A Reward Shaping Approach for Reserve Price Optimization using Deep Reinforcement Learning

A Reward Shaping Approach for Reserve Price Optimization using Deep Reinforcement Learning

Reza Refaei Afshar, Jason Rhuggenaath, Yingqian Zhang and Uzay Kaymak
Eindhoven University of Technology, Eindhoven, The Netherlands
Email: r.refaei.afshar@tue.nl, j.s.rhuggenaath@tue.nl, YQZhang@tue.nl, u.kaymak@jads.nl

***Abstract*—Real Time Bidding is the process of selling and buying online advertisements in real time auctions. Real time auctions are performed in header bidding partners or ad exchanges to sell publishers' ad placements. Ad exchanges run second price auctions and a reserve price should be set for each ad placement or impression. This reserve price is normally determined by the bids of header bidding partners. However, ad exchange may outbid higher reserve prices and optimizing this value largely affects the revenue. In this paper, we propose a deep reinforcement learning approach for adjusting the reserve price of individual impressions using contextual information. Normally, ad exchanges do not return any information about the auction except the sold-unsold status. This binary feedback is not suitable for maximizing the revenue because it contains no explicit information about the revenue. In order to enrich the reward function, we develop a novel reward shaping approach to provide informative reward signal for the reinforcement learning agent. Based on this approach, different intervals of reserve price get different weights and the reward value of each interval is learned through a search procedure. Using a simulator, we test our method on a set of impressions. Results show superior performance of our proposed method in terms of revenue compared with the baselines.**

***Index Terms*—Real Time Bidding, Reinforcement Learning, Reward Shaping, Deep Learning.**

I. Introduction

In recent years, *Real Time Bidding (RTB)* has become the main platform for trading online advertisements (ads). Considering its extremely high turnover, most website owners participate in this business by selling some blocks in their websites to the advertisers. In display advertising, these blocks are called ad slots or ad placements. Basically, selling and buying ad slots are performed through online RTB auctions. These auctions are performed in ad networks or Ad Exchange (AdX) markets in real time and the winner of an auction advertises in corresponding ad slot. The auctions are mainly second price auctions where the winner pays as much as the second highest bid. In second price auctions, the website owner adjusts a *reserve price* which determines the minimum price of the ad slot. The final price of sold ad slots is the maximum of the reserve price and the second highest bid which is paid to the ad publishers and the ad is shown in the end user's browser [1].

When a browser in an end user's computer loads a website, each ad slot in the newly viewed webpage generates an *impression*. These impressions are the assets to be sold in RTB auctions. For each impression, an *ad request* is sent to ad networks or AdX. Depending on the way of sending the ad requests, publishers opt either for the waterfall strategy or Header Bidding (HB) [2]. In the waterfall strategy, the requests are sent to the ad networks sequentially with separate reserve prices. In HB, the process of sending the ad requests to the so called Header Bidding Partners (HBPs) is performed simultaneously and each HBP returns its bid separately [3]. Normally, HB auctions are first price auctions and no reserve price is determined for them [4]. In a practical framework used in business, HB and waterfall strategy are combined where the publisher, first, sends ad requests to HBPs and receives their bids. Then, a reserve price is set according to the HBPs bids and another ad request is sent to an AdX. If AdX's winner bid is higher than the highest bid of HBPs, the impression goes to AdX winner, otherwise, it goes to the HBP winner [5].

In common practice, the highest bid of HBPs is the reserve price. Reserve price directly determines the revenue and higher reserve prices may increase the revenue. Since AdX might outbid higher reserve prices, using highest bid of HBPs as the reserve price may not be optimal. Therefore, the main problem of this work is to determine reserve price in impression level to uplift the revenue of ad publishers. Solving this problem is difficult in dynamic environments where the distribution of the bids is unknown. On one hand, if the reserve price is too high, appropriate bidder would hardly be found. On the other hand, low reserve prices affect the revenue of the publishers [6]. Therefore, optimizing the reserve price is an important task of ad publishers. In this paper, we focus on the problem of reserve price optimization from publisher's point of view.

The reserve price could be optimized using optimization methods if the bid distributions of the advertisers are known. Nevertheless, RTB environment is highly dynamic and the publisher has no information about the bids. In other words, we assume that the publisher does not have access to the AdX winner bid which is true for most RTB systems. Besides, the process of adjusting the reserve price can be considered as a sequential decision making task in a possibly infinite horizon where in each decision moment, an impression is generated and a reserve price is determined. In addition, as shown in [5], the bid values of HBPs and also the winner bid of AdX would not be predicted well without any information about the end users and advertisers. These reasons make the problem suitable for Reinforcement Learning (RL). Since the states and actions spaces are large and continuous, tabular RL can no longer be

helpful and so we opted for Deep Reinforcement Learning (DRL) as a proper function approximation method for such a problem.

Basically, in RTB auctions, the number of high-valued bids are less than the number of low-valued bids, because the bidders place high bids only for valuable impressions, which is not too much according to RTB historical data. For this reason, the agent trains a policy that mainly provides low-valued reserve prices. Although the number of high-valued bids is lower, they greatly influence the revenue. One way for solving this issue is to follow the idea of replay memory in [7]. However, this would not be helpful in our case because replay memory prevents forgetting and it does not change the distribution of observed bids while larger bids are of more importance.

In order to set appropriate high reserve prices, we propose a novel reward shaping method to learn the reward function according to the properties of pricing problems in auctions. To the best of our knowledge, this work is the first to shape and learn reward function using limited feedback of environment. Our proposed method extends the sold-unsold feedback of the RTB environment and assigns a reward vector and a weight vector for each impression. The inner product of these two vectors provides the scalar reward value, which is used during the training. In order to evaluate the method, we develop a simulator using historical RTB data to generate AdX's winner bids. Our contributions in this paper are as follows.

