Functional Optimization Reinforcement Learning for Real-Time Bidding

Yining Lu*, Changjie Lu*, Naina Bandyopadhyay, Manoj Kumar, Gaurav Gupta

***Abstract*—Real-time bidding is the new paradigm of programmatic advertising. An advertiser wants to make the intelligent choice of utilizing a Demand-Side Platform to improve the performance of their ad campaigns. Existing approaches are struggling to provide a satisfactory solution for bidding optimization due to stochastic bidding behavior. In this paper, we proposed a multi-agent reinforcement learning architecture for RTB with functional optimization. We designed four agents bidding environment: three Lagrange-multiplier based functional optimization agents and one baseline agent (without any attribute of functional optimization) First, numerous attributes have been assigned to each agent, including biased or unbiased win probability, Lagrange multiplier, and click-through rate. In order to evaluate the proposed RTB strategy's performance, we demonstrate the results on ten sequential simulated auction campaigns. The results show that agents with functional actions and rewards had the most significant average winning rate and winning surplus, given biased and unbiased winning information respectively. The experimental evaluations show that our approach significantly improve the campaign's efficacy and profitability.**

***Index Terms*—Real-Time Bidding Optimization, Multi-Agents Deep Reinforcement Learning, Functional Optimization.**

I. Introduction

Advertising media has transitioned from television to online apps as the Internet has grown in popularity, making real-time advertising more accessible and prevalent. Advertisers have realized that the Internet allows them more flexibility and control over their advertising materials. Therefore it has become a widely used advertising medium. According to the advertising revenue report [1], programmatic advertising (automated buying and selling of online advertising) takes up about 81% of $57 billion non-search advertising in 2019.

Real-Time Bidding(RTB) based advertisement auction plays an essential role in daily life and becomes the new paradigm for internet advertising [2]. Advertisers and media buying agencies use DSP to place bids, which allows them to bid automatically. For each request from the users, the advertiser in DSP must give the bid price within 100 milliseconds [3]. DSP can have the information of the coming request, including the position of the advertisement, the information of the users, etc. DSP enable users to optimize based on pre-defined key performance indicators(KPIs) such as cost per click (CPC), cost per action (CPA), and their budget.

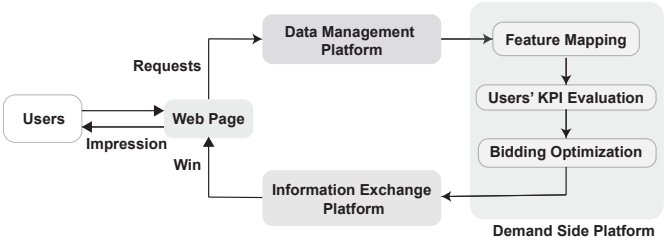


Fig. 1. The Workflow of RTB. The web page detect the users' visits and send the ad requirement to the data management platform. DSP provide optimal bids based on user's and advertisement historical data. The highest bidding price in the information exchange platform will win the impression and pay the second-highest price.

New challenges have arisen as a result of the introduction of RTB and header bidding technologies. Typically, there are two types of dynamic auctions, first-price auction, and second-price auction. In a first-price auction, multiple advertisers compete for impressions, with the highest bidder winning. Second-price auctions were created to allow marketers to bid up to the maximum amount allowed by their budget. Exploring the optimal bid price in a first-price auction is tricky as the advertiser will know only their bidding price and whether they win or not. However, in a second-price auction, the second-highest price will be charged for the winner, which means the winner could know two information: **(1)** its bid price and **(2)** the second-highest bid price from the market. The bidding becomes more competitive in a second-price auction because the DSP has a greater grasp of the situation by access to additional data. The workflow of RTB is described in the Fig 1.

In RTB campaigns, predicting the optimal bid price for each impression is one of the most common challenges. The bid request is won by the advertiser, who submits the highest bid price. Advertisers' objective is to win as many impressions as possible under the budget constraint by running the customized bidding strategies on DSP.

The pacing model is a prominent sort of bidding strategy for branding campaigns that focuses on budget control [4]–[6]. The game theory approach has been widely used in RTB problems, especially in the second price auction. Any pure strategy [7], [8] will be detected by others so that advertisers can modify their bid strategy and win effectively. Therefore, researchers often come up with mixed strategy [9] like operational research or parameter optimization problem [10]. In

* indicates equal contribution

Yining Lu, Changjie Lu, and Gaurav Gupta are with the School of Mathematics, College of Science and Technology, Wenzhou-Kean University, Wenzhou 325060, China(e-mail: yiningl@kean.edu, lucha@kean.edu, ggupta@kean.edu)

Naina Bandyopadhyay is with the Findability Sciences Private Limited, Mumbai 400057, India(e-mail: bnaina@findabilitysciences.com)

Manoj Kumar is with the Department of Computer Science, Babasaheb Bhimrao Ambedkar University, Lucknow 226025, India

addition, the efforts in [11]–[14] attempt to use even model-free reinforcement strategy to achieve higher performance. To deal with high dimension features, the advertisers apply the factorization method, such as DeepFM, IFM, FEFM, FWFM etc [15]–[18], which maps the high dimensional sparse data into latent space so that advertisers can learn users' behavior.

