \Phi, \forall S, S' \in X\}$ and the total revenue *T(A),* is the sum of revenue of the elements in *A,* which is a bid price.

2.2 Characteristics of CAA

CAA is an ascending auction which accepts combinatorial bids. In contrast with the standard one-shot combinatorial auction or the multi-round auction where all the bidders have to be present at the beginning of the auction, CAA has the following features:

1. Bidders can place bids at any timing.
2. Auctioneer valuates bids whenever new bids are accepted, and determines a provisional allocation
3. There are three provisional results for the bid. The first possibility is winning (W), where the bid wins the goods at this moment. The second possibility is losing (L), where the bid does not and will not win the goods. The last possibility is pending (P) where the bid is not winning at this moment but it is possible to win in the future with the help of bids for other combinations.
4. The auctioneer keeps and updates the combination table *(CT)* for storing maximum bids $b_j=\{g_j, p_j(g_j)\}$, and states for each elements of combination *G* , and allocation table (AT) for storing member of the allocation, *A* until the auction ends.

Specifically, the valuation will be done in following three steps.

1. For an incoming bid $b_j=\{g_j, p_j(g_j)\}$, compare $p_j(g_j)$ with $p_t(g_j)$ which is a price for g_j in *CT*. If $p_j(g_j)$ is smaller than $p_t(g_j)$, then we set the state of b_j to be L, and quit the valuation process. Else, we overwrite *CT* by b_j and go to the next step. At this moment, b_j can take one of W or P.
2. We check the *CT*. If there exists $\{g_i\ g_x\}$ such that, $p_j(g_i) + p_t(g_x) > p_t(\{g_i\ g_x\})$

then{ g_i g_x } does not have a possibility of being a winner. Thus we set the state of the bid for {g_i g_x } in CT to be L.

3. In *CT*, we re-compute the revenue from the new bid and compare with the revenue at current *AT*. If the revenue from the new bid is greater than the revenue from the current allocation, we make the new allocation a new provisional winning allocation, and make a bid contained in the new winning allocation a winner. We set the states of other bids to be P.

3. Complexity Issues

3.1 Complexity issues for CA and existing research

Combinatorial auction is computationally expensive. Thus in order to solve the problem in polynomial time, several methods have been proposed. The first category is pruning and the second category is obtaining an approximate solution.

As examples of algorithms in the first category, Sandholm et al[6] proposed an algorithm to search a best allocation based on iterative deepening algorithm. Also, Fujishima et al[5] proposed an algorithm to search for a best allocation based on the algorithm in depth first search. This algorithm is called CASS (combinatorial auction structured search).

As an example in the second category, Sakurai et. al[7] used limited discrepancy search (LDS) which limits the search effort to the range where the optimal solution can lie in high probability. This is good for any-time characteristics and they can achieve high quality for only short running time.

3.2 Complexity in CAA

In contrast with the previous combinatorial auction, the characteristics of CAA are as follows: Firstly, since the bids arrive over time, the bid is processed immediately so that feedback can be given, rather than being processed all at once after the auction is closed. Secondly, since it is an ascending auction, the previous result can be used for valuation. As a result, especially for CAA, there are still rooms for improvement in traditional methods.

4. Algorithm

4.1 Our approach

Our approach is basically to make use of existing *CT* and *AT*, while doing CASS or LDS for search.

We have two improvements: Imp 1) to determine an allocation for items which are not used in the incoming bid then compared with a current allocation, and Imp 2), pruning.

Let $D_k=\{d_{k1}, d_{k2}\ldots\}$ be an element of a_k which is a member of *A*. Let *h(i)* to be the heuristic function which gives us the potential highest value for item *i*. This is defined as follows (see [6]):

$$h(i) = \sum_{i \in S} \{\max b(S) / |S|\}$$

For Imp 1), we set F_h to be *A* minus the elements of the incoming bid, b_h. First, determine the allocation for F_h (=$T(F_h)$) ,and compare $T(F_h)+p(g_h)$ with $T(A)$. If $T(F_h)+p(g_h)$ is larger than $T(A)$, we set the elements of F_h and g_h to be the winner, and replace the current allocation. Otherwise, we set the state of the element of g_h to be P.

For Imp 2), we have two pruning methods. For the first pruning, if we find the subset of *A*,

called A', which has the same members with g_h, then we will determine the state by comparing $T(A')$ and $p(g_h)$. For instance, when $g_h = \{d, e\}$ and $A=\{(a,b,c)(d)(e)\}$ then A' is $\{(d)(e)\}$. In this case, if $p(g_h)$ is larger, b_h will be W, and we set it as a current allocation and delete A' from the current allocation. Otherwise, it becomes a loser, and we set its state to be L.

