Adapting Constrained Markov Decision Process for OCPC Bidding with Delayed Conversions

LEPING ZHANG and XIAO ZHANG, Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
YICHAO WANG, Huawei Noah's Ark Lab, Shenzhen, China
XUAN LI, Huawei AppGallery Development Department, Nanjing, China
ZHENHUA DONG, Huawei Noah's Ark Lab, Shenzhen, China
JUN XU, Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China

Nowadays, optimized cost-per-click (OCPC) has been widely adopted in online advertising. In OCPC, the advertiser sets an expected cost-per-conversion and pays per click, while the platform automatically adjusts the bid on each click to meet advertiser's constraint. Existing bidding methods are based on feedback control, adjusting bids to keep the current cost-per-conversion close to the expected cost-per-conversion to avoid compensation. However, they overlook the conversion lag phenomenon: There always exists a time interval between the ad's click time and conversion time. This interval makes existing methods overestimate the cost-per-conversion and results in over conservative bidding policies which finally hurts the revenue. To address the issue, this article proposes a novel bidding method, Bidding with Delayed Conversions (Bid-DC) which predicts the conversion probability of the clicked ads and used it to adjust the cost-per-conversion values. To ensure the bidding model can satisfy the advertiser's constraint, constrained Markov decision process (CMDP) is adapted to automatically learn the optimal parameters from the log data. Both online and offline experiments demonstrate that Bid-DC outperforms the state-of-the-art baselines in terms of improving revenue. Empirical analysis also showed Bid-DC can accurately estimate the cost-per-conversion and make more stable bids.

CCS Concepts: • **Information systems** → **Computational advertising**;

Additional Key Words and Phrases: Online advertising, Display advertising, Delayed user feedback

This work was funded by the National Key R&D Program of China (2023YFA1008704), the National Natural Science Foundation of China (No. 62376275), Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China, fund for building world-class universities (disciplines) of Renmin University of China. Work partially done at Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education.

Authors' Contact Information: Leping Zhang, Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China; e-mail: lepingzhang@ruc.edu.cn; Xiao Zhang, Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China; e-mail: zhangx89@ruc.edu.cn; Yichao Wang, Huawei Noah's Ark Lab, Shenzhen, China; e-mail: wangyichao5@huawei.com; Xuan Li, Huawei AppGallery Development Department, Nanjing, China; e-mail: lixuan13@huawei.com; Zhenhua Dong, Huawei Noah's Ark Lab, Shenzhen, China; e-mail: dongzhenhua@huawei.com; Jun Xu (corresponding author), Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China; e-mail: junxu@ruc.edu.cn.

ACM 1558-2868/2025/1-ART47
https://doi.org/10.1145/3706420

ACM Reference format:
Leping Zhang, Xiao Zhang, Yichao Wang, Xuan Li, Zhenhua Dong, and Jun Xu. 2025. Adapting Constrained Markov Decision Process for OCPC Bidding with Delayed Conversions. *ACM Trans. Inf. Syst.* 43, 2, Article 47 (January 2025), 29 pages.
https://doi.org/10.1145/3706420

1 Introduction

Nowadays, **optimized cost-per-click (OCPC)** has become one of the major pricing methods in online advertising. Studies have shown that OCPC can achieve a precise matching between the bid and the traffic quality of page view requests, and therefore improve advertising efficiency [30, 32, 88]. Different from the traditional pricing methods such as **cost-per-click (CPC)** [38, 47] and **cost-per-mille (CPM)** [69, 71], OCPC requires the advertisers first to set an expected cost-per-conversion and a constraint (e.g., ensuring that the cost-per-conversion[1] stays within a specific range, e.g., $\leq$ 120%× the expected cost-per-conversion), and then pay per click. The platform is responsible to automatically bid on each click based on the user's clicks and conversions to meet the advertiser's constraint. Figure 1 shows the workflow of OCPC.[2] The OCPC process can be divided into three periods, the preparing period, the advertising period and the end of advertising. In the preparation period, the platform will prepare models and parameters for advertising period. During the advertising period, the platform will display ads and adjust bids accordingly. Finally, at the end of advertising, the platform checks whether to compensate to the advertiser based on the final cost-per-conversion and the expected cost-per-conversion. The advertising period has two processes: the online bidding process and the feedback updating process. Those processes can be described as follows (the following numbers correspond to the numbers in Figure 1).

(1) During the preparing period, the advertiser negotiates with the platform to set the expected cost-per-conversion (i.e., how much the advertiser expects to pay for a conversion.), the advertising period (i.e., the start and end time of the ad campaign) and the conversion goal (i.e., which user's action will be considered as a conversion, e.g., user registration or purchase).
(2) The platform will use the log data to train the CTR/CVR model [55, 56] and the bidding model [3] offline. The CTR/CVR model acts as a transformer, converting the bidding model's bids for conversions into bids for impressions by multiplying by predicted Pctr and Pcvr and the bidding model will adjust its bids for conversions based on the current cost-per-conversion and the expected cost-per-conversion to maximize the platform's revenue while meet the advertiser's constraint.
(3) When a user enters the app, the online bidding process starts.
(4) The platform will use the bidding model (decide the bid on the conversions) as well as the CTR/CVR model (transform the bid on conversions to the bid on impressions) to form the bids.
(5) All ads will be ranked based on their bids, then present them to the user (i.e., Ads with higher bids will display in better positions).
(6) If the user clicks an ad, the platform will charge the corresponding advertiser by its bid.
(7) If the user converts, the platform will record the conversions.