- Modeling the problem of reserve price optimization in RTB systems based on AdX and HB as DRL.
- Developing a reward shaping approach for pricing problems in auctions. Our model derives a scalar value for the reward by extending the binary feedback of RTB environments. This approach could be used for online pricing problems in general.
- Developing a simulator for RTB systems based on HB and AdX using probability density functions in order to estimate the winner bid of AdX's auction. The simulation is made public and can be used for other research.

II. Related Work

Several works in the literature focus on reserve price optimization, reward learning for reinforcement learning, reinforcement learning for real time bidding and real time bidding based on HB and AdX. In the following paragraphs, some of the recent works in each direction are reviewed.

a) Reserve Price Optimization: The impact of the reserve price on revenue is studied in [8]. In [9], setting up the reserve price for second price auctions is optimized by several empirical algorithms including optimal auction where the publisher knows bidder's bid distributions, modeling as a simple game between a publisher and advertisers, and algorithms based on Bayes' rules. In [10], the process of setting the reserve price in online RTB and offline channels is studied and the reserve price is set according to the winning probability of advertisers and valuation of impressions. The problem of multi-channel RTB is the main topic of some other works [11], [12]. We focus on setting up the reserve price in AdX and HB systems where no information about the bidders is available. This makes our work different from previous reserve price setting works.

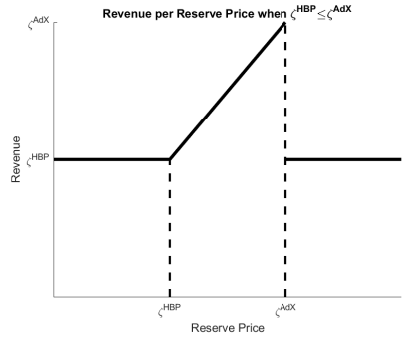
b) RL for RTB: Most of previous works in this direction model the problem from advertiser side. In [13] deriving optimal bidding setting for each impression is performed by DRL in sponsored search auctions, which is quite different from display advertising. In [14], A3C algorithm is generalized for multi-objectives in RTB setting to optimize the bidding process. In [15], RL is used for dynamic pricing in sponsored search auctions. The problem of allocating impressions to guaranteed contract or RTB is modeled by multi-agent RL in [16]. Multi-agent RL is also used to optimize the bidding strategy of the advertisers [17]. Ordering the ad networks in the waterfall strategy is another problem in RTB, which is modeled by RL in [18], [19]. To the best of our knowledge, no work studied the problem of reserve price optimization in AdX and HB systems using RL.

c) Reward Learning: Although the reward function is usually defined by experts, learning reward functions and reward shaping may show promising improvements of the performance. In [20], the theoretical implications of potential-based reward shaping and difference rewards in single objective multi-agent RL are discussed. In potential-based reward shaping each state contains a potential which expresses a preference for the agent [21], and a difference reward is a shaped reward that quantifies each agent's individual contribution to a multi-agent system [22]. Application of reward shaping in spoken dialogue systems is discussed in [23], where a set of recurrent neural networks are trained to provide reward. Recently, a reward shaping approach is developed for robot navigation that learns the reward value by inner product of observations and weights, where the weights are learned during a pre-processing step [24]. However, this method is not applicable on our problem because the provided information by environment is limited to sold-unsold binary value and this makes our reward shaping approach different from other works. The purpose of our algorithm is to enrich the reward value using impression properties and the highest bid of HBPs.

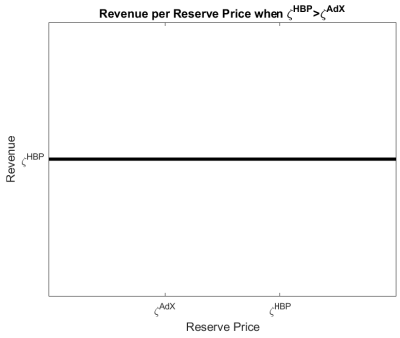
d) RTB systems containing AdX and HB: This area is relatively new and few works study the HB environments. The failure rate of reserve price in HB and AdX systems is studied in [25] using survival analysis. The idea of survival analysis is extended in [5] to increase the reserve price. In [26], the focus is on optimizing the revenue of AdX. Modeling as multi-armed bandit is another approach for optimizing the publisher's strategy in HB which is explored in [27] and [28]. We develop a novel DRL-based model for optimizing the reserve price using impression information information.

III. Problem Definition and Method Overview

The problem is to determine a reserve price for each impression that is sent to AdX. Upon generating an impression, first, an ad request is sent to HBPs and their bids are received. Then, another request containing a reserve price is sent to AdX



(a) $\zeta_t^{HBP} \leq \zeta_t^{AdX}$



(b) $\zeta_t^{HBP} > \zeta_t^{AdX}$

Fig. 1: The revenue per reserve price based on the relation between ζ_t^{HBP} and ζ_t^{AdX}.

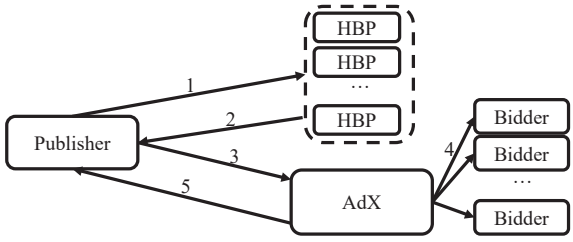


Fig. 2: RTB system based on HB and AdX. The process of selling impression t is as follows:(1) sending requests to HBPs, (2) receiving ζ_t^{HBP}, (3) sending an ad request containing ζ_t^{HBP} to AdX, (4) AdX runs an auction among bidders, (5) receiving a sold-unsold binary value β_t.

and the response is a sold/unsold binary value. This system is shown in Fig. 2.