In this paper, we proposed a multi-agent reinforcement learning architecture, combining functional optimization and a model-free algorithm. We give agents different states, rewards, and actions associated with winning probability, click-through rate, etc. The main contributions of this paper are:

1) Designed functional optimization agents (FOAs) using Lagrange multiplier
2) Combination of deep reinforcement learning and functional optimization to learn the agent's causality.
3) Demonstrate the effectiveness of the functional optimization using the multi-agent bidding scenario.

The rest of this paper is organized as follows: we introduce some related works and assumptions in the next section, followed by functional optimization in Section III. We present the multi-agent reinforcement learning architecture and the detailed design of the FOA in Section IV. We evaluate the FOA in simulated data and analyze the experiments result in Section V. Finally, we conclude our work and future works in Section VI.

II. Related Work

A. Key Performance Indicator

KPI can be click-through rate(CTR) or click conversion rate(CVR) [13], and it influences bidders in making bidding decisions. Advertisers are more likely to bid at higher prices for users with higher KPIs to win these valuable impression opportunities. As KPI estimation is the fundamental step of RTB, its accuracy directly affects the estimation of the bidding price. Traditional KPI prediction models try to use machine learning methods, such as regression, random forest, and gradient boosting [19], [20]. However, these methods can only learn the users' features on the surface and cannot explain the high order feature interactions. Therefore, Rendle, in 2010, came up with the factorization machine, which can map the original features into high dimension latent vectors and determine their relationships in second or higher levels [21]. After that, the deep learning method increased rapidly and was gradually replaced. In 2017, Guo et al. proposed the DeepFM with integrating the Factorial Machine and deep learning and achieve good performance [15]. Since that, the modified method boosted [16], [17].

B. Winning Probability

The winning probability is one of the keys to finding the optimal price solutions. It works together with KPIs to influence bidding decisions. Computing the auction winning probability seems simple because the cumulative sum of winning densities is the winning probability of the corresponding price. And this cumulative density distribution is always monotonically increasing; However, it is biased because the distribution is only learned from winning data. On the contrary, Zhang et al., in [22] models the auction winning probability with a bid landscape based on nonparametric survival analysis for second-price auction. It learns from the winning bids as well as from the losing bids so that the learned winning probability distribution is unbiased.

C. Operational Approachs

The optimal RTB is typically treated as a dynamic programming problem. Within the constrain of the budget, DSP attempts to win more impressions with less cost, i.e., reach-based campaign. Therefore, [23]–[27] design the objective function to maximize the expected impression value and minimize the expected cost. Many authors often utilize the Lagrange multiplier method to solve this problem by converting it to a dual problem. However, they always assume the fixed distribution for some terms in the equation. For instance, Zhang et al., in [12] assume a convex relationship of wining probability and bid price. Karlsson et al., in [28] propose a concave relationship of impression value and optimal bid price.

D. Model-Free Approachs

To optimize the RTB campaign, Several criteria are examined, including the duration of the campaign, the preferences of each advertising group, the rivals' bids and tactics, the publishers' reserve pricing, and the number of networks [29]. Therefore, optimal RTB is a mixed-strategy game. There is no pure strategy to bid. With the fast development of reinforcement learning, [30] has tried the model-free method and shown excellent performance. Alibaba Group also proposes a multi-agent reinforcement learning [31]. In the model-free methods, advertisers are considered agents and the state's budget consumption rate, etc. Each advertiser can design its reward function to maximize the profit. Therefore, for a given request, the agent will inference the bid price. After the information is exchanged in the campaign, the agent will get the feedback and calculate the reward, then update the action network and repeat. Based on previous studies, reinforcement learning achieves excellent performance and is distributed on most platforms. However, the model-free approach is closed to the statistical analysis. Fu et al., in [32] proved that unconstrained bidding strategies are hard to learn. We would better focus on even monotone bidding strategies. To solve this problem, we propose a functional optimization reinforcement learning method.