For the second pruning, after calculating the summation of heuristic function $h()$ for the member of F_h, called $SUMH$, we compare $p(g_h) + SUMH$ with $T(A)$. If it is smaller than $T(A)$, then it becomes loser and the valuation ends.

4.2 Detailed Mechanism

Valuation is carried out as follows:

1) Initialization: As an initialization, we make a dummy bid with value 0 for all items and initial allocation, $T(A) = 0$.
2) Every time when the auctioneer accepts incoming bids, b_h, does the following:
3) Compare $p(g_h)$ with the price of g_h in CT, and if it is larger than the price of g_h in CT, then the auctioneer updates CT and recalculate h for the item of g_h.
4) Check whether b_h meets the condition of pruning (1), and if it meets, then apply pruning (1). If b_h is a loser, valuation ends. If b_h becomes a winner, replace bh with A' and update the state then valuation is finished.
5) Calculate F_h.
6) Check whether b_h meets the condition of pruning 2 and if it meets, then apply pruning 2. If b_h becomes a loser, valuation is finished.
7) Apply the Imp1). If b_h becomes a loser, valuation ends. If b_h becomes a winner, then replace g_h with A' and update the state then valuation is finished.
8) Update all the states in CT according to the valuation.

4.3 Examples

We explain how the valuation works by showing actual example. Table 1 shows the CT at a certain point. Here, $h(i)$ for each good is calculated.

Table1: CT at a certain point

g	p(g)	h(i)	State
a	5	8.3	P
b	7	8.3	P
c	9	9	P
d	6	6	W
e	7	7.5	W
a,b	13	-	P
a,e	15	-	P
d,e	10	-	L
a,b,c	25	-	W

Here, $A=\{(a,b,c), (d), (e)\}$ and $T(A) = 38$. We show three examples.

First, Let $b_1=\{(d,e), 14\}$ be the incoming bid. In this case, at step 3), $A'=\{(d),(e)\}$ is found and $T(A')$=13. As $p(b_1)$ is larger than $T(A')$, it wins and new allocation becomes $\{(a,b,c), (d,e)\}$.

Second, Let $b_2=\{(c,d), 13\}$ be the incoming bid. In this case, at step 5), $SUMH = h(a)+h(b)+h(e)$ = 23.1. As $p(b_2) + SUMH$ (= 36.1) is smaller than $T(A)$, then b_2 becomes a loser and valuation ends.

Third, Let $b_3=\{(c,d), 20\}$ be the incoming bid. As $p(b_3)+SUMH$ (=43.1) is larger than T(A), step 5) is skipped and move onto step 6). Here, $F_3 = \{a, b, e\}$. Next calculate the allocation and

A'={(a,e), (b)} and *T(A')* = 22. As $p(b_3)+T(A')$ (=42) is larger than *T(A)*, b_3 wins and new allocation is *{(a,e), (b), (c,d)}*.

5. Evaluation

In this section, we evaluate our proposed mechanism in terms of the performance and quality by comparing with existing batch methods.

5.1 Settings

Parameters for the evaluation are as follows:

- The number of items: M=32
- The number of biddings: N=30,60,90
- Bid distribution: Two patterns
  - Random: For each bid, pick the number of items randomly from 1 to M and pick the prices from [0, 1]*(number of items in the combination)
  - Uniform: Draw the same number of randomly chosen items for each bid. The number is 3 and prices from [0, 3]
- New incoming bid for our model: Random combination and price. We calculate the average by trying 100 times.
- Target calculation methods:
  - LDS
  - CASS
  - Incremental LDS: LDS with applying proposed improvements.
  - Incremental CASS: CASS with applying proposed improvements.

5.2 Results

The number of calculation steps and the total revenue for each calculation method for two bid distributions are shown in the following figures.

Figure 1 depicts the calculation steps required for each method in the case of Random distribution. For CASS, 360354 for N=60 and 637681 for N=90, which are too large to be presented in the figure. Here, both incremental methods using the proposed technique are effective and the step decreases about 10 times less than original methods.

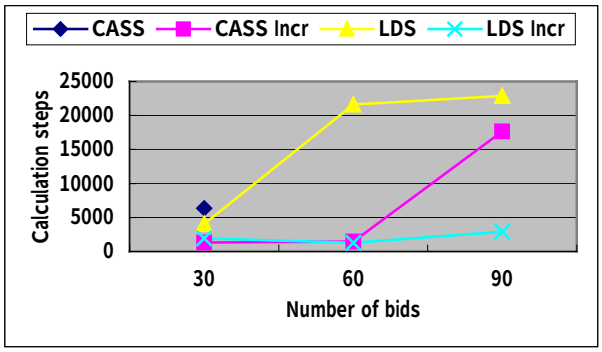

Figure 1 Calculation steps for Random bids

Figure 2 shows the calculation steps required for each method in the case of Uniform distribution. For CASS, 233273 for N=30, 535793 for N=60, and 819724 for N=90, which are large to be presented in the figure. Both incremental methods using the proposed technique are effective and the step decreases about 2 to 3 times for LDS and 10 times for CASS less than original methods