[1] The total costs of the advertiser divided by the number of conventions.
[2] The strategy has been commonly adopted in real Ads platforms. For example: https://developer.Huawei.com/consumer/en/doc/promotion/ocpc-0000001310577317

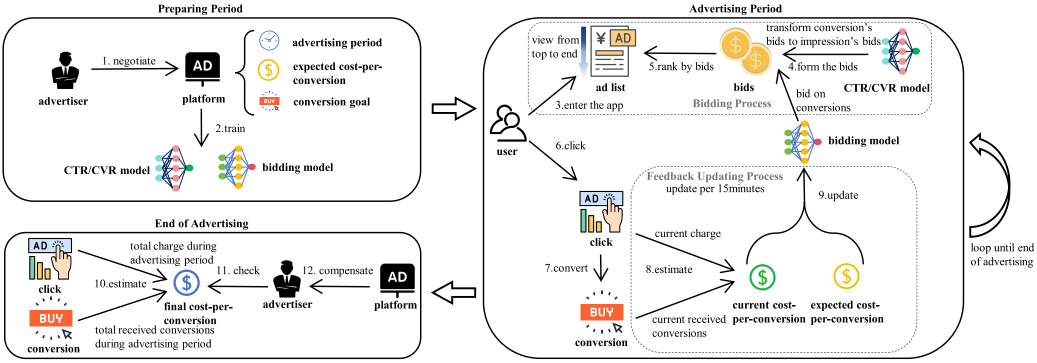


Fig. 1. The workflow of OCPC.

(8) For every 15 minutes, the feedback updating process starts. The platform will use the current total charges divides the current total received conversion numbers to estimate the current cost-per-conversion (i.e., how much does it currently cost the advertiser for a conversion?).
(9) Then the current cost-per-conversion and the expected cost-per-conversion (also with other factors, like time) will be used to update the state of the bidding model.
(10) Finally, at the end of the advertising period, the advertiser will calculate the final cost-per-conversion based on the total charge the total received conversions.
(11) The platform will check whether it breaks the constraint based on the final cost-per-conversion.
(12) If the platform breaks it, the platform will compensate to the advertiser.

Bidding (process 4 of Figure 1) is one of the most important tasks within the advertising period for both the advertisers and the platform. As for the advertisers, they want to improve their **Return on investment (ROI)** [82, 83]. As for the platform, it aims to maximize its revenue while avoiding breaking advertiser's constraints [59, 81]. This article stands for the platform's perspective, focusing on maximizing revenue while avoiding breaking advertiser's constraints. Bidding is not a trivial task. On one hand, if the platform bids too high, it will result in excessive charges to the advertisers and even violate the constraint, leading to the platform compensation. On the other hand, if the platform bids too low, it will decrease the platform's income on each click, lower the ranking and exposure possibility [11], and eventually decrease the likelihood of being clicked and converted. An ideal bidding policy should not only maximize the platform revenue but also ensure the advertisers' constraint [50, 51, 74].

To avoid compensation, most existing bidding methods are developed based on the idea of feedback control: bid high if the current cost-per-conversion is lower than the expected cost-per-conversion, while bid low if the current cost-per-conversion is higher than the expected cost-per-conversion. For example, the widely adopted **proportional-integral-derivative (PID)** controller [6] and the **waterlevel (WL)**-based controller [19] make use of the difference between the current cost-per-conversion and the expected cost-per-conversion to adjust the bid. At each time step, to ensure the advertiser's constraint, the feedback controller tries to bring the cost-per-conversion closer to the expected cost-per-conversion, by calculating the deviation between them. **Reinforcement learning (RL)** has also been utilized for learning the bidding models. For example, RSDRL [59] sets a fixed threshold on the ROI. When the current advertiser's ROI exceeds the threshold, the model will lower the bid (i.e., by forcing the model bids twice and taking the lower one) to prevent the bid from exceeding the advertiser's constraint. See also [68, 83].

Though effective and have been widely adopted in OCPC platforms, existing methods still have much room to improve because they overlook an important phenomenon in OCPC: delayed conversions. In OCPC, after an ad being clicked by a user, the platform immediately charges the advertiser for the click. Then after clicking, the user has two choices: not convert (e.g., abandon after the click) or convert. The platforms need to record the conversion signals to estimate the cost-per-conversion. However, the conversion signals are inevitably delayed: the platform needs to wait a period of time (often not very short) to get a success conversion signal, or waits until the end of advertising period expires to know that it is not converted.

Existing methods assume that an ad click is not converted if no success conversion signal was received (i.e., in process 8 of Figure 1, platform only considers the received conversions), which inevitably results in overestimated cost-per-conversion values. The phenomenon is easy to explain: suppose that at time t, the advertiser has been charged based on the user's clicks and bids. Some of the payed clicks have been converted while others not. The cost-per-conversion at time t, therefore, is estimated as the total charges before t divided by the number of conversions received until t. Please note that those clicks occurred before t while converted after t (delayed conversions) are considered as not converted, making the estimated cost-per-conversion larger than its real value. To make the cost-per-conversion satisfy the constraint and avoid compensation, the bidding algorithms have to decrease the bids, which further hurts the ranking and exposure of the ad, and eventually decreases the future clicks and conversions.

Figure 4(a) shows the estimated cost-per-conversion curve as well as the bidding curve by PID, for an ad from a real industrial OCPC platform. The curves show that PID controlled the estimated cost-per-conversion around the line of expected conversion value (denoted as "expected cost-per-conversion"). However, if we adjust the estimated cost-per-conversion values by counting in the delayed conversions (i.e., at the time t, the number of conversions is calculated as the sum of received conversions before t, and the delayed conversions that clicked before t but converted after t), we found that the adjusted cost-per-conversion curve (denoted as "true cost-per-conversion" in the figure) is much lower, indicating the problem of overestimating cost-per-conversion in existing methods. The over estimated cost-per-conversion results in lower bid. Moreover, as shown in Figure 1 process 3, the ads will be sorted based on the bids, therefore a lower bid results in a lower ad rank, which will decrease ad's exposure and, consequently, reduce the likelihood of user conversions.[3] Therefore, the delayed conversion issue reduces both the platform's revenue and advertiser's conversion numbers. If the platform knew the future, it could safely increase the bid to boost the delivery of this ad, and then wait for the upcoming conversions to lower the real cost-per-conversion to meet the constraint. The challenge is that the platform cannot know how many delayed conversions will happen in the future in real world.