E. Assumptions and Notations

To make the equations in the next section solvable, we have made the following assumptions:

- Assume KPI estimation $c(\boldsymbol{x})$ is determined by features $\boldsymbol{x}$.
- Assume the bidding strategy only depends on KPI estimation $c(\boldsymbol{x})$, like the previous work in [12].
- Assume the winning probability function with respect to bidding price is monotonically increasing and only

TABLE I
SYMBOLS AND DESCRIPTIONS

Symbols	Description
$\boldsymbol{x}$	One bid request represented by its features
$P_{\mathbb{X}}(\boldsymbol{x})$	Probability density function of bid request $\boldsymbol{x}$
$c(\boldsymbol{x})$	The predicted key performance indicator(KPI) for advertisers if winning the bids. It could be CTR or CVR $\boldsymbol{x}$
$P_{\mathbb{C}}(c(\boldsymbol{x}))$	Probability density function of c
B	Total Budget of one campaign
N	Number of requests in one campaign
$b(c(\boldsymbol{x}))$	Bid price function which determines the bidding strategy by taking $\boldsymbol{x}$ and $c(\boldsymbol{x})$ as input. Assume the following dependency: $\boldsymbol{x}\rightarrow c\rightarrow b$
$w(b(c(\boldsymbol{x})))$	Win probability estimation function. Calculate from the dependency assumption: $\boldsymbol{x}\rightarrow c\rightarrow b\rightarrow w$

influenced by the bidding price. Previous works [13], [33], [34] also make the the similar assumption.

above assumptions makes feature $\boldsymbol{x}$ influence the win probability with the following process: $\boldsymbol{x}\rightarrow c\rightarrow b\rightarrow w$.

III. Functional Optimization

In this section, we formulated the RTB problem mathematically [13]. For a given bid request with the high dimensional feature vector $\boldsymbol{x}$, DSP's bidding engine decides to participate in the auction and how much to pay for the current request to win. This paper aims to provide a combined optimization framework that considers user reaction, market competitiveness, and bidding strategy to maximize the advertiser's total profit.

Optimal Bidding Function The optimal bidding function $b_{optimal}()$ is necessary to maximize the estimated impression value per campaign while staying within the campaign's budget B. Therefore, the RTB problem can be formulated as an optimization problem, [13] as follows:

$$\boldsymbol{b}_{optimal} = \underset{b(c(\boldsymbol{x}))}{\arg\max}\ N\int_{\boldsymbol{x}} c(\boldsymbol{x})w(b(c(\boldsymbol{x})))P_{\mathbb{X}}(\boldsymbol{x})\boldsymbol{dx} \quad (1)$$
$$\text{s.t. } N\int_{\boldsymbol{x}} b(c(\boldsymbol{x}))w(b(c(\boldsymbol{x})))P_{\mathbb{X}}(\boldsymbol{x})\boldsymbol{dx} \leq B$$

The product of the KPI $c(\boldsymbol{x})$ and winning probability $w(b(c(\boldsymbol{x})))$, gives the expected KPI per impression auction. $P_{\mathbb{C}}(c(\boldsymbol{x}))$ is the prior probability distribution of feature vectors $\boldsymbol{x}$. Now, the objective is to marginalize winning probability and KPI over feature space and then multiply by the total number of requests to yield the expected impression value per campaign.

According to Zhang, et al [13] and our assumptions, the above optimization problem can be expressed with respect to KPI prediction $c(\boldsymbol{x})$ or c:

$$\boldsymbol{b}_{optimal} = \underset{b(c)}{\arg\max}\ N\int_{\boldsymbol{c}} cw(b(c)P_{\mathbb{C}}(c)\boldsymbol{dc} \quad (2)$$
$$\text{s.t. } N\int_{\boldsymbol{c}} b(c)w(b(c))P_{\mathbb{C}}(c)\boldsymbol{dc} \leq B$$

the Lagrangian function of equation (2) is

$$\mathcal{L}(b(c),\lambda) = \int_c cw(b(c))P_{\mathbb{C}}(c)\boldsymbol{dc} - \lambda\int_c b(c)w(b(c))P_{\mathbb{C}}(c)\boldsymbol{dc} + \frac{\lambda B}{N} \quad (3)$$

where λ is the Lagrange multiplier. After solving the Euler-Lagrange function (3),

$$cP_{\mathbb{C}}(c)\frac{\partial w(b(c))}{\partial b(c)} - \lambda P_{\mathbb{C}}(c)\left[w(b(c)) + b(c)\frac{\partial w(b(c))}{\partial b(c)}\right] = 0 \quad (4)$$

$$\lambda w(b(c)) = [c - \lambda b(c)]\frac{\partial w(b(c))}{\partial b(c)} \quad (5)$$

As proposed by Zhang et. al [13], the winning probability function, which is monotonically increasing and deemed to have a concave shape is assumed to have the form: $w(b(c)) = \frac{b(c)}{a+b(c)}$. However, only one parameter a gives agents limited freedom to learn the λ from the states or take λ as a reward for evaluating its bidding quality. The agents will have low expressiveness and performance. To keep the concave shape of the winning probability function while providing sufficient parameters and making the function is easily differentiable. Instead, we assume the winning probability function has the standard cubic form:

$$w(b(c))) = m_1b^3 + m_2b^2 + m_3b + m_4 \quad (6)$$

Take the partial derivative of $w(b(c)))$ with respect to b and substitute into equation (5)

$$\lambda(m_1b^3 + m_2b^2 + m_3b + m_4) = (c - \lambda b)(3m_1b^2 + 2m_2b + m_3) \quad (7)$$

where b is the bidding price determined by the estimated KPI c. Rearranging the above equation gives us the Lagrange multiplier

$$\lambda = \frac{c(3m_1b^2 + 2m_2b + m_3)}{4m_1b^3 + 3m_2b^2 + 2m_3b + m_4} \quad (8)$$