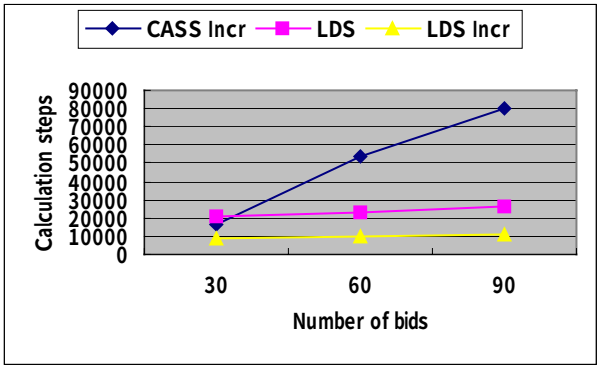

Figure 2 Calculation steps for Uniform bids

Figure 3 and 4 show the comparison between LDS and Incremental LDS in terms of the total

revenue. Incremental LDS can have as much revenue as LDS for both bid distributions.

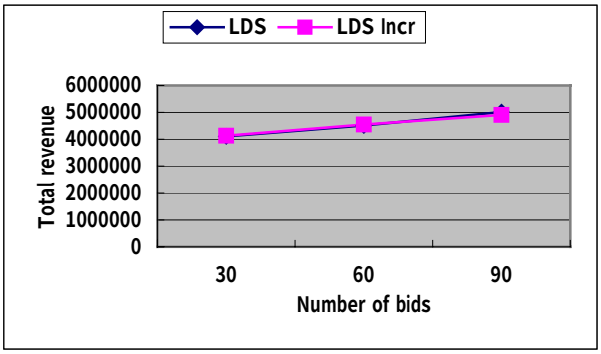

Figure 3 Efficiency for Random bids

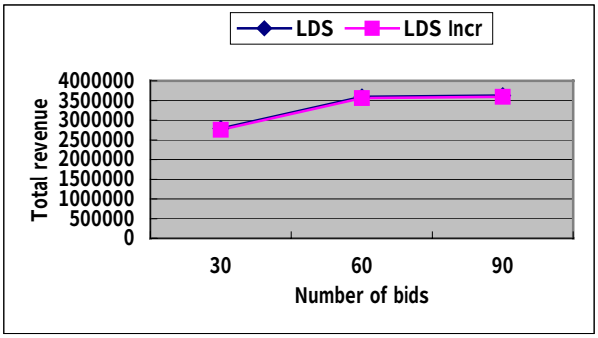

Figure 4 Efficiency for Uniform bids

As a result, we can say that our proposed mechanism is effective when incoming bids arrive sequentially or at most less than 10 bids at the same time.

To be practical, in general, since ascending auctions have a tendency of having last-minute bidding behavior, it can be a good way to have our incremental algorithm from the beginning of the auction to a certain period of time to get quick feedback, and switch the algorithm to the normal method at the very last moment when many bidders give up getting prompt feedback.

6. Conclusion

In this paper, we proposed an efficient winner determination algorithm for combinatorial ascending auction with two improvements against existing methods. Then we demonstrated the effectiveness of our algorithm through evaluation.

We now have several further studies for improving our algorithm. First, as we know the characteristics of the ascending auctions that same bidder may repeat the bids, we may have another technique by using this tendency. Second, we may have approximate reduction method for incremental search though, this time, we do not use approximate method for improvement but for original method. Third, we may make use of the history of bids from the beginning of the auction to the middle to dynamically change the strategy.

Acknowledgement

The authors wish to thank, Mr. Tohru Asami, Director of KDDI R&D Laboratories, Inc. for his continuous guidance for this study.

Reference

[1] http://www.ebay.com

[2] http://www.yahoo.com

[3] Sven de Vries, Rakesh Vohra, “Combinatorial Auctions: A Survey”, 2000

[4] C. Ono, et al, “Proposal of Combinatorial Ascending Auction”, Proc. of DPS , 105-10, p53-58, 2001(in Japanese)

[5] Y. Fujishjima, et al “Taming the Computational Complexity of Combinatorial Auctions: Optimal and Approximate Approaches”, Proc. of IJCAI 1999

[6] Tuomas Sandholm, “An Algorithm for Optimal Winner Determination in Combinatorial Auctions, Proc. of IJCAI, 1999

[7] Y. Sakurai et al, “An Efficient Approximate Algorithm for Winner Determination in Combinatorial Auctions”, Proc of EC-00, 2000