Moreover, existing bidding methods require many hyper-parameters to be determined which are crucial for their bidding performance. Taking the fixed threshold in RSDRL [59] as an example, this threshold denotes our optimism about the future. A low threshold means we believe there will be few conversions in future, therefore, we will lower the bid to avoid compensation even the current cost-per-conversion is slightly higher than the target and vice versa. Different ads have different conversion goals, as Figure 2 shows, the optimal thresholds[4] for different ads are different. Such phenomenon indicates that to ensure the bidding performance of each ad, we need to manually set the optimal threshold for each ad, which requires a considerable amount of effort.

[3] An ad requires exposure before it can lead to a conversion.

[4] From the perspective of the platforms, the optimal threshold means the threshold that maximizes the platform's revenue while avoiding any compensation.

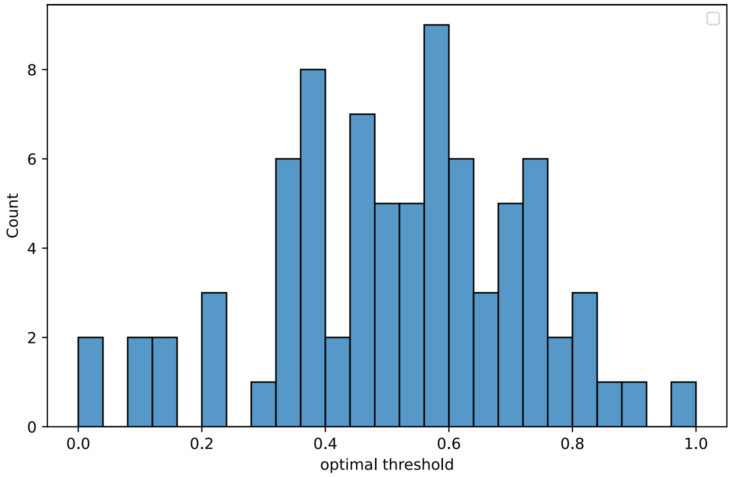


Fig. 2. The distribution of optimal threshold values over 80 advertisers randomly selected in an OCPC platform. The threshold values are normalized for avoiding privacy issues.

To address these issues, in this article, we propose a novel bidding model tailored for OCPC where each bid is estimated not only based on the observed clicks, bids, and conversions, but also *the prediction of the delayed conversions*. To maximize the platform revenue and avoid the risk of violating the advertiser's constraint, **Constrained Markov Decision Process (CMDP)** and constrained RL [13, 14, 23, 66, 73] are adapted to automatically learning the optimal parameters from the data without any human effort. In this way, the bidding model can automatically enhance the revenue while avoid breaking the constraint. Specifically, by formalizing the distribution of the delayed conversion time as an exponential distribution and given the time the click has past, the model first predicts the probability of receiving future conversion of a clicked ad. Based on this, CMDP formulates the bidding process as a sequential decision-making, where the users and platform correspond to the environment and agent, respectively. At each time step, the agent selects an action which corresponds to bid price for the click, and the reward and constraint value are calculated based on the user's feedback. Analysis shows that **Bidding with Delayed Conversions (Bid-DC)** has the ability of enhancing the future chances of ad exposures, increasing the bid stability and preventing high fluctuations, and being adapted for other conversion goals.

The major contributions of the article can be summarized into three aspects:

(1) We analyzed the delayed conversions in OCPC, and found that current bidding methods (e.g., PID) often overestimate the cost-per-conversion values, resulting in lower bids and finally hurting the platform revenue and future exposure of the ads. Moreover, to ensure their performance on each ad, we need to manually set the optimal hyper-parameters for each ad, which requires a considerable amount of effort.
(2) We propose a novel bidding model called Bid-DC to deal with the issue, which makes more accurate bids by taking the prediction of future conversions into consideration, and utilizes the constraint RL framework to automatically learn the optimal parameters from the data without any human effort.
(3) We conduct both online and offline experiments on Bid-DC. The experimental results show that Bid-DC can effectively enhance the platform revenue while still avoiding the compensation. Empirical analysis also showed that Bid-DC improved the bidding stability.

2 Literature Review

This section reviews the literature related to the research topic, the bidding methods in OCPC and the constraint RL.

2.1 OCPC and the Bidding Methods in OCPC

As one of the most important step in Internet advertising, real-time-bidding has attracted much research effort in recent years [12, 78–80]. For example, Yang et al. [74] convert the bidding problem into a **linear programming (LP)** problem and leverage the primal-dual method to derive the optimal bidding strategy. Arapakis et al. [3] propose a novel auction format based on a pay-per-attention scheme which can inherit the desirable properties (strategy-proofness and efficiency). RL has also been adapted to generate bids [46, 70, 86]. Zhao et al. [85] propose a robust Markov decision process model at hour-aggregation level to deal with the stochastic and complex bidding environment. Mou et al. [48] propose a sustainable online RL framework to avoid the problem of inconsistency between online and offline. He et al. [27] unify the constrained bidding formulation and propose an RL method to dynamically adjust parameters to achieve the optimal bid.