IV. Multi-agent Reinforcement Learning Architecture

To find the optimal solution of the bidding function, we designed a bidding strategy by combining a model-free reinforcement learning model with resultant λ as a tuning parameter for the bidding function. Specifically, we utilized the Deep-Q learning model to determine the optimum action for each state. Fig 2 shows the proposed multi-agent reinforcement learning (MARL) architecture. The following are the steps of the proposed model in more detail:

A. Environment Setup

Due to the stochastic bidding environment of each campaign, it is difficult to evaluate the performance of advertisers and verify the efficacy of the operating method in the real bidding market. As a result, we depict the market with a virtual environment.

We designed four agents in the bidding environment: three

functional optimization agents (FOAs) and one baseline agent. The winning agent will have two information: its bid price and the second-highest bid price from the market. The other agents can only know they are losing this request with their bidding prices.

B. Agents Setup

The functional optimization agents are designed as given in Algorithm 1, namely, the Lagrange multiplier as shown in equation (8). For functional optimization agents, the first agent consider the Lagrange multiplier as a component of state, which is updated over each request. The second agent consider Lagrange multiplier as its action, aiming to learn the potential best Lagrange multiplier during the bidding so that its actions will maximize the objective function (2). The third agent treats the Lagrange multiplier as one component of its reward function, reflecting the previous bidding quality. The winner has to pay the second-highest bid price for current request.

However, when considering the time complexity, it is impractical to numerically fit the cubic winning probability curve (6) after per action for **FOA II** and **IV**. Additionally, the pre-experiments also demonstrate the fitted continuous curves sometimes are not monotonically increasing. To address these problems, we discretize the equation (5) by substituting a partial derivative $\frac{\partial w(b(c))}{\partial b(c)}$ with

$$\frac{\Delta w(b)}{h},\ where\ \Delta w(b) = w(b_i) - w(b_{i-1})$$

b_i and b_{i-1} are the sequential bid price in the current bid set and h is their difference. So for **FOA III** which aims to calculate bidding price based on action λ, we have

$$b = \frac{c\Delta w(b) - h\lambda w(b)}{\lambda \Delta w(b)} \tag{9}$$

and for **FOA II** and **IV** which take λ as reward, the equation (5) becomes:

$$\lambda = \frac{c\Delta w(b)}{hw(b) + b\Delta w(b)} \tag{10}$$

where $w(b)$, can be biased or unbiased, is discrete probability distribution learned simultaneously over the campaigns.

C. Deep Q-learning

Deep Q-learning uses neural network which maps input states to (action, Q-value) pairs. The proposed DQN setup is as follows:

- $State :=$ [Remaining Budget,Budget Consumption Rate, Win Rate, Step].
  1) R_B denotes remaining budget
  2) The budget consumption rate, $\frac{\sum win\ bid}{200}$, is updated after per bids. (Budget consumption rate is calculated in a deque of which the capacity is 200)
  3) Win rate represents the overall win rate in one campaign
  4) Step represents the progress of one campaign. The maximum step for each agent is the number of total bids N.
- *Action*. Each agent will choose an bid price as the output except the *FOA III*. The action of *FOA III* is to leverage λ. The bid price of this agent will be numerically approximated from equation (9).
- *Reward*. If the agent wins this request, it will be rewarded as 5. Otherwise, it will be rewarded as -1. If the agent wins most at the end of each campaign, it will be rewarded as 200. For the **FOA IV**, the reward function will contain an additional λ which is directly obtained from equation (8).
- *Action-value* $f : Q(s, a)$. We utilize the Deep Q-Network as each agent's action-value function, which has three hidden layers with 128 units for each. After every request, the information will be sent to the experience replay buffer. The capacity of the replay buffer is 1000. After every 20 bids, we will randomly sample 20 bidding information to learn. To train this network, firstly, we will calculate the current expected value for the action.

$$Q^*(s,a) = \mathbb{E}_{s'\sim\mathcal{E}}\left[r + \gamma \max_{a'} Q^*(s', a') \mid s, a\right] \tag{11}$$

where γ is the importance of future, s' and a' represent the next state and action, r indicates the reward. So the loss function becomes

$$L_j(w_j) = \mathbb{E}_{s,a\sim\rho(\cdot)}\left[(y_j - Q(s,a;w_j))^2\right],\ where$$

$$y_j = \begin{cases} r_j \ \text{ for terminal } \mathrm{s}_{j+1} \\ r_j + \gamma \max_{a'} Q(\mathrm{s}_{j+1}, \mathrm{a}'; w) \ \text{ for non-terminal } \mathrm{s}_{j+1} \end{cases} \tag{12}$$

w represents the weight of the Q-Network. To update the network, we have to find the gradient of this loss function.

