Revenue prediction in budget-constrained sequential auctions with complementarities

(Extended Abstract)

Sicco Verwer *
Radboud University Nijmegen, The Netherlands
S.Verwer@cs.ru.nl

Yingqian Zhang
Erasmus University Rotterdam, The Netherlands
yqzhang@ese.eur.nl

ABSTRACT

When multiple items are auctioned sequentially, the ordering of auctions plays an important role in the total revenue collected by the auctioneer. This is true especially with budget constrained bidders and the presence of complementarities among items. It is difficult to develop efficient algorithms for finding an optimal sequence of items. However, when historical data are available, it is possible to learn a model in order to predict the outcome of a given sequence. In this work, we show how to construct such a model, and provide methods that finds a good sequence for a new set of items given the learned model. We develop an auction simulator and design several experiment settings to test the performance of the proposed methods.

Categories and Subject Descriptors

I.2.6 [**Computing Methodologies**]: Artificial Intelligence—*Learning*; J.4 [**Social and Behavioral Sciences**]: Economics

General Terms

Algorithms, Design, Economics, Experimentation

Keywords

Sequential auctions, revenue maximization, learning

1. INTRODUCTION

Auctions are becoming increasingly popular for allocating resources or items in business-to-business and business-to-customer markets. Often sequential auctions are adopted in practice. Previous research has shown the revenue is heavily dependent on the ordering of items in sequential auctions [1], especially when bidders have budget constraints or when they have preference over bundles of items.

Much of the existing work that studies optimal ordering in auctions focuses on theoretical analysis on bidders' strategy and conditions when the optimal ordering exits. However, it is difficult to apply these results to actual auctions as they rely on strong assumptions which rarely hold in practice. In this paper, we develop a novel method for finding revenue-maximizing orderings that can generalize to real-world auctions. Our method is based on techniques from machine learning. It uses historical auction data in order to quickly learn which orderings have high expected revenue.

Our main contribution is providing a method that first transforms this information into a data set, then learns models for predicting the revenue of orderings, and finally uses a best-first search algorithm in order to find a good ordering for a new set of items. We implement an auction simulator, and design several experiment settings to test the performance of the proposed learning method.

2. LEARNING GOOD ORDERINGS

We assume there is a set of agents A who have budget constraints on purchasing items. Let $R = \{r_1, \ldots, r_l\}$ denote the collection of the item types, and the quantity of each item type can be more than 1. Each bidding agent A_i has a valuation for each type of item or each bundle of different item types $v_i : R \rightarrow \mathbb{R}^+$. In one round of auction, a set of items S with type set $R' \subseteq R$ will be auctioned sequentially with an order that is announced before the auction starts. We assume that the auction is repeated over time, and each auction sells possibly different items, with possibly different set of agents. At the end of each sequential auction, we have the following information at our disposal: (1) the ordering of auctioned items; and (2) the revenue of each sold item. Our goal of the sequential auction design is that given a set of items $r_1, \ldots, r_i$ for sale, deciding the ordering of items such that the revenue collected is maximized.

In order to simplify the learning problem, we make the following two modeling assumptions: (1) *Bidder independence*: in every round of sequential auctions, the set of participating bidders and their valuation functions are similar. (2) *Ordering independence*: the expected revenue for an item depends on which items were sold before and which items are still to be sold, but not on their ordering. The assumptions effectively reduce the difficulty of the learning problem to that of a standard machine learning setting: learn a single model from orderings and their rewards for predicting the expected reward for a given new input ordering.

We view the prediction of the revenue of an auction as a regression problem. Like an MDP, we split this problem into the subproblems of predicting the revenue of the auc-

*This work was performed while the first author worked at KU Leuven, Belgium.
The full version of the paper can be found in [2].

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), 4-8 June 2012, Valencia, Spain.