To meet the advertisers' diverse needs, the advertising platforms have developed multiple pricing methods, including CPM for brand promotions, CPC and **cost-per-action (CPA)** for immediate increasing profits [4, 49]. Different from these traditional pricing methods where the bids are fixed, OCPC is a type of pricing method where the bid is automatically determined based on the estimated cost-per-conversion [43, 88]. As one of the most important problems in OCPC, bid optimization has attracted a growing body of research efforts [65, 75]. The existing bidding methods tailored for OCPC bidding problem can be categorized into three groups:

- *Feedback control methods* are standard methods in the bid optimization problem. Their core idea is setting the expected cost-per-conversion as the reference and adjusts bids based on the difference between the current cost-per-conversion and the reference, thereby keeping the current cost-per-conversion close to the reference to avoid compensation. For example, PID controller [74] uses proportional, integral and derivative terms which are all linear combinations of the difference to combine the bid, while the WL controller [19] only uses the proportional term. Despite their success, the feedback control methods have several drawbacks. First, they fail to capture the non-linear relationship between the difference and the bid, which is sub-optimal for the platform [8]. Second, as mentioned above, they overlook the delayed conversions which hurt their performance. Third, they need to determine several hyper-parameters, e.g., the proportional term's gain factor, which requires much effort [82].
- *Model based RL bidding methods* can capture non-linear relationships and learn better policies than the linear approaches [8], such as the feedback controller. Their core idea is to model the bidding process as a Markov Decision Process and use RL to train the bidding policy in a simulated environment [27]. For example, RSDRL [59] employs distributional RL methods IQN [17] to learn the optimal policy with a fixed ROI threshold to avoid compensation. Similarly, USCB [27] uses DDPG [45] to learn the policy and introduces a hyper-parameter to control the severity of penalty when policy exceeds the constraint. Although they can capture the non-linear relationships between the difference and the bid, they cannot address the other two disadvantages of the feedback control methods, because they require hyper-parameters to adjust the penalty and also overlook the delay feedback problem.
- *Offline RL bidding methods* train the policy directly from log data, eliminating the effort of building a simulated environment. For example, TEE [42] employs IQL [40], CQL [41] to train the bidding policy from the log data and uses Parameter Space Noise [53] to improve the

enhance data quality. However, they still require the determination of the hyper-parameters and cannot address the overestimation problem.

2.2 Constraint RL

One important characteristic of OCPC is the cost-per-conversion constraint (i.e., at the end of the advertising period, the cost-per-conversion cannot higher than the expected cost-per-conversion), which poses new difficulties in bidding [23]. From the platform's perspective, it aims to maximize its own revenue while also preventing breaking the constraint. Such problem formulation exactly corresponds to the form of constraint RL [26, 35]. Based on the CMDP, the common constraint RL problem can be formulate as:

$$\max_{\pi \in \Pi} J_R(\pi) \quad s.t.\ J_C(\pi) \leq 0,$$

where $J_R(\pi)$ denotes the cumulative reward of policy π (i.e., the total reward that policy π gains within an episode), $J_C(\pi)$ denotes the cumulative constraint cost of policy π (i.e., the number of constraints that policy π breaks within an episode). The aim of constraint RL is searching a policy which maximize the cumulative reward while avoid breaking any constraint. To solve the above equation, Achiam et al. [1] propose **Constrained Policy Optimization (CPO)** which utilizes trust region at each policy update step to enforce the constraints. Jayant and Bhatnagar [33] use Lagrangian relaxation to find an appropriate and optimal policy.

In this article, we consider the revenue as the cumulative reward and whether breaking the advertiser's constraintas the cumulative constraint cost, respectively, then employ constraint RL framework to conduct bids while ensuring the compliance with the constraint.

3 Problem Formulation

This section formulates the problem of platform bidding in OCPC.

The bidding problem in OCPC can be formulated as follows. In the preparing period, the advertiser chooses a deep conversion goal (e.g., user registration or purchase) for ad l and sets the expected cost-per-conversion v^l. Then, the platform negotiates a compensation rate $\beta \in [0, +\infty)$ with the advertiser on ad l, i.e., at the end of the advertising period (e.g., at the end of a day), the platform will compensate the loss if the actual cost-per-conversion is higher than $(1 + \beta) \cdot v^l$.

During the online advertising period, when a user u_t comes at time step t, the platform sets a bid $a_t^l > 0$ on the ad l (process 4 of Figure 1). The ad l as well as other ads are presented to u_t, ranked based on their bids (and other considerations such as estimated CTR/CVR etc, process 5 of Figure 1) [11]. We use $c_t^l \in \{0, 1\}$ to denote whether u_t clicked the ad l. $c_t^l = 1$ if clicked, otherwise 0. The platform charges the advertiser for a_t^l if $c_t^l = 1$. Meanwhile, we use $\alpha_{t,t'}^l \in \{0, 1\}$ to denote whether the ad l clicked at time step t has converted before the time step t' ($t' \geq t$).[5] $\alpha_{t,t'}^l = 1$ means it has converted, otherwise 0. At each time t, if the user clicks or converts on ad l, the platform will update the cost-per-conversion for ad l at time t:

$$cpc_t^l = \frac{\sum_{j=1}^{t} c_j^l \cdot a_j^l}{\sum_{j=1}^{t} \alpha_{j,t}^l}, \tag{1}$$

to adjust the state of the bidding model.

At the end of the advertising period (e.g., at the end of the day), the platform needs to ensure the final cost-per-conversion $cpc_N^l \leq (1 + \beta) \cdot v^l$, where N is the total number of time steps in the whole episode. Otherwise, the platform should compensate the advertiser for the loss.

[5] Here, we consider the conversion follows the post-click attribution in which an impression needs to have been clicked to be considered as having led to a conversion [10].