$$\nabla_{w_j} L_j(w_j) = \mathbb{E}_{s,a\sim\rho(\cdot);s'\sim\mathcal{E}}\Big[(r + \gamma \max_{a'} Q(s', a'; w_{j-1}) - Q(s,a;w_j))\nabla_{w_j} Q(s,a;w_j)\Big] \tag{13}$$

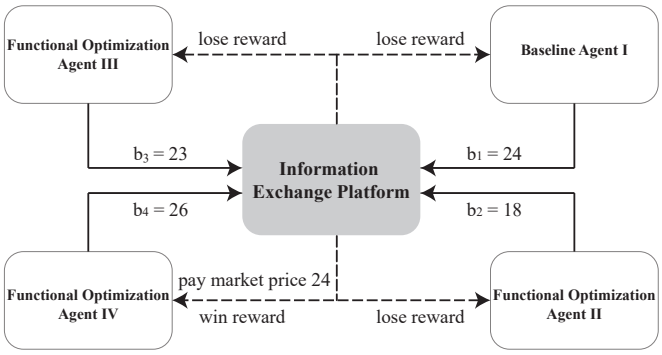


Fig. 2. The MARL Architecture for Second Price Auction. For each request in the campaign, agents will give an optimal bid price based on their strategy. The highest bid price wins the impression on the information exchange platform. Finally, the agents get the feedback from the results and do the strategy update.

Algorithm 1 Baseline and Functional Optimization Agents

N_c := number of campaigns
Initialize experience replay D with capacity $\frac{N}{5}$
Initialize action-value Q-network
Input Operation research function, Bidding environment
Output Q-network $\boldsymbol{Q}(s, a, w)$
for $i = 1 : N_c$ **do**
  Agents setup
  for $j = 1 : N$ **do**
    Select random action a_j with probability ϵ
    Otherwise select action $a_j = \underset{a}{max} Q(s_j, a; w)$
    if Baseline I then
      $b := a_j$
      store bidding results into varuable *result*
      calculate the reward function r
    else if FOA II: then
      $b := a_j$
      store bidding results into variable $result$
      update $w(b)$ using $result$
      update λ from equation (10) using $c, w(b), h$
      calculate the reward function r
      update state using λ
    else if FOA III: then
      $\lambda := a_j$
      solve the equation (9) to get the bid price b
      store bidding results into variable *result*
      update $w(b)$ using $result$
      calculate the reward function r
    else if FOA IV: then
      $b := a_j$
      store bidding results into variable $result$
      update $w(b)$ using $result$
      update λ from equation (10) using $c, w(b), h$
      calculate the reward function r using λ
    end if
    Store transition(s_j, a_j, r, s_{j+1}) in D
    Set state $s_{j+1} = s_j$
    Sample random minibatch from D
    Set $y_j = \begin{cases} r_j & \text{terminal} \\ r_j + \gamma \max_{a'} Q(\mathrm{s}_{j+1}, \mathrm{a}'; w) & \text{non-terminal} \end{cases}$
    Perform a gradient descent $\left(y_j - Q\left(s_j, a_j; w\right)\right)^2$
  end for
end for

V. Experiments and Results

A. Experiments Setup

To demonstrate the efficiency of the proposed framework, we simulate ten campaigns (second-price autions) sequentially ($N_c = 10$) and assume that agents bid together.

As discussed in Section II, both biased and unbiased winning probability approaches are used and two sets of experiments are conducted separately. In addition, we set diverse budgets to imitate the real-world situation of budget shortage and abundance to test the durability of the FOAs in varied settings. For campaigns with biased winning probability, we set the number of bid requests $N_{biased} = 50K$ and assign budget $B_{baised} = \{250K, 500K, 750K, 100K, 1250K\}$ in dollar for each campaigns. While for campaigns utilizing unbiased winning probability, due to the time complexity in camparing global prices and computing unbiased probability, we set $N_{unbiased} = 1K$ and $B_{unbiased} = \{5K, 10K, 15K, 20K, 25K\}$ so that

$$\frac{B_{biased}}{N_{biased}} = \frac{B_{unbiased}}{N_{unbiased}}$$

The bidding price interval is discretized from \$10 to the cap price \$100, where the minimum difference is \$1. All experiments are performed on a Inter® i9-10850K 10-core CPU with 16 GB RAM.

