Reinforcement Mechanism Design, with Applications to Dynamic Pricing in Sponsored Search Auctions

May 2020

1 Introduction

Internet Advertising is a proven profitable model. In search engines, when user hits a search query, along with the result page we have few ads related to the keywords of search. For any keyword, Advertisers place their bids for these ad slots, search engines computes price for slots and allocates them accordingly. Now, based on bidder's advertisement feedback, i.e. number of impressions, click-through-rates, price of slots, etc. bidders modify their bids. So throughout bidders behaviour and number of bidders keeps changing. In general these sponsored search auction follow Generalized Second Price Auctions with reserved price for each bidder (based on Myerson). Here we assume, that bids depends on type of bidders (private information), all bidders are rational and there's common knowledge. But in reality, bids depends on lot of parameters , bidders have complex private information and arbitrary rationality levels. Based on the feedback received, bidders keep updating, behaviour is dynamic and can change. So instead of static reserve prices for each bidder, we need dynamic reserve prices.

2 Problem Addressed

How to set dynamic reserve price in sponsored search auctions (Generalized Second Price Auctions with reserve price) such that it maximizes revenue without involing unrealistic assumptions like rationality, simple private information, common knowledge, etc. Also how frequently how change these reserve price.

3 Solution

The paper takes a data driven approach, instead of standard game theoretic assumptions. First it trains Buyer's behavior model (Markov Model) using real bidding data set, i.e. train the model based on the feedback KPI (Key Performance Index) received after each round and past bidding data instead of private information. Once we have our Buyer's model, to find reserve price that maximizes revenue, we form a Markov Decision Process, and solve with using Monte Carlo Search Tree.

Simulations were done on real search engine - Baidu. Results show that Markov model aligns with bidders behaviour and this mechanism outperforms static mechanisms like STATIC_OPT, BAIDU and STATIC_50 and dynamic mechanisms like GREEDY

4 Approach

4.1 Setting of problem

We consider auctions of single keywords. There are N bidders, K slots. For each slot, the probability that a user will click is called click-through-rate, denoted by q_k for k^{th} slot. Assume $q_1 \geq q_2 \geq \ldots \geq q_K \geq 0$ In Baidu, K is 3. Each bidder i reports bid b_i, bid profile is $b = (b_1, b_2, \ldots, b_n)$. We consider that b_i. bidder i gets i^{th} advertisement on winning. Mechanism $\mathcal{M} = (x, p)$ where $x \in [0, 1]^N$ is the allocation vector and $p \in R^N$ is reserved price, p_i is the reserved price of bidder i When a user clicks on bidder i's advertisement, his payment will be FORMULA HERE

4.2 Bidder Behaviour Model

We assume that bidders follows time homogeneous Markov property i.e. bids update their bids only based on their previous bids and the feedback KPI (Key Performance Index) provided by the system. Let $s_i^{(t)}$ denote bid distribution of bidder i at start of round t, and $h_i^{(t)}$ denote the KPI received after the end of round $t-1$, so our model will predict $s_i^{(t+1)}$ based on this two. And on the end of round t, the system will give us KPI $h_i^{(t+1)}$, and this goes on.

$$s_i^{(t+1)} = g_i(s_i^{(t)}, h_i^{(t)})$$

We assume that all the bidders bid in a finite space $\mathcal{B}$, and we are trying to learn the bid distribution over this space. For simulation, we have discretize the distribution over 100 intervals, so $s_i^{(t)}$ is a 100 x 1 vector, each entry containing probability of bidding in that interval. KPI of bidder i, at the end of round t, is scalar value i.e. encoding information about the number of impressions, the number of clicks and the amount of payments.

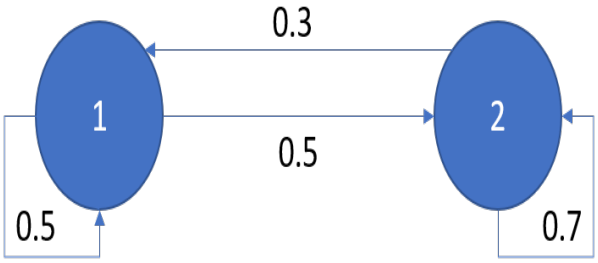

Figure 1: Example of a bidder model

In the above image, we consider bidder can either bidder 1 or 2. The transition table will be P = $\begin{bmatrix} 0.5 & 0.5 \\ 0.3 & 0.7 \end{bmatrix}$ p_{ij} is the probability of bidder to go from state i to state j. For example, if initially bidder was going to bid 1 with probability 1, then $s_0 = \begin{bmatrix} 1 & 0 \end{bmatrix}$, and then $s_1 = s_0 P$, i.e. $s_1 = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix}$ and so on.