Table 1. Summary of the Notations

Notation	Description
t	time step t
N	total steps in an episode (e.g., one day)
l	an advertisement l
u_t	the user accessed the system at step t
$v^l \in \Re^+$	advertiser's expected cost-per-conversion on l
$\beta \in \Re^+$	compensation rate
$a_t^l \in \Re^+$	the bid on click of advertisement l exposed at step t
$c_t^l \in \{0, 1\}$	whether u_t clicked ad l which was exposed to her at t
$\alpha_{t,t'}^l \in \{0, 1\}$	observed conversion corresponds to c_t^l at time $t' > t$
$\Delta t \in \Re^+$	delayed time, length between click and conversion
$\alpha_{t,N}^l \in \{0, 1\}$	whether the click c_t^l eventually converted
$\hat{\alpha}_{t,N}^l \in \{0, 1\}$	estimation of the conversion probability for c_t^l
$cpc_t^l \in \Re^+$	cost-per-conversion, estimated on log data before t
$\widetilde{cpc}_t^l \in \Re^+$	true cpc value with the final conversion signals $\alpha_{t,N}^l$'s
$\widehat{cpc}_t^l \in \Re^+$	estimation of $\widetilde{cpc}_t^l$

From the platform's perspective, its goal is to increase the revenue without incurring any compensation. Therefore, the goal on ad l can be formulated as:

$$\max_{\{a_1^l,\ldots,a_N^l\}} \sum_{t=1}^{N} c_t^l \cdot a_t^l \qquad (2)$$
$$s.t. \quad cpc_N^l \le (1+\beta) \times v^l.$$

Existing methods usually formulate the online bidding as a problem of sequential decision-making. That is, at each step t, the bidding model sets the bid a_t^l based on the currently estimated cpc_t^l: decreasing the bid if $cpc_t^l > v^l$, or increasing the bid if $cpc_t^l < v^l$. For example, PID [6] uses the difference $v^l - cpc_t^l$ to adjust the bid, and RSDRL [59] generates a lower bid when $\frac{cpc_t^l}{v^l}$ is larger than the pre-set threshold. Table 1 summarises the notations used in the article.

4 Proposed Method: Bid-DC

This section proposes a novel bidding model to deal with the delayed conversion challenge in OCPC, called Bid-DC. Bid-DC includes two parts: the delay part and the bidding part. The goal of the delay part is to estimate $\widetilde{cpc}_t^l$ and the bidding part is to bid at each bidding process through the estimated $\widehat{cpc}_t^l$. In Bid-DC's bidding part, the bidding process is formulated as a CMDP. The design of the constraints is based on the estimated $\widetilde{cpc}_t^l$. RL is used to learn the optimal model parameters and conduct online bidding.

4.1 Impacts of the Delayed Conversion Signals

One issue with the above methods is that they overlook the important delayed conversion phenomenon in OCPC. That is, after being exposed and clicked the ad l, u_t may take a while (usually not very short) time to return the corresponding conversion signal $\alpha_{t,t}^l$ at some step t'.[6] More

[6] In real world, u_t also need some time to click ad l. In this article, we assume that the user clicks immediately after the exposure for ease of notation. The assumption will not change the conclusions in the article.

Table 2. Click and Conversion Signals of the ad l Exposed at Time t

Observed at $t' > t$: $(c_t^l, \alpha_{t,t'}^l)$	At the end of episode: $(c_t^l, \alpha_{t,N}^l)$
(0, 0)	(0, 0)
(1, 0)	$\begin{cases}(1,1) & \textbf{if converted}\\ (1,0) & \text{if not converted}\end{cases}$
(1, 1)	(1, 1)

Left: signals observed at step $t' > t$ (denoted as $(c_t^l, \alpha_{t,t'}^l)$; Right: true signals at the end of the episode (denoted as $(c_t^l, \alpha_{t,N}^l)$).

specifically, after u_t clicks ad l at t, the platform observes $c_t^l = 1$ and the advertiser is charged a_t^l. Then, they initialize $\alpha_{t,t}^l = 0$ and wait the user conversion. There are two possible results: (1) they wait a period of time and receive the success conversion signal at time t', then set $\alpha_{t,t'}^l = 1$; (2) they wait until the advertising period ends, and $\alpha_{t,t'}^l$ is kept to 0 for any time $t' \geq t$. If u_t did not click ad l, i.e., $c_t^l = 0$, there will be no conversion and $\alpha_{t,t'}^l = 0$ for any t'.

We can observe that, $\alpha_{t,N}^l \in \{0, 1\}$ indicates whether the click at time t is converted eventually before the end of the advertising (i.e., before the time step N). We can consider $\alpha_{t,t'}^l$ as the observed variable at time step $t' \in [t, N]$ before the end of the advertising. Table 2 shows the relationship between $\alpha_{t,N}^l$ and $\alpha_{t,t'}^l$. That is, if we observe a pair $(c_t^l = 1, \alpha_{t,t'}^l = 0)$ at some time step t' $(t < t' < N)$, it is possible to observe the user conversion between t' and N. Therefore, if there is another user comes at t', the platform has to bid based on the $cpc_{t'}^l$ which is calculated by Equation (1) where all the observed $(c_i^l = 1, \alpha_{i,t}^l = 0)$ pairs (for $i = 1, \ldots, t$) are charged and considered as no conversion, though some of them may be converted in the future.

We propose $\widetilde{cpc}_t^l$ to denote the true cost-per-conversion value at time t for ad l, which calculates as

$$\widetilde{cpc}_t^l = \frac{\sum_{j=1}^t c_j^l \cdot a_j^l}{\sum_{j=1}^t \alpha_{j,N}^l}, \tag{3}$$

for all $t = 1, \ldots, N$. Equation (3) also indicates that Equation (1) always overestimates the cost-per-conversion values, i.e., $cpc_t^l \geq \widetilde{cpc}_t^l$ for all $t \leq N$ because $\alpha_{j,N}^l \geq \alpha_{j,t}^l$ for all $j \leq t$. The analysis also confirmed the results reported in Figure 4(a) that the curve of "true cost-per-conversion" ($\widetilde{cpc}_t^l$ w.r.t. t) is lower than "cost-per-conversion" (cpc_t^l w.r.t. t). The overestimated cpc_t^l results in over-conservative bidding strategy (curve "PID bids") and finally hurts the exposure of the ad and future clicks and conversions. We note that in Figure 4(a), curves "true cost-per-conversion" and "cost-per-conversion" equal at time step N, i.e., $cpc_N^l = \widetilde{cpc}_N^l$. This is because the advertising ends until all of the clicked ads convert or timeout. At the time step N, the observed conversion equals to the final conversion $\alpha_{t,N}^l$.