TABLE II
Average Win Rate and Surplus Over 10 Campaigns with Biased Winning Probability

Budget	BA I	FOA II	FOA III	FOA IV
\$250K	12.90% \$23.59	12.78% \$22.51	**62.10%** **\$48.19**	12.22% \$22.58
\$500K	17.65% \$23.07	18.08% \$22.21	**46.73%** **\$46.68**	17.54% \$21.41
\$750K	24.73% \$22.67	**26.59%** \$24.12	23.53% **\$81.19**	25.15% \$21.45
\$1000K	30.33% \$23.76	30.47% \$22.40	8.71% **\$33.55**	**30.50%** \$22.51
\$1250K	**34.72%** **\$23.75**	31.97% \$21.82	6.39% \$17.64	26.91% \$20.59

TABLE III
Average Win Rate and Surplus Over 10 Campaigns with Unbiased Winning Probability

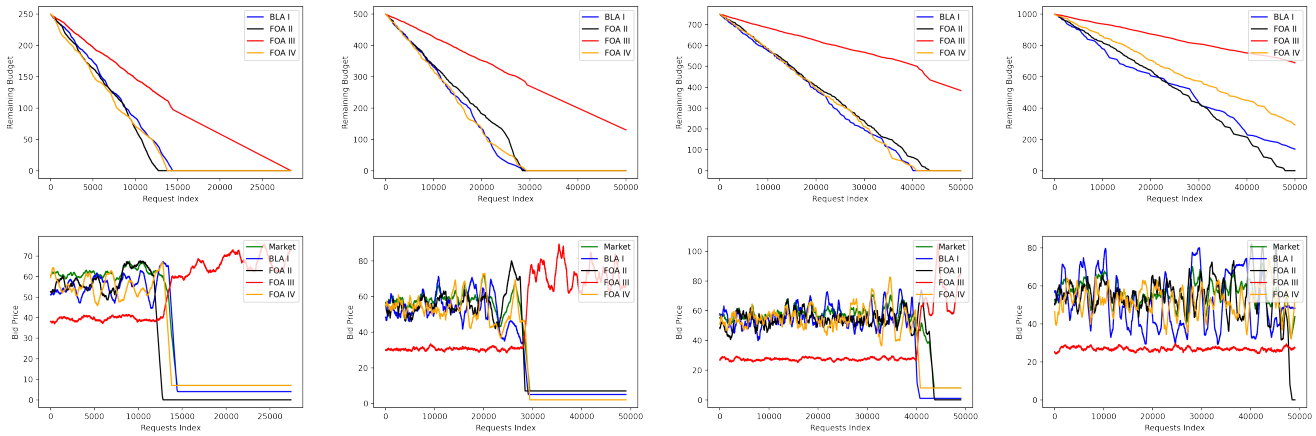
Budget	BA I	FOA II	FOA III	FOA IV
\$5K	25.37% \$22.16	20.26% \$20.05	22.90% **\$30.91**	**31.47%** \$24.54
\$10K	26.84% \$23.05	25.15% \$25.07	13.78% \$26.96	**34.23%** **\$30.13**
\$15K	22.36% \$18.24	27.44% \$22.31	12.32% \$26.85	**37.88%** **\$31.68**
\$20K	**31.27%** **\$25.54**	30.94% \$16.98	10.11% \$22.24	27.68% \$20.00
\$25K	**32.87%** \$21.21	29.32% **\$23.47**	8.95% \$17.73	28.86% \$21.46

B. Parameter Setup

We set ϵ taking about 10K requests to decay from 1 to 0.01. γ = 0.95, batch size = 20. For equation (9) (10), $h = 1$ for biased winning probability where h is dynamic for unbiased probability. For CTR $c = 0.001$ which is a constant after checking the real bidding data.