Let ζ_t^{HBP} be the highest of all HBPs bids and ζ_t^{AdX} be the winner bid of the auction run in AdX for impression t which is not observable for the publisher due to the blackbox nature of AdX. According to the values of ζ_t^{HBP} and ζ_t^{AdX}, two situations are possible as illustrated in Figs. 1(a) and 1(b):

(a) $\zeta_t^{AdX} \geq \zeta_t^{HBP}$: In this case, the impression goes to the winner bidder of AdX's auction.

(b) $\zeta_t^{AdX} < \zeta_t^{HBP}$: In this case, the impression goes to HBP that provided the highest bid regardless of the AdX's auction. For all values of the reserve price, the revenue is fixed as shown in Fig. 1(b).

The first situation is the one that optimizing the reserve price uplifts the revenue. In this case, any reserve price between ζ_t^{HBP} and ζ_t^{AdX} increases the revenue as shown in Fig. 1(a). The reserve price of an impression t is denoted by a_t. Although the ζ_t^{HBP} is usually used as a_t for AdX's auction, higher values may also be outbid by AdX. Upon sending a request to AdX, it sends back a binary value β_t determining the status of the auction for impression t. This value is one, if the impression is sold. Otherwise it is zero. Since the reserve price directly influences the revenue, higher reserve prices where $\beta_t = 1$, lead to higher revenue. Therefore, the problem is to set a reserve price a_t for an impression t using the feature vector s_t. The feature vector s_t contains the information of an impression that is sent to the AdX. This feature vector is elaborated in Section IV-B.

Our proposed method is based on DRL and a variant of actor-critic policy gradient methods. In this algorithm, a separate Deep Neural Network (DNN) is trained for each of the policy and the value functions. The inputs of the policy and the value networks are the impression information. The output of the policy network is a probability distribution for the reserve price and the output of the value network is a value for the input state. The state is the feature vector corresponding to an impression and the action is the reserve price. The reward function $R(a_t, \zeta_t^{HBP}, \beta_t)$ is obtained by the inner product of a reward vector $\vec{r}(a_t, \zeta_t^{HBP}, \beta_t)$ and a weight vector $\vec{w}$. Since the only feedback from AdX is β_t, our proposed reward shaping method extends this feedback by assigning varying priorities for different intervals in the reserve price domain.

IV. DRL METHOD FOR RESERVE PRICE OPTIMIZATION

The process of adjusting the reserve price for each impression is performed by using a DNN policy that is trained by DRL. In each decision moment, our proposed method receives the information of a generated impression and returns the reserve price for the AdX's auction.

A. DRL framework

Among DRL approaches, value-based methods like DQN [29] are not suitable, because the action space is continuous for our method and there is no guarantee for the convergence of the value-based method with a continuous action space. Besides, it is not possible to consider a separate output in the Q network of DQN for every actions, because there are numerous actions in continuous space. For these reasons, we opted for an actor-critic policy gradient method to train the policy and the value networks.

The policy and the value networks are denoted by $\pi(a_t|s_t, \theta_t^{\pi})$ and $V(s_t, \theta_t^{V})$ respectively, where a_t is the action, s_t is the state, and θ_t^{π} and θ_t^{V} are the parameter sets of the policy DNN and the value DNN, respectively. These two DNNs and their parameters are updated using the policy gradient learning algorithm shown in (1) and (2) [30].

$$\theta_{t+1}^{\pi} = \theta_t^{\pi} + \alpha \nabla_{\theta^{\pi}} \ln(\pi(a_t|s_t, \theta_t^{\pi}))A_t \quad (1)$$

$$\theta_{t+1}^{V} = \theta_{t}^{V} + \alpha \nabla A_t^2, \tag{2}$$

where $A_t = A(s_t, a_t) = r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$ is the advantage function. This algorithm is called advantage actor-critic (A2C) [31].

Due to the exploration in the environment, taking some actions may cause large gradients. Large gradients update the DNN parameters dramatically which is usually not suitable. In order to prevent this, we use Proximal Policy Optimization (PPO) algorithm which clips the gradient to be in a particular interval [32]. The surrogate objective function of PPO which is used for updating the policy DNN, is shown in (3).

$$L^{CLIP}(\theta^{\pi}) = \hat{\mathbb{E}}\left[\min(e_t(\theta^{\pi})A_t, \text{clip}(e_t(\theta^{\pi}), 1-\epsilon, 1+\epsilon)A_t)\right]. \tag{3}$$

$$e_t(\theta^{\pi}) = \frac{\pi(a_t|s_t, \theta^{\pi})}{\pi(a_t|s_t, \theta^{\pi}_{old})}, \tag{4}$$

where, θ^{π}_{old} and θ^{π} are the policy parameters before and after updating, respectively. The clip function clips the gradient if it is larger than $1+\epsilon$ or smaller than $1-\epsilon$ for a hyper-parameter ϵ.

B. State and Action Definition

A state representation corresponding to an impression is the input of the policy and the value DNNs in the PPO algorithm. Basically, this state definition could be used in every online pricing tasks where the relation between revenue and reserve price is similar to Fig. 1(a). In this work, we focus on setting the reserve price for impressions in RTB as an example of pricing in auctions.

Common impression information in RTB systems based on HB and AdX are URL, size and location of ad slot and time of generating the impression. Due to GDPR[1], user data are not used in our method. The aforementioned information and ζ_t^{HBP} constitute the state s_t for impression t as shown in (5).

$$s_t = (\varphi_t, \Upsilon_t, \xi_t, \ell_t, \tau_t, \zeta_t^{HBP}), \tag{5}$$

where φ_t is a unique identifier for the ad slot generating the impression t, Υ_t is the URL of the webpage containing the ad slot, ξ_t is the size of the ad slot, ℓ_t is the location of the ad slot, τ_t is the time of sending the ad request and ζ_t^{HBP} is the highest bid of HBPs.