The analysis shows that to achieve more platform revenue, it is necessary to adopt a more aggressive and precise bidding strategy which is guided by $\widetilde{cpc}_t^l$ instead of cpc_t^l. However, calculating $\widetilde{cpc}_t^l$ is not trivial: at each of the bidding time $t < N$, $\widetilde{cpc}_t^l$ cannot be calculated directly because some $\alpha_{i,N}^l$'s $(i < t)$ are un-observable (i.e., those not converted clicks $(c_i^l = 1, \alpha_{i,t}^l = 0)$).

4.2 Bid-DC's Delay Part

We observed that for an ad, the delayed conversion time Δt (the interval between the click time and the real conversion time) roughly obeys the exponential distribution, as shown in Figure 3.

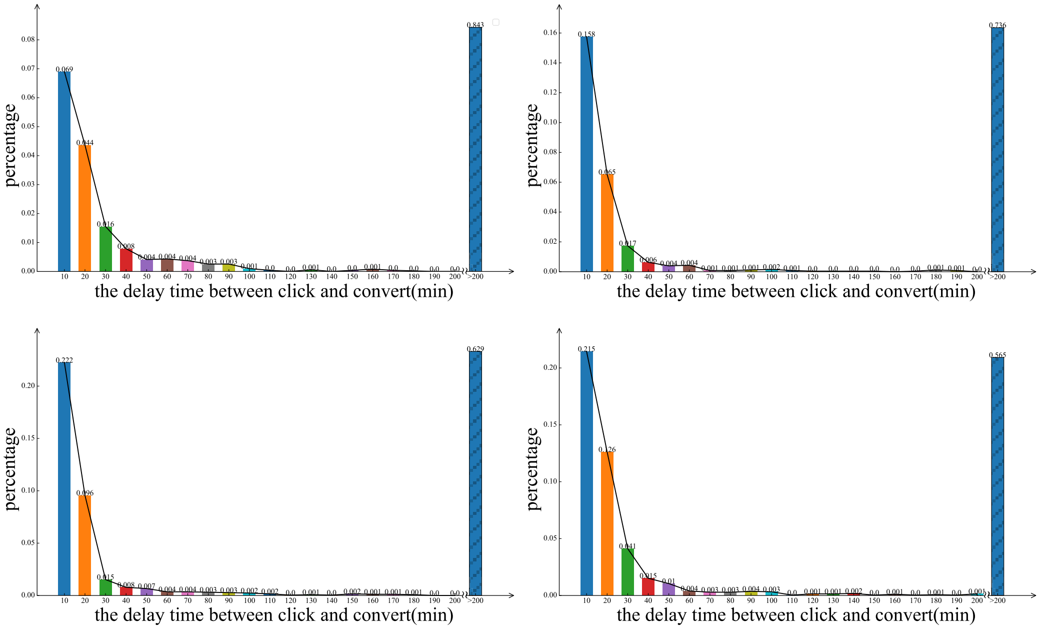


Fig. 3. The delayed time distribution for 4 ads in an industrial dataset.[7] The percentage of the conversions w.r.t. the delay time (in minutes) in the industrial dataset. The delay time is calculated as the interval between the user click time and the corresponding convert time. Note that the last column displays all the conversions whose delayed time > 200 minutes.

This similar phenomenon was also observed in existing studies [10]. Following the practices in [10, 84], Bid-DC's delay part also uses the exponential distribution to formulate the probability that a click's delayed time length is $\Delta t > 0$:

$$f(\Delta t|x) = \lambda(x)e^{-\lambda(x)\Delta t}, \lambda(x) \geq 0, \tag{4}$$

where $\lambda = \lambda(x)$ is a parameter determined by the click's feature x.

Therefore, for a clicked but not converted signal pair ($c_t^l = 1, \alpha_{t,t'}^l = 0$), which observed at time t', and has passed $\Delta t = t' - t$ length, the probability that the click will eventually convert, i.e., denoted as $P(\alpha_{t,N}^l = 1|c_t^l = 1, \alpha_{t,t'}^l = 0, \Delta t)$, is equal to the probability that the ad's delay time is greater than Δt but smaller than $N - t$, which can be calculated as:

$$\begin{aligned} P(\alpha_{t,N}^l = 1|c_t^l = 1, \alpha_{t,t'}^l = 0, \Delta t, x) &= \int_{\Delta t}^{N-t} f(u|x)du \\ &= \int_{\Delta t}^{N-t} \lambda(x)e^{-\lambda(x)u}du \\ &= e^{-\lambda(x)\Delta t} - e^{-\lambda(x)(N-t)}. \end{aligned} \tag{5}$$

[7] Here, the distribution refers to the time distribution between a click and a conversion, including un-converted clicks. Moreover, although the un-converted clicks are far more than the converted ones, their delay times are infinite, i.e., delay time → ∞. Their percentages will be spread across an infinitely long x-axis, making the percentage at each point very small, therefore the distribution is close to an exponential distribution. Existing study [84] also uses the exponential distribution to model the distribution.