To fit the function g_i, we implement a standard Long Short-Term Memory (LSTM) recurrent neural network with 128 units. We learn bidder behaviour model for each bidder, as each round proceeds, the model keeps getting updated.

4.3 Finding the optimal dynamic reserved price

4.3.1 Setting a Markov Decision Process

Now, we have already learnt our Markov model, we setup a Markov Decision Process.

For example, consider two bidders $N = 2$, bidder 1 has bid space $S_1 = \{0.2, 0.8\}$, and bidder 2 has bid space $S_2 = \{0.5, 1.5\}$ So the states of our MDP will be $S = S_1 x S_2 = \{(0.2, 0.5), (0.8, 0.5), (0.2, 1.5), (0.8, 1.5), \}$.

Actions of a bidder will be all possible reserve price we can have to improve objective. For action space, we will explore small neighbourhood of current research price for stability and not making system too large. In the paper, reserve prices explored are 95%, 100% and 105% times the current reserve price for the bidder.

In this example, $R_1 = \{0.4, 0.7\}$, $R_2 = \{1, 1.5\}$ and $R = R_1 x R_2 = \{(0.4, 1), (0.4, 1.5), (0.7, 1), (0.7, 1.5), \}$ Reward of (s, r), i.e. on taking action r when in state s, will be the expected revenue when reserve price profile is r

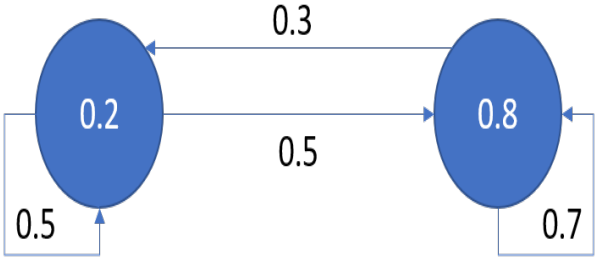

Figure 2: Example of a bidder 1 model

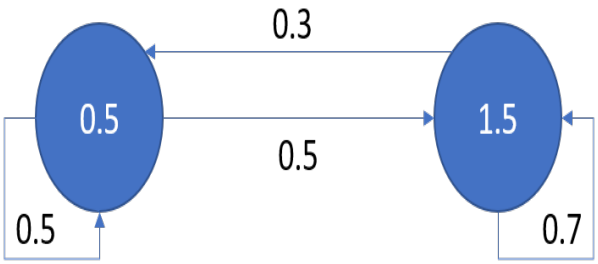

Figure 3: Example of a bidder 2 model

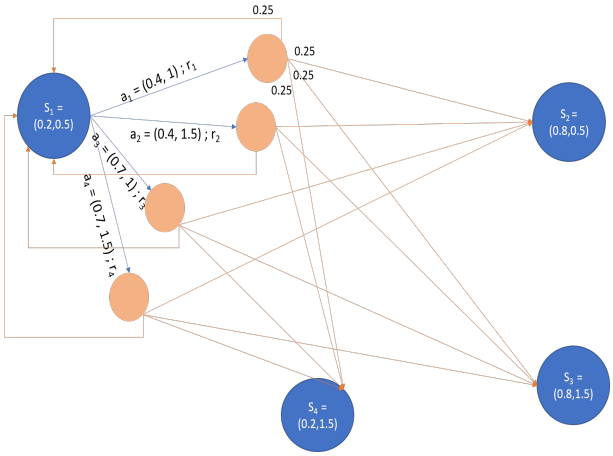

Figure 4: Example MDP (only showing actions from one state)

As seen in figure, once you are in an state, you take some action, you go to a chance node (orange nodes), and based on the probability you move to next node.

Formally defining, The long-term revenue maximization problem is a Markov decision process $(\mathcal{N}, S, R, G, REV(s, r), \gamma)$ where

- $\mathcal{N}$ is the set of bidders with $|\mathcal{N}| = N$
- $S = S_1 x S_2 x \dots x S_N$ is the state space where S_i is set of all possible bid distribution of bidder i
- $R = R_1 x R_2 x \dots R_N$ is the action space, where R_i is the set of all possible reserve prices that the mechanism designer can set for bidder i
- $G = (g_1, g_2, \dots, g_N)$ is the set of state transition functions
- REV(s, r) is the immediate reward function that gives the expected revenue for setting reserve price profile r when the state is s
- γ is the discount factor with $0 < \gamma < 1$