The action a_t is the value of the reserve price which is obtained from the output of the policy network. Since a_t is continuous, the policy network has a single output. Basically, the policy network provides a continuous probability density function (PDF) over possible continuous actions. This PDF is a Gaussian PDF whose mean is directly obtained from the output of the policy network and its standard deviation is fixed. The reserve price a_t is sampled from Gaussian PDF $\mathcal{N}(\mu, \sigma)$ for each impression t, where μ and σ are mean and standard deviation respectively.

[1]General Data Protection Regulation, https://gdpr-info.eu/.

V. Reward Shaping

AdX receives an ad request containing a_t and returns β_t to the publisher. The first option for the reward function is to use this binary value. In this situation, the reward is 1 if $\zeta_t^{HBP} < a_t \leq \zeta_t^{AdX}$. This reward is not helpful for increasing the revenue because for all the a_t between ζ_t^{HBP} and ζ_t^{AdX}, the reward is the same and the agent can only learn to sell the impressions in AdX's auctions rather than optimizing the revenue.

The second option is to use the revenue of each impression as the reward function according to Fig. 1(a). The reward function of this case is shown in (6).

$$R(a_t, \zeta_t^{HBP}, \beta_t) = \begin{cases} \zeta_t^{HBP} & a_t < \zeta_t^{HBP} \\ a_t & \beta_t = 1 \text{ and } a_t \geq \zeta_t^{HBP} \\ \zeta_t^{HBP} & \beta_t = 0 \text{ and } a_t \geq \zeta^{HBP} \end{cases} \tag{6}$$

This reward function is also not very useful for increasing the revenue because the trained agent prefers to take reserve prices close to ζ_t^{HBP} to ensure that the β_t is one and the revenue is higher than ζ_t^{HBP}. In other words, the agent learns to set a_t slightly higher than ζ_t^{HBP} which is most probably outbid by AdX's auction but it affects the revenue. Since there is no information about ζ_t^{AdX} and this value is variable for different impressions, the DRL agent learns to set a_t in an interval where β_t is most likely one. As shown in section VI, the number of large ζ_t^{AdX} is small and setting a large a_t increases the risk of not being sold by AdX's auction.

In order to increase the revenue by setting larger a_t, we develop a novel reward shaping approach inspired from [24]. Although setting larger reserve prices increases the risk of being unsold, the total revenue could be increased. Unlike [24], in our work, the only feedback from the environment is β_t and there is no more information to generate a reward vector. Following the fact that larger a_t may lead to higher revenue, the main objective of the reward shaping is to assign proper weight to larger a_t. To achieve that, the interval between ζ_t^{HBP} and ζ^{max} which is the average of all ζ_t^{AdX} in the training, is divided into n equal sub-intervals and a particular weight w_j for $j \in \{1, ..., n\}$ is assigned to the interval j. Vector $\vec{w} \in \mathcal{W}$ contains the weights w_j and $\mathcal{W}$ is the space of candidate weights vectors. Vector $\vec{r}(a_t, \zeta_t^{HBP}, \beta_t)$ assigns a reward $r_{t,j}$ to each interval j. Let $\mathcal{D}$ be the set of possible definitions for reward function and $d \in \mathcal{D}$ be a strategy of defining $\vec{r}(a_t, \zeta_t^{HBP}, \beta_t)$. For example, setting a_t as $r_{t,j}$ if a_t is in interval j and setting zero for other entries of $\vec{r}(a_t, \zeta_t^{HBP}, \beta_t)$, is a possible strategy. Assuming interval zero for values smaller than ζ_t^{HBP} and interval $n+1$ for values larger than ζ^{max}, the inner product of vectors $\vec{w} = (w_0, w_1, ..., w_{n+1})$ and $\vec{r}(a_t, \zeta^{HBP}, \beta_t) = (r_{t,0}, r_{t,1}, ..., r_{t,n+1})$ provides the reward value for each impression as shown in (7).

$$R(a_t, \zeta^{HBP}, \beta_t) = \vec{r}(a_t, \zeta^{HBP}, \beta_t) \cdot \vec{w}. \tag{7}$$

By fine-tuning the definitions of $\vec{r}(a_t, \zeta^{HBP}, \beta_t)$ and the values of weight vector $\vec{w}$, the agent could utilize the learned

reward function and train a suitable policy. Algorithms 1 and 2 show our proposed method.

Algorithm 1 DRL using Learned Reward

Input: A set of impressions $\mathcal{I}^{train}$, number of different weight vectors and reward definitions T, reward definitions $\mathcal{D}$ and space of candidate weights vectors $\mathcal{W}$
Output: Policy Network $\pi(a_t|s_t,\theta_t^\pi)$

1: Initialize all the entries of best weights vector $\vec{w}$ with zero
2: $d \leftarrow none$, $p \leftarrow 0$ {d: best reward definition, p:maximum total reward by summing up the reward of impressions}
3: $t \leftarrow 0$
4: $\nu \leftarrow$ number of impressions in $\mathcal{I}^{train}$ for training, e.g. 3×10^5
5: **repeat**
6: $\vec{w}^c$, $d^c \leftarrow GetCandidateParameters(\mathcal{W},\mathcal{D})$
7: $\pi(a_t|s_t,\theta_t^\pi) = PPOAgent(\mathcal{I}^{train},\nu,\vec{w}^c,d^c)$
8: Find $p^c = \sum_{t\in\mathcal{I}^{train}} R(a_t,\zeta_t^{HBP},\beta_i)$
9: **if** $p^c > p$ **then**
10: $p \leftarrow p^c$, $d \leftarrow d^c$, $\vec{w} \leftarrow \vec{w}^c$
11: **end if**
12: $t \leftarrow t+1$
13: **until** $t \geq T$
14: **return** $PPOAgent(\mathcal{I}^{train},\nu,\vec{w},d)$

Algorithm 2 $PPOAgent$

Input: A set of impressions $\mathcal{I}^{train}$, number of impressions ν, weight vector $\vec{w}$, reward definition d
Output: Policy Network $\pi(a_t|s_t,\theta_t^\pi)$