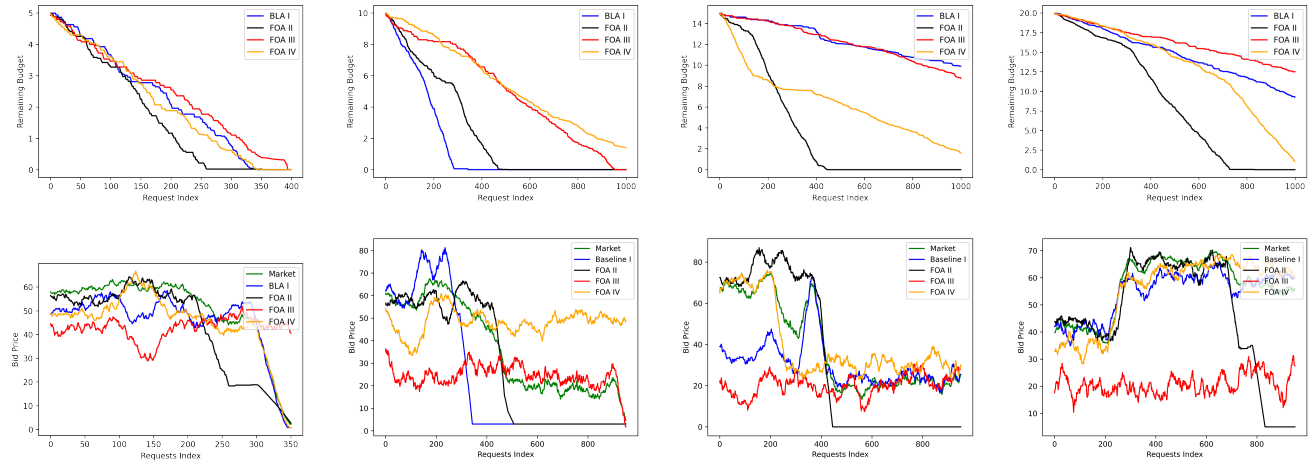
C. Results

The graphical comparison results are shown in Fig 3 and Fig 4, with the first row displaying the budget consumption and the second row displaying the distribution of bid price and winning. In the second row, we use moving average windows to smooth the high-frequency data. Fig 3 results shows that the decline rate of remaining budget for **FOA III** is significantly greater than others over different budgets. Consequently, for biased winning probability when budget is limited (under \$750K in our experiment), FOA III is crafty and learns to bid less at the begining until other agents have



(a) Budget=250K (b) Budget=500K (c) Budget=750K (d)Budget=1000K

Fig. 3. Biased algorithm comparison results: Budget Consumption (first row) and Distribution of the bid price and winning (second row).



(a) Budget=5K (b) Budget=10K (c) Budget=15K (d) Budget=20K

Fig. 4. Unbiased algorithm comparison results: Budget Consumption (first row) and Distribution of the bid price and winning (second row).

exhausted their budgets and then get plenty of opportunities to win. The bidding price subfigures in Fig 3 also illustrate the bidding strategy learned by FOA III.

Table II and III indicate the average Win Rate and Win Surplus Over for 10 campaigns for biased and unbiased Winning Probability respectively with different budgets. Win surplus indicates the difference between the first price(winning price) and the second price(market price). The higher win surplus demonstrates the agent can help users save more money while guaranteeing the chance of winning.

According to Table II, for diverse budgets 250K, 500K, 750K, the FOA III gets the average win surplus of $48.19\$, 46.68\$, 81.19\$$ respectively. Function agents II and IV perform better than baseline agent I by capturing more data from the whole campaign. Therefore, the knowledge of the prior biased distribution is good guidance for the agent to bid. However, any strategy will become useless when each agent has adequate budget(more than \$1000K in our case). Becasue the constraint of equation (2) is no longer strict and agents have more freedom to bid ,so that the impact of Lagrange multiplier λ in helping agents make optimal decisions becomes slighter. As a result, the baseline agent wins at 34.72% at the budget 1250K.

Table III shows the superiority of **FOA IV** with unbiased winning probability. Still, for limited budget (\$5K-\$15K), FOA IV which considers λ as reward has the robust performance with the highest win rate and win surplus among the four. Due to equation (10), the unbiased probability allows the agent to get feedback from the whole market and serve as its reward to make optimal decisions from a global perspective. The second row of Fig 4 also intutively shows the bidding price of FOA IV is generally higher than baseline agent I's. However, as the budget increases, the same problems arise as biased campaigns. Table III shows for budget 25K, the baseline agent has slightly greater performance than FOA II and FOA IV and is significantly more powerful than FOA III.

VI. Conclusion

In this paper, we design the multi-agent reinforcement learning algorithm combined with functional optimization. We created Lagrange multiplier-based agents to take use of functional optimization's capabilities. The results on simulated campaigns demonstrate the effectiveness of functional agents in RTB. In case of limited budget, some functional agents perform better than the baseline agent and have robust performance when the budget varies. The functional agent (as action) bids towards winning data in the situation of a biased winning probability model. When the unbiased winning probability model is used, the functional agent (as reward) takes the campaign's global information and uses the unbiased reward to make the best decisions.

In future work, we can implement the proposed framework to real data to test the efficiency of the use of functional optimization with reinforcement learning. We can also try to develop more strategies which can be stable for the performance of the functional optimization agents on a different pattern of campaigns.

References

[1] J. Xu and X. Gao, "E-payment systems, e-marketing, and e-advertising," *E-business In The 21st Century: Essential Topics And Studies*, vol. 7, p. 125, 2021.

[2] S. Muthukrishnan, "Ad exchanges: Research issues," in *International workshop on internet and network economics*, pp. 1–12, Springer, 2009.

[3] J. Wang, W. Zhang, and S. Yuan, "Display advertising with real-time bidding (rtb) and behavioural targeting," *arXiv preprint arXiv:1610.03013*, 2016.

[4] J. Xu, K.-c. Lee, W. Li, H. Qi, and Q. Lu, "Smart pacing for effective online ad campaign optimization," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2217–2226, 2015.

[5] K.-C. Lee, A. Jalali, and A. Dasdan, "Real time bid optimization with smooth budget delivery in online advertising," in *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising*, pp. 1–9, 2013.

[6] D. Agarwal, S. Ghosh, K. Wei, and S. You, "Budget pacing for targeted online advertisements at linkedin," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1613–1619, 2014.