Through Equation (5), we can estimate $\widetilde{cpc}_t^l$ as:

$$\widehat{cpc}_t^l = \frac{\sum_{j=1}^{t} c_j^l \cdot a_j^l}{\sum_{j=1}^{t} \hat{\alpha}_{j,N}^l}, \tag{6}$$

where the estimation $\widetilde{\alpha}_{j,N}^l$ is defined as

$$\hat{\alpha}_{i,N}^l = \begin{cases} 1 & \text{if } c_i^l = 1 \wedge \alpha_{i,t}^l = 1; \\ 0 & \text{if } c_i^l = 0 \wedge \alpha_{i,t}^l = 0; \\ P(\alpha_{i,N}^l = 1 | c_i^l = 1, \alpha_{i,t}^l = 0, \Delta t) & \text{if } c_i^l = 1 \wedge \alpha_{i,t}^l = 0. \end{cases} \tag{7}$$

Equation (7) shows, during the estimation of the $\widehat{cpc}_t^l$, if the system observed a clicked while unconverted ad, Bid-DC's delay part replaces the observed signal $\alpha_{i,t}^l = 0$ with the probability that it will be eventually converted (defined in Equation (5)). For other cases, according to the analysis in Table 2, we know that $\hat{\alpha}_{i,N}^l = \alpha_{i,t}^l$ (the observed value).

4.3 Fitting the Parameter in Bid-DC's Delay Part

To learn the optimal $\lambda(\cdot)$ in Equation (4), we adopt the idea of minimizing negative log-likelihood over the empirical data distribution.

Suppose the learning data includes n history clicks, $\{x_i, e_i, d_i, \delta_i\}_{i=1}^n$, where x_i represents the features of the click, e_i indicates the elapsed time since the click, d_i represents the delay time between the click time and the conversion time and δ_i represents whether the click has been converted before we collect the data.

However, due to the delay feedback problem, there also exists a right-censoring problem in the training data. When we collect the history data, some of the click's conversion may not received (i.e., $\delta_i = 0$), for those data, we do not know the exact delay time d_i, we only know the elapsed time e_i and $d_i \geq e_i$ since the conversion has not received after e_i time.

Inspired by the survival analysis technique [5, 67, 87], using D to donate the delay time random variable and the likelihood can be shown as follows:

$$\begin{aligned} \mathcal{L}(\lambda) &= \prod_{i=1}^{n} P(D = d_i | x_i)^{\delta_i} P(D \geq e_i | x_i)^{1-\delta_i} \\ &= \prod_{i=1}^{n} (f(d_i, x_i))^{\delta_i} \left(\int_{e_i}^{\infty} f(e_i, x_i) dt \right)^{1-\delta_i} \\ &= \prod_{i=1}^{n} \lambda(x_i)^{\delta_i} exp\{-\lambda(x_i)(e_i + (d_i - e_i)\delta_i)\}. \end{aligned}$$

The learning process amounts to minimizing the following negative log-likelihood:

$$\begin{aligned} NLLH(\lambda) &= -\log(\mathcal{L}(\lambda)) \\ &= \sum_{i=1}^{n} \delta_i \log(\lambda(x_i)) - \lambda(x_i)(e_i + (d_i - e_i)\delta_i). \end{aligned} \tag{8}$$

4.4 The Bid-DC's Bidding Part

Based on the above observation and analysis, Bid-DC's bidding part formulates bidding process as an CMDP. At each time step $t = 1, 2, \cdots, N$, the CMDP corresponding to ad l decides the bids based on the current state. The state, action, reward, constraint, policy are set as:

State S: We design the state at step t as a tuple $s_t^l = [\mathbf{c} = \langle c_{t-k}^l, \cdots, c_t^l \rangle, \mathbf{d} = \langle d_{t-k}, \cdots, d_t \rangle, \mathbf{f} = \langle f_0, f_1, \cdots, f_m \rangle$, where $\mathbf{c} \in \{0,1\}^k$ records the user click information at the past k time steps to denote the user history information, where k is a hyper-parameter; $\mathbf{f} = \langle f_0, f_1, \cdots, f_m \rangle]$ denotes the bidding feature vector which includes the user features (like gender, shopping level, age, etc.), ad features (ad category, ad group, ad position, etc.), where m is a hyper-parameter; $\mathbf{d} = \langle d_{t-k}, \cdots, d_t \rangle$ represents the difference between cost-per-conversion and the expected cost-per-conversion at the past k steps to denote the bidding history information. For $j = 0, 1, \cdots, k$, d_{t-j} is defined as

$$d_{t-j} = v^l - cpc_{t-j}^l,$$

where cpc_{t-j}^l is defined in Equation (1). For the cases that $t < k$, we use zero to padding.

Action A: $A \subseteq \mathfrak{R}^+$ is the set of actions (bids) the agent can choose from. That is, at each t, the agent chooses $a_t^l \in A$ for conducting the bidding.

Reward R: The reward can be considered as an evaluation of the quality of the bidding. From the viewpoint of platform, an important goal is to maximize platform revenue. Therefore, at time step t, we set the reward of issuing bid a_t^l on state s_t^l as the charges to the advertiser:

$$r_t^l = a_t^l \cdot c_t^l,$$

where $c_t^l \in [0,1]$ denotes whether the user clicked the ad, since the platform will charge the advertiser for a_t^l if the user clicks the ad.

Constraint C: To avoid compensation, Bid-DC makes use of CMDP which augments the traditional MDP with a set of auxiliary cost functions C which restrict the bids that the model can choose. Instead of setting the constraint based on cpc_t^l (i.e., which has been used in existing bidding methods), Bid-DC sets the cost based on the estimation of $\widetilde{cpc}_t^l$ by Bid-DC's delay part, which has the potential to achieve higher revenue.