1: Initialize $\pi(a_t|s_t,\theta_t^\pi)$ and $V(s_t,\theta_t^V)$ networks
2: $i \leftarrow 0$
3: **repeat**
4: Select an impression $t \in \mathcal{I}^{train}$
5: Set a_t according to $\pi(a_t|s_t,\theta_t^\pi)$
6: Find $\vec{r}(a_t,\zeta_t^{HBP},\beta_t)$ according to d
7: $R(a_t,\zeta_t^{HBP},\beta_t) \leftarrow \vec{r}(a_t,\zeta_t^{HBP},\beta_t)\cdot\vec{w}$
8: Update θ^π and θ^V according to (3) and (2)
9: $i \leftarrow i+1$
10: **until** $i \geq T$
11: **return** $\pi(a_t|s_t,\theta_t^\pi)$

Algorithm 1 works as follows: line 1 initializes the best weight vector with zero. Line 2 initializes the best reward definition and the best total reward, and line 4 defines the number of impressions for training. We consider $\nu = 3 \times 10^5$ as the method converges and overfitting is also prevented. Line 6 provides candidate weights and reward definition based on a black box optimization method. This method could be as simple as line search among some candidate values or more sophisticated methods like Bayesian optimization. Line 7 trains the PPO agent with candidate weights and reward definition, and line 8 finds the total expected revenue by using the policy of line 7. If the performance is improved, the best weights and reward definition are updated in line 10.

Algorithm 2 starts with initializing $\pi(a_t|s_t,\theta_t^\pi)$ and $V(s_t,\theta_t^V)$. Then the PPO algorithm is performed in lines 4 to 9 and the reward value is yielded using d and $\vec{w}$. Finally, the policy π is returned from the algorithm to be used for setting the reserve price.

VI. RTB Simulator

Interacting with an RTB system is the best way to train the agent. However, it would affect the revenue of the ad publisher, because the agent needs to explore the environment by taking non-optimal actions. For this reason, using real environment for training and evaluating the method is not practical and we opt for developing an RTB simulator to provide the opportunity of exploration for the agent. In order to develop a simulator, ζ_t^{AdX} needs to be generated.

As mentioned before, AdX provides no information about ζ_t^{AdX} while this value is crucial for determining whether a reserve price is outbid. We use RTB historical data that contains the information shown in (5) and β_t in order to estimate ζ_t^{AdX} for each impression t.

For generating ζ_t^{AdX}, we use ζ_t^{HBP} and β_t that are included in the historical data. The historical data is from an RTB system based on HB and AdX where ζ_t^{HBP} is used as a_t. According to the value of β_t, we can conclude that ζ_t^{AdX} is larger than ζ_t^{HBP} if $\beta_t = 1$. For these impressions, AdX is winner and ζ_t^{HBP} is a lower bound for ζ_t^{AdX}. Therefore, we use ζ_t^{HBP} for all t where $\beta_t = 1$ to generate ζ_t^{AdX}.

For this purpose, first, all the impressions with $\beta_t = 1$ are retrieved. Second, these impressions are grouped based on the values of features φ_t, Υ_t, ξ_t, ℓ_t and τ_t to obtain a list of ζ_t^{HBP} for each set of features. Third, for each set of aforementioned features, a PDF is drawn using the list of ζ_t^{HBP}. The histogram of ζ_t^{HBP} for four randomly selected features sets are shown in Fig. 3. We tested some PDFs including skew-normal, half normal and Beta distributions. The fitted curves using Beta distribution are shown in Fig. 3. We fit a Beta distribution for each features set and ζ_t^{AdX} is sampled from corresponding distribution for each impression.

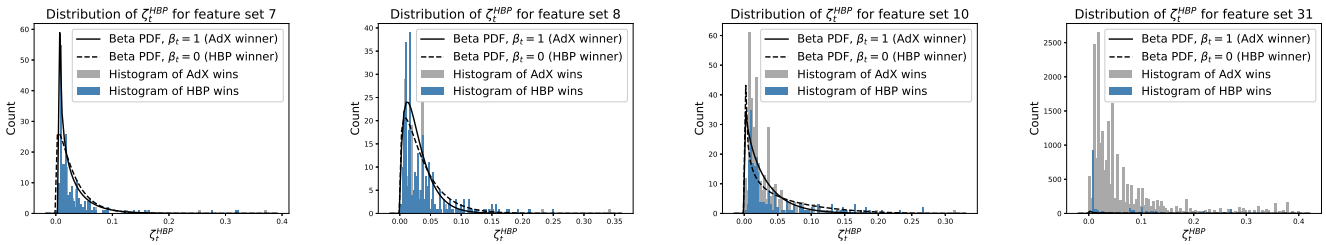
Figs. 3 shows the PDF of ζ_t^{HBP} when either AdX or HBPs are the winner. Figs. 3(a), 3(b) and 3(c) correspond to feature sets that are roughly rare and the number of observed impressions for them is low. In contrast, Fig. 3(d) depicts the histogram of a frequent feature set. The fitted PDF curves are not visible, because the area under the PDF is one and they are very close to the horizontal line. We do not scale the figure to show the curves because the number of impressions are not visible by scaling. Fig. 3(d) illustrates that the number of impressions that goes to AdX is far more than the impressions that goes to HBPs. This shows the importance of setting optimal reserve prices.

VII. Experiments and Results

A set of impressions containing the features URL, slot id, time, size, location of ad slot, ζ_t^{HBP} and β_i is used to evaluate the method. This historical data is obtained from our industrial partner and ζ_t^{AdX} is added to it by using the simulator presented in Section VI. Before explaining the baselines and the results, the details of the implementation is elaborated.

A. Configuration of the method

The policy and value DNNs are two separate networks and each has two hidden layers of 64 nodes. In order to capture



(a) Feature set a. (b) Feature set b. (c) Feature set c. (d) Feature set d.