[7] R. W. Rosenthal, "A class of games possessing pure-strategy nash equilibria," *International Journal of Game Theory*, vol. 2, no. 1, pp. 65–67, 1973.

[8] E. Maskin, "Nash equilibrium and welfare optimality," *The Review of Economic Studies*, vol. 66, no. 1, pp. 23–38, 1999.

[9] V. P. Crawford, "Learning behavior and mixed-strategy nash equilibria," *Journal of Economic Behavior & Organization*, vol. 6, no. 1, pp. 69–78, 1985.

[10] W. Zhang, *Optimal real-time bidding for display advertising*. PhD thesis, UCL (University College London), 2016.

[11] H. Cai, K. Ren, W. Zhang, K. Malialis, J. Wang, Y. Yu, and D. Guo, "Real-time bidding by reinforcement learning in display advertising," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 661–670, 2017.

[12] C. Perlich, B. Dalessandro, R. Hook, O. Stitelman, T. Raeder, and F. Provost, "Bid optimizing and inventory scoring in targeted online advertising," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 804–812, 2012.

[13] W. Zhang, S. Yuan, and J. Wang, "Optimal real-time bidding for display advertising," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1077–1086, 2014.

[14] H. Zhu, J. Jin, C. Tan, F. Pan, Y. Zeng, H. Li, and K. Gai, "Optimized cost per click in taobao display advertising," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2191–2200, 2017.

[15] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "Deepfm: a factorization-machine based neural network for ctr prediction," *arXiv preprint arXiv:1703.04247*, 2017.

[16] Y. Yu, Z. Wang, and B. Yuan, "An input-aware factorization machine for sparse prediction.," in *IJCAI*, pp. 1466–1472, 2019.

[17] H. Pande, "Field-embedded factorization machines for click-through rate prediction," *arXiv preprint arXiv:2009.09931*, 2020.

[18] J. Pan, J. Xu, A. L. Ruiz, W. Zhao, S. Pan, Y. Sun, and Q. Lu, "Field-weighted factorization machines for click-through rate prediction in display advertising," *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*, 2018.

[19] L. Shi and B. Li, "Predict the click-through rate and average cost per click for keywords using machine learning methodologies," in *Proceedings of the International Conference on Industrial Engineering and Operations ManagementDetroit, Michigan, USA*, 2016.

[20] T. Cakmak, A. Tekin, C. Senel, T. Coban, Z. E. Uran, and C. O. Sakar, "Accurate prediction of advertisement clicks based on impression and click-through rate using extreme gradient boosting.," in *ICPRAM*, pp. 621–629, 2019.

[21] S. Rendle, "Factorization machines," in *2010 IEEE International conference on data mining*, pp. 995–1000, IEEE, 2010.

[22] W. Zhang, T. Zhou, J. Wang, and J. Xu, "Bid-aware gradient descent for unbiased learning with censored data in display advertising," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 665–674, 2016.

[23] X. Yang, Y. Li, H. Wang, D. Wu, Q. Tan, J. Xu, and K. Gai, "Bid optimization by multivariable control in display advertising," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1966–1974, 2019.

[24] T. Maehara, A. Narita, J. Baba, and T. Kawabata, "Optimal bidding strategy for brand advertising.," in *IJCAI*, pp. 424–432, 2018.

[25] K. Ren, W. Zhang, K. Chang, Y. Rong, Y. Yu, and J. Wang, "Bidding machine: Learning to bid for directly optimizing profits in display advertising," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 4, pp. 645–659, 2017.

[26] P. Grigas, A. Lobos, Z. Wen, and K.-c. Lee, "Profit maximization for online advertising demand-side platforms," in *Proceedings of the ADKDD'17*, pp. 1–7, 2017.

[27] Y. Chen, P. Berkhin, B. Anderson, and N. R. Devanur, "Real-time bidding algorithms for performance-based display ad allocation," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1307–1315, 2011.

[28] N. Karlsson and Q. Sang, "Adaptive bid shading optimization of first-price ad inventory," in *2021 American Control Conference (ACC)*, pp. 4983–4990, IEEE, 2021.

[29] 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*, pp. 418–423, 2014.

[30] D. Wu, X. Chen, X. Yang, H. Wang, Q. Tan, X. Zhang, J. Xu, and K. Gai, "Budget constrained bidding by model-free reinforcement learning in display advertising," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 1443–1451, 2018.

[31] 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*, pp. 2193–2201, 2018.

[32] H. Fu and T. Lin, "Learning utilities and equilibria in non-truthful auctions," *arXiv preprint arXiv:2007.01722*, 2020.

[33] W. Zhang, Y. Zhang, B. Gao, Y. Yu, X. Yuan, and T.-Y. Liu, "Joint optimization of bid and budget allocation in sponsored search," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 08 2012.

[34] B. Kitts and B. LeBlanc, "Optimal bidding on keyword auctions," *Electronic Markets*, vol. 14, pp. 186–201, 09 2004.