Algorithm 1 Computing a good ordering

Require: A set of items S, historical data on orderings and their revenues D, a maximum number of iterations m
Ensure: Returned is a good (high expected revenue) ordering
  Transform D into a data set
  for every item type r_T **do**
    Learn a regression model from D for predicting the revenue of item type r_T
  end for
  Initialize a hashtable H and a priority queue Q
  Add the empty data row to Q
  while Q is not empty and the size of H is less than m **do**
    Pop the row of features F with highest value v from Q
    if H does not contain F with a value $\geq v$ **then**
      Add F with value v to H
      Let L be the set of remaining items in F
      for every item type r_T of items in L **do**
        Let i_k be an item of Type r_T in L
        Let L' be a random ordering of $L - i_k$
        Use the learned models to evaluate the value v' of auctioning the ordering $i_k L'$ after F
        Create new features F' for auctioning i_k after F
        Add F' to Q with value $v + v'$
      end for
    end if
  end while
  return The highest evaluated ordering

tioned items. We then sum these up to obtain the overall objective function: $V(r_1 \dots r_n) = \sum_{1 \leq k \leq n} R(r_k, \{r_j \mid j < k\}, \{r_l \mid k < l\})$, where $R(r_k, J, L)$ is a regression function that determines the expected revenue of r_k given that J was auctioned before and L will be auctioned afterwards. We use regression trees as a regression function and train it using features based on the items auctioned before and after the current item r_k. Currently, we provide the following features: (1) For every item type r_T, the amount of r_T items already auctioned; (2) For every item type r_T, the amount of r_T items still to be auctioned; (3) For every pair of item types r_T and $r_{T'}$, the difference between the amount of r_T and $r_{T'}$ items already auctioned. (4) For every item type r_T, the amount of revenue obtained from auctioning r_T items. These features model the influence of utility functions with complementarities and of budget constraints.

We transform an ordering and its obtained revenues into a data set using these 4 types of features. The data set obtained in this way can be given as input to any standard regression method from machine learning. In our case, we learn a regression tree for every item type using recursive partitioning techniques. The result is a set of predictive models for the expected revenue of items, and by summing these revenues we obtain the expected revenue of an auction.

To test an ordering, we employ a best-first search strategy that can be terminated at anytime in order to return the best solution found so far. We show how to compute a good ordering in Algorithm 1.

3. EXPERIMENTS

We developed an auction simulator and created three types of agents who have different bidding strategies: (1) *myopic* agents bid as soon as the asking price reaches their true value; (2) *smart* agents know all the valuation functions of all agents; and (3) *simulator* agents have access to the auction simulator. They bid what the smart agents bid, only if a run of the simulator results in a higher utility.

We first generate a set of agents, and run simulations of 250 random orderings of randomly selected items. These 250 orderings and their obtained revenues are transformed to a data set and provided to the regression tree learner. The resulting models are used to provide an ordering for a new set of randomly generated items using Algorithm 1 with a maximum number of 1000 iterations. We then run 10 simulations of this ordering and average the resulting revenues. We use this average revenue of different sets of items for the learned ordering method as a performance indicator.

We compare our method with two ordering strategies: (i) a random ordering, and (ii) a fixed ordering that auction the most valuable item first. In addition, we include a lower bound on the average revenue of an optimal ordering. This is computed by running simulations of 250 random orderings and selecting the one with the highest revenue.

Table 1: The performance with (top to bottom): 8 myopic agents, 8 smart agents, and 8 simulator agenst.

ordering strategy	1	2	3	4	5	sum
random	192	252	216	246	230	1136
most valuable first	176	214	184	251	223	1048
best first	204	247	225	245	229	1150
best first with sum	196	243	232	249	231	1151
best auction found	208	284	235	255	232	1214

ordering strategy	1	2	3	4	5	sum
random	260	237	231	158	262	1148
most valuable first	259	235	223	180	253	1150
best first	269	242	235	172	264	1182
best first with sum	269	228	232	176	266	1171
best auction found	278	259	237	183	279	1236

ordering strategy	1	2	3	4	5	sum
random	225	183	203	225	192	1028
most valuable first	212	182	204	225	180	1003
best first	240	181	209	234	193	1057
best first with sum	237	181	211	234	194	1057
best auction found	239	191	216	323	197	1166

The results with myopic agents, smart agents, and simulator agents are shown in Table 1.[1] Our preliminary results are encouraging as they show that our method is able to learn a good ordering for a complex auction setting (4 resource types with random values, popularities, and complementarities, of which 2 to 5 of each are auctioned every auction) from a small amount of examples (250 historical orderings along with their revenues). In addition, surprisingly they show that a random ordering strategy also performs well in complex auction settings. This is counter-intuitive. Further investigations are required to explain this result.

4. REFERENCES

[1] Wedad Elmaghraby. The importance of ordering in sequential auctions. *Manage. Sci.*, 49:673–682, May 2003.

[2] Sicco Verwer and Yingqian Zhang. Revenue prediction in budget-constrained sequential auctions with complementarities. Technical report, August 2011.

[1] We refer to [2] for more results.