Fig. 3: The distributions of ζ_t^{HBP} for four different sets of features.

the information of previously set reserve prices as time series information, LSTM cells are used in the hidden layers. Our exploratory experiments show superior performance of using LSTM layers. Learning rate is set to 0.001.

The weights vector $\vec{w}$ is a set of positive integer numbers. Since each run takes around twenty minutes, it is possible to test different candidate weights in reasonable time. A set of integer numbers is also considered to learn the number of intervals n. By running algorithm 1 on candidate weights and intervals, $n = 3$ and $\vec{w} = (1, 1, 10, 4, 1)$ provide better performance and higher total reserve price.

In order to learn reward definition d, the definitions showed in Table I are tested. These definitions are mainly based on the value of reserve price and the lower bound of each interval. For each of them, all the $r_{t,j}$ are zero except the one corresponding to the interval containing a_t. Formally speaking, let l_j be the lower bound of interval j. Based on this definition, l_1 and l_{n+1} are ζ_t^{HBP} and ζ^{max} respectively. If $l_j \leq a_t < l_{j+1}$, then $r_{t,j}$ is not zero, otherwise it is zero. Similarly, $r_{t,0}$ is not zero if $a_t < l_1$ and $r_{t,n+1}$ is not zero if $a_t > l_{n+1}$. Among the definitions shown in Table I, the reward definition with index 5 is selected for learning the policy since it has the best performance. Fig. 4 shows the reward based on the reserve price. The intervals for $n = 3$ is also shown in this figure.

$$r_{t,j} = \begin{cases} a_t - \zeta_t^{HBP} & a_t < l_1 \\ \beta_t(a_t - \zeta_t^{HBP}) & l_j \leq a_t \leq l_{j+1}, \\ & s.t.\ j \in \{1, ..., n\} \\ \zeta^{max} - a_t & a_t > \zeta^{max} \end{cases} \quad (8)$$

The first term of (8) is a negative value to penalize reserve prices lower than ζ_t^{HBP}. Similar penalty is assigned for very large reserve prices. The second term is a positive reward if the reserve price is outbid in AdX's auction. Otherwise, the reward is zero and this is determined by the value of β_t. If $\beta_t = 1$, the positive reward is the distance between a_t and ζ_t^{HBP}. As an example, assume that $n = 3$, $\vec{w} = (1, 1, 10, 4, 1)$, $\zeta_t^{HBP} = 0.09$, $\zeta_t^{max} = 0.27$ and $\beta_t = 1$. The lower bounds of intervals are $(l_1, l_2, l_3, l_4) = (0.09, 0.15, 0.21, 0.27)$. The reward vector for $a_t = 0.22$ is $\vec{r}(a_t, \zeta_t^{HBP}, \beta_t) = (0, 0, 0, 0.13, 0)$ and the reward value is $\vec{w} \cdot \vec{r} = 0.65$.

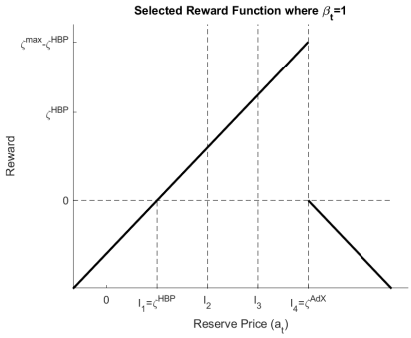


Fig. 4: The selected reward function when $n = 3$.

TABLE I: Candidate Reward Definitions $\mathcal{D}$. If a_t is not in interval j, $r_{t,j}$ is zero.

Index	$r_{t,j}$ if $l_j \leq a_t < l_{j+1}$	$r_{t,j}$ if $a_t < \zeta_t^{HBP}$	$r_{t,j}$ if $a_t > \zeta^{max}$
1	$c > 0$ (c is fixed)	$c > 0$	$c > 0$
2	$c > 0$	$-c$ where $c > 0$	$-c$ where $c > 0$
3	a_t	$-a_t$	$-a_t$
4	$a_t - l_j$	$a_t - \zeta_t^{HBP}$	$\zeta_t^{max} - a_t$
5	$\beta_t(a_t - \zeta_t^{HBP})$	$a_t - \zeta_t^{HBP}$	$\zeta_t^{max} - a_t$

B. Baselines and Metrics

Our proposed method with learned reward is denoted by DRL-LR. Each test data contains 10000 impressions with $\zeta_t^{AdX} > \zeta_t^{HBP}$ because for other impressions the winner is HBP regardless of the value of a_t. Moreover, ζ_t^{AdX} is larger than a fixed threshold in order to evaluate the method on valuable impressions which can highly affect the revenue. For smaller ζ_t^{AdX}, the impressions mainly goes to HBPs and the difference in revenue is negligible. The result of applying our proposed approach is compared with the following baselines.

- H3-2: This approach is a heuristic method. Based on this method, the interval between ζ_t^{HBP} and ζ_t^{max} is divided into $n = 3$ equal sub-intervals and $a_t = l_2$ is the reserve price. This approach is tested for all $l_j, j \in \{1, ..., n+1\}$ and $j = 2$ works better than the others.
- H5-3: This is similar to H3-2 where $n = 5$ and $j = 3$.
- DRL-DA: This approach trains a DRL agent with discrete actions. The intervals and the definitions of n and l_j are the same as our proposed method. However, there are n discrete actions where each one corresponds to a fixed

reserve price. The values of $l_j,\ j \in \{1, ..., n\}$ represent the set of possible actions. In our experiments, n is 3.

- DRL-RTB: This approach is a DRL approach which its reward function is defined by (6). The other parameters and configuration of the method is the same as DRL-LR.
- SA-PM: This method is proposed in [5] which is based on survival analysis and prediction models. Although in [5] the bids and AdX responses are based on predicted values, we use historical data.

The metrics that used to compare different baselines are as follows.