The objective is to select a sequence of reserve price profiles $\{r_t\}$ that maximizes the sum of discounted revenues

$$OBJ = \sum_{t=1}^{\infty} \gamma^t REV(s_t, r_t)$$

4.3.2 Problems with using traditional MDP solving algorithm

- Value iteration or Policy iteration : We have uncountable states, even after we do discretization. Computation expensive

- deep Q-learning network(DQN) or Asynchronous advantage actor-critic : depends highly on the bidder behavior model. agents leave and enter. do not have reasonable estimations of the Q-value for each state and action, which will slow down the training process

4.3.3 Monte Carlo Tree Search (MCTS)

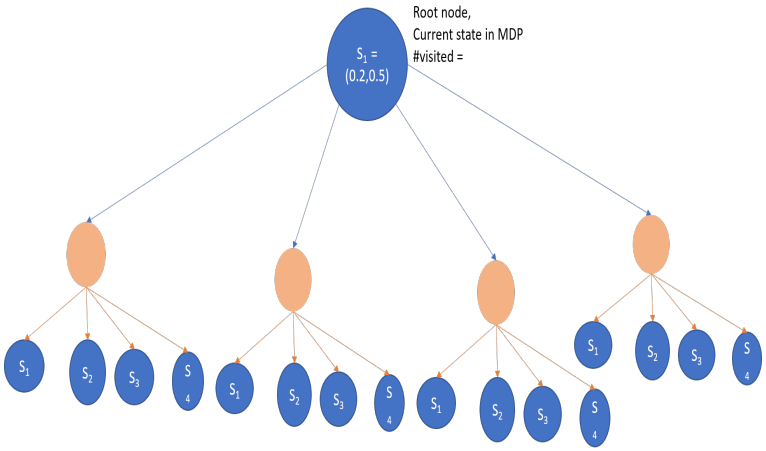

Figure 5: Example of MCTS

For MCTS, our current state in MDP will become the root node here. Further we choose a child node *i.e.* action from our current node based on some selection values (UCT). This traversal goes on until we reach a leaf node or maximum depth. From there we rollout, *i.e.* randomly choose actions (in simulation it was done for 5 millions auctions) and assign the expected revenue using backpropogation SARSA method. So there are three steps of this algorithm - Selection, Expansion and Rollout and Backpropagation.

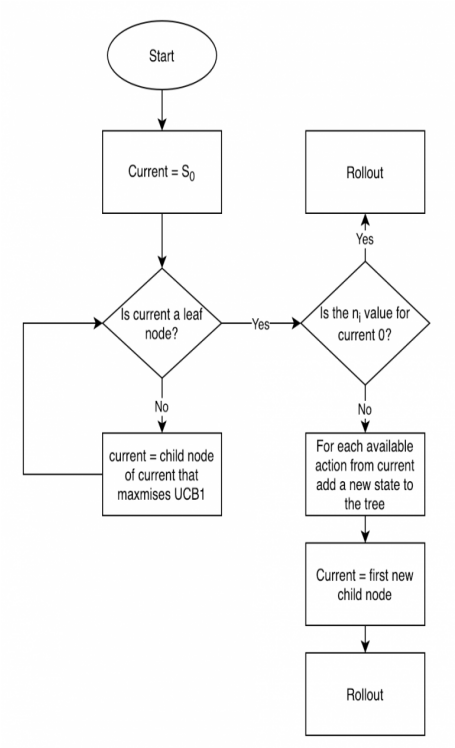

Figure 6: MCTS Algorithm (Ref:analyticsvidhya.com)

5 Experimental Results

This model achieve the best performance and gain higher revenue in the long run. And also changing reserve prices more frequently, will result into more revenue

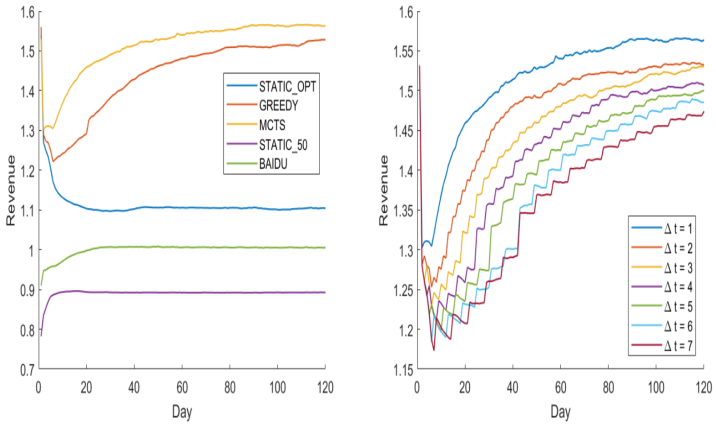

Figure 7: Simulation Results