- $\sum_t \zeta_t^{AdX}$: This is the sum of all AdX's winner bids which is considered as an upper bound for the revenue.
- $\sum_t \zeta_t^{HBP}$: This is the sum of the highest bids of HBPs for all impressions.
- $\sum_t a_t$: This value shows the revenue of each algorithm. Since, the second highest bid of AdX is unknown for the agent, the revenue is considered as the sum of reserve prices for all impressions. Note that this value is obtained from policy network if $\beta_t = 1$, Otherwise $a_t = \zeta_t^{HBP}$.
- $\%_{a_t}$: This value is the performance ratio, measured by $\sum_t a_t / \sum_t \zeta_t^{AdX}$.

C. Results

Table II shows the results of performing different algorithms on $\mathcal{I}_1^{test}$ and $\mathcal{I}_2^{test}$. Each of these two sets contains 10000 randomly selected impressions of a particular day. The policy network is trained on the $\mathcal{I}^{train}$ containing the impressions of another day. This policy network is evaluated by applying on the impressions of the next two days. Hence, $\mathcal{I}^{train}$, $\mathcal{I}_1^{test}$ and $\mathcal{I}_2^{test}$ correspond to three consecutive days. The results show that the policy network works well in determining the reserve prices for the impressions of coming days.

TABLE II: Results of applying different algorithms

Test Data	Algorithm	$\sum_t \zeta_t^{HBP}$	$\sum_t a_t$	$\sum_t \zeta_t^{AdX}$	$\%_{a_t}$
$\mathcal{I}_1^{test}$	DRL-LR	628.00	**1527.23**	2075.86	**73.57%**
	H3-2		1315.82		63.38%
	H5-3		1430.96		68.93%
	DRL-DA		1315.82		63.38%
	DRL-RTB		1437.20		69.23%
	SA-PM		986.88		47.54%
$\mathcal{I}_2^{test}$	DRL-LR	599.32	**1521.77**	2065.43	**73.67%**
	H3-2		1305.82		63.22%
	H5-3		1420.92		68.79%
	DRL-DA		1305.82		63.22%
	DRL-RTB		1414.03		68.46%
	SA-PM		955.09		46.24%

As shown in table II, DRL-LR outperform all the other algorithms in terms of revenue. Based on these results, the performance of DRL-DA and H3-2 are the same, because the trained policy always takes the lower bound of interval 2 as the reserve price which is the same as H3-2. This shows that discrete actions work no better than heuristics.

In order to evaluate the reserve price of individual impressions, the difference between a_t and ζ_t^{AdX} for the impressions

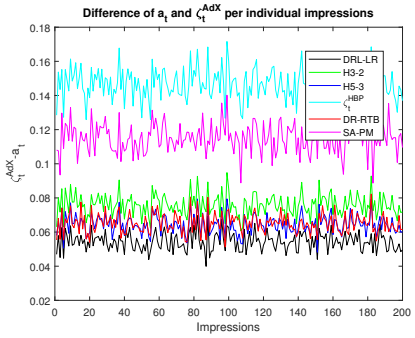


Fig. 5: Comparing individual reserve prices of different methods. In order to smooth the graph, average of 50 reserve prices is plotted.

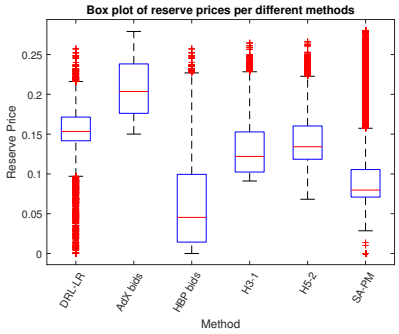


Fig. 6: Box plot of reserve prices for different baselines.

$\mathcal{I}_1^{test}$ is calculated and plotted in Fig. 5. These differences are averaged over every 50 impressions in order to have smoother graphs. The differences between a_t of DRL-LR and ζ_t^{AdX} is closer to zero than other algorithms and this shows closer to optimal reserve prices obtained by our proposed method.

The box plot of reserve prices of different approaches together with the ζ_t^{HBP} and ζ_t^{AdX} are shown in Fig. 6. Based on this figure, the median, lower quartile and upper quartile of reserve prices obtained by our proposed method are higher than the same parameters for the other baselines. In fact, not only the aggregated revenue of our proposed method is higher than the aggregated revenue of other baselines, but also individual reserve prices of our method are closer to ζ_t^{AdX}.

VIII. Conclusion and Future Work

In this paper, a DRL-based approach is presented for adjusting the reserve price of AdX's auctions in RTB systems based on HB and AdX. In order to improve learning and enrich the reward function, a novel reward shaping method is developed for defining the reward function. Moreover, a simulation using historical HB bids is built to derive the winner bids of AdX auction. Using the data obtained from simulation show that the revenue of an ad publisher may increase dramatically by applying our proposed method.

The main limitation of our work is lack of AdX's auction data. Typically, AdX provides aggregated revenue per hour or per day and there is no information about the revenue of single

impressions. If the distributions of the bidders are known, the problem could be solved by optimal auction algorithms. Besides, if the winner bids of AdX are available, optimization methods including DRL can use real data and the theoretical performance is more aligned with real performance. However, not only there is no information of bidder's distributions, but also the exact bids of winners are unknown. We manage this limitation by developing a simulator in this paper.

As future work, we aim to enhance the performance by improving the quality of the policy network. One possible approach is to make the weight vector more dynamic and embed different objectives in it. In other words, exploring the possibility of defining a dynamic weight vector that assigns different weights for different impressions is an interesting future direction. One other direction for future research is to consider social welfare and learn policies in a multi-agent RTB system including advertisers, HBPs and AdX.

ACKNOWLEDGMENT

This work was supported by EU EUROSTARS (Project E! 11582). The authors would like to thank the Headerlift Team and the Triodor R&D Team from Azerion and Triodor in collecting the data for this research, and dr. Murat Firat for his comments on an earlier version of this manuscript.

REFERENCES

[1] Y. Yuan, F. Wang, J. Li, and R. Qin, "A survey on real time bidding advertising," in *Proceedings of 2014 IEEE International Conference on Service Operations and Logistics, and Informatics*. IEEE, 2014, pp. 418–423.

[2] J. Wang, W. Zhang, S. Yuan *et al.*, "Display advertising with real-time bidding (rtb) and behavioural targeting," *Foundations and Trends® in Information Retrieval*, vol. 11, no. 4-5, pp. 297–435, 2017.

[3] M. Pachilakis, P. Papadopoulos, E. P. Markatos, and N. Kourtellis, "No more chasing waterfalls: a measurement study of the header bidding ad-ecosystem," in *Proceedings of the Internet Measurement Conference*, 2019, pp. 280–293.

[4] S. Despotakis, R. Ravi, and A. Sayedi, "First-price auctions in online display advertising," *Available at SSRN 3485410*, 2019.

[5] R. R. Afshar, Y. Zhang, M. Firat, U. Kaymak, A. I. Metin, G. S. Tarakçıoğlu, and C. Baş, "Reserve price optimization with header bidding and ad exchange," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2020, pp. 830–835.

[6] O. Busch, "Programmatic advertising," *New York: Springer*, vol. 10, pp. 978–3, 2016.

[7] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[8] J. Li, X. Ni, Y. Yuan, R. Qin, X. Wang, and F.-Y. Wang, "The impact of reserve price on publisher revenue in real-time bidding advertising markets," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2017, pp. 1256–1261.

[9] S. Yuan, J. Wang, B. Chen, P. Mason, and S. Seljan, "An empirical study of reserve price optimisation in real-time bidding," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 1897–1906.

[10] J. Li, X. Ni, and Y. Yuan, "The reserve price of ad impressions in multi-channel real-time bidding markets," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 2, pp. 583–592, 2018.

[11] J. Rhuggenaath, A. Akcay, Y. Zhang, and U. Kaymak, "Optimal display-ad allocation with guaranteed contracts and supply side platforms," *Computers & Industrial Engineering*, vol. 137, p. 106071, 2019.

[12] B. Chen, S. Yuan, and J. Wang, "A dynamic pricing model for unifying programmatic guarantee and real-time bidding in display advertising," in *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*, 2014, pp. 1–9.

[13] J. Zhao, G. Qiu, Z. Guan, W. Zhao, and X. He, "Deep reinforcement learning for sponsored search real-time bidding," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 1021–1030.

[14] C. Yang, J. Lu, X. Gao, H. Liu, Q. Chen, G. Liu, and G. Chen, "Motiac: Multi-objective actor-critics for real-time bidding," *arXiv preprint arXiv:2002.07408*, 2020.

[15] W. Shen, B. Peng, H. Liu, M. Zhang, R. Qian, Y. Hong, Z. Guo, Z. Ding, P. Lu, and P. Tang, "Reinforcement mechanism design: With applications to dynamic pricing in sponsored search auctions," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 02, 2020, pp. 2236–2243.

[16] D. Wu, C. Chen, X. Yang, X. Chen, Q. Tan, J. Xu, and K. Gai, "A multi-agent reinforcement learning method for impression allocation in online display advertising," *arXiv preprint arXiv:1809.03152*, 2018.

[17] J. Jin, C. Song, H. Li, K. Gai, J. Wang, and W. Zhang, "Real-time bidding with multi-agent reinforcement learning in display advertising," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 2193–2201.

[18] R. R. Afshar, Y. Zhang, M. Firat, and U. Kaymak, "A reinforcement learning method to select ad networks in waterfall strategy," in *11th International Conference on Agents and Artificial Intelligence, ICAART 2019*. SCITEPRESS-Science and Technology Publications, Lda., 2019, pp. 256–265.

[19] R. R. Afshar, Y. Zhang, M. Firat, and U. Kavmak, "A decision support method to increase the revenue of ad publishers in waterfall strategy," in *2019 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)*. IEEE, 2019, pp. 1–8.

[20] P. Mannion, S. Devlin, J. Duggan, and E. Howley, "Reward shaping for knowledge-based multi-objective multi-agent reinforcement learning," *The Knowledge Engineering Review*, vol. 33, 2018.

[21] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *ICML*, vol. 99, 1999, pp. 278–287.

[22] D. H. Wolpert, K. R. Wheeler, and K. Tumer, "Collective intelligence for control of distributed dynamical systems," *EPL (Europhysics Letters)*, vol. 49, no. 6, p. 708, 2000.

[23] P.-H. Su, D. Vandyke, M. Gasic, N. Mrkšić, T.-H. Wen, and S. Young, "Reward shaping with recurrent neural networks for speeding up online policy learning in spoken dialogue systems," in *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2015, pp. 417–421.

[24] H.-T. L. Chiang, A. Faust, M. Fiser, and A. Francis, "Learning navigation behaviors end-to-end with autorl," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2007–2014, 2019.

[25] A. Kalra, C. Wang, C. Borcea, and Y. Chen, "Reserve price failure rate prediction with header bidding in display advertising," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 2819–2827.

[26] R. Qin, Y. Yuan, and F.-Y. Wang, "Optimizing the revenue for ad exchanges in header bidding advertising markets," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2017, pp. 432–437.

[27] G. Jauvion, N. Grislain, P. Dkengne Sielenou, A. Garivier, and S. Gerchinovitz, "Optimization of a ssp's header bidding strategy using thompson sampling," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 425–432.

[28] J. Rhuggenaath, R. R. Afshar, A. Akcay, Y. Zhang, U. Kaymak, F. Çolak, and M. Tanyerli, "Maximizing revenue for publishers using header bidding and ad exchange auctions," *Operations Research Letters*, 2021.

[29] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[30] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[31] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

[32] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.