Specifically, following the practices in [1], the constraint for Bid-DC can be defined as a set $C = \{(C_t^l, \zeta_t^l = 0)\}$ where each constraint (for ad l at the step t) C_t^l is a mapping to a binary constraint cost and $\zeta_t^l = 0$ are the limits.[8] C_t^l is a function defined on the log $S = \{(c_i^l, \alpha_{i,t}^l)\}_{i=1}^t$ which is collected for l until step t:

$$C_t^l = \begin{cases} 0 & \widehat{cpc}_t^l \leq v^l \times (1+\beta); \\ 1 & \text{else}, \end{cases} \tag{9}$$

where β is the fixed compensation rate negotiated by the advertiser and platform, and $\widehat{cpc}_t^l$ is the estimation of $\widetilde{cpc}_t^l$ by Bid-DC's delay part (i.e., Equation (6)).

Policy π: In Bid-DC, the policy is implemented as a 3-layer MLP which takes $\mathbf{s}_t^l$ as inputs (concatenating the sequence of $\mathbf{c} = \langle c_{t-k}^l, \cdots, c_t^l \rangle$, sequence $\mathbf{f} = \langle f_0, f_1, \cdots, f_m \rangle$ and sequence of $\mathbf{d} = \langle d_{t-k}, \cdots, d_t \rangle$ as a $2 \times (k+1) + m$ input nodes) and output a non-negative real number which represents the bid for the clicking of ad l, parameterized by a set of parameters $\mathbf{\Theta}$, denoted as $\pi(s_t^l; \mathbf{\Theta})$.

[8] Please note that we define all of the limits as zero because $C_t^l \in \{0,1\}$ and we don't want any of the bids violating its constraints.

In Bid-DC, the CMDP also requires that the policy be chosen from a set of feasible policies:

$$\Pi_C = \{\pi \in \Pi_\Theta : \forall i, J_{C_i}(\pi) \leq 0\}, \tag{10}$$

where Π_Θ is the set of all possible policies defined by the neural network parameterized by Θ, and

$$J_{C_i}(\pi) = \underset{\tau\sim\pi}{E}\left[\sum_{t=0}^{N} \gamma^t C_t^l\right],$$

denotes the expected discounted return of policy π with respect to the binary constrained cost function C_t^l, and $\gamma \in [0,1]$ is the discount factor. In Equation (10), $J_{C_i}(\pi) \leq 0$ means the upper limit is zero, and all of the actions should satisfy the constraint.

Note that the cost value is C_t^l in Equation (9) and the upper limit is set to 0, which means we do not allow any action to violate any constraint.

4.5 Online RL for Bid-DC's Bidding Part

Constraint RL is adopted for determining the parameters Θ. Theoretically, the learning of optimal policy π^* amounts to the following optimization problem [61, 77]:

$$\pi^* = \underset{\pi\in\Pi_C}{\arg\max}\ J(\pi) = \underset{\tau\sim\pi}{E}\left[\sum_{t=0}^{N} c_t^l \cdot a_t^l\right], \tag{11}$$

where τ is the episode sampled according to π, and the objective $J(\pi)$ represents the expected total revenue of the platform.

In Bid-DC's bidding part, RL algorithm of CPO [1] is used to conduct the optimization in an online manner. Specifically, after collecting the data $\mathcal{D}$ using the Algorithm 1, CPO is executed on $\mathcal{D}$ which actually conducts local policy search [52]. Specifically, for the kth iteration of the training, its goal is to update π_k to π_{k+1} by solving the following problem:

$$\begin{aligned} \pi_{k+1} &= \underset{\pi\in\Pi_C}{\arg\max}\, J(\pi) \\ s.t. \quad & KL(\pi, \pi_k) \leq \delta, \end{aligned} \tag{12}$$

where "KL" is the KL-divergence [7], and $\delta > 0$ is a step size. Here we use KL-divergence constraint to prevent excessively large model updates and ensure training stability [57]. However, Equation (12) cannot be directly solved to update the policy, because it requires offline evaluation of the policy π_{k+1} to determine whether the policy π_{k+1} can ensure the constraints which is known to be difficult [34]. Therefore, motivated by the CPO algorithm [1], we replace the objective and constraints in Equation (12) with surrogate functions. The approximation to Equation (12) is:

$$\begin{aligned} \theta_{k+1} &= \underset{\theta}{\arg\max}\ g^T(\theta - \theta_k) \\ s.t. \quad & c + b^T(\theta - \theta_k) \leq 0 \\ & \frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k) \leq \delta, \end{aligned} \tag{13}$$

where θ denotes the parameter in policy π, g denotes the gradient of the objective, b denotes the gradient of the constraint, H is the Hessian of the KL-divergence and $c \doteq J_C(\pi_k)$. A dual to Equation (13) is:

$$\max_{\nu\succeq 0, \lambda\geq 0} \quad \frac{-1}{2\lambda}(g^T H^{-1} g - 2r^T \nu + \nu^T S \nu) + \nu^T c - \frac{\lambda\delta}{2}, \tag{14}$$

Algorithm 1: Data Collection and Online Bidding

```
Require: ad $l$, expected cost-per-conversion $v^l$; compensation rate $\beta$
Ensure: replay buffer $\mathcal{D}^l$
$\mathcal{D}^l \leftarrow \emptyset$    ▹ Init replay buffer
$(\mathbf{ck}^l, \mathbf{cv}^l) \leftarrow (\emptyset, \emptyset)$    ▹ Init click and conversion sequences
TotalCost $\leftarrow 0$    ▹ Init total cost
for $t = 1, 2, \dots$ do
    Observe state $s_t^l$
    Bidding with $a_t^l \leftarrow \pi(s_t^l; \Theta)$, displaying $l$ and other ads
    Receive user feedback signals $c_t^l$ and $\{\alpha_{t',t}^l : t' \leq t\}$    ▹ user click on current bid, and some delayed conversions
    TotalCost $\leftarrow$ TotalCost $+ a_t^l \cdot c_t^l$    ▹ Add the cost
    $\mathbf{ck}^l \leftarrow \mathbf{ck}^l \oplus \{c_t^l\}$    ▹ Append to click sequence
    $\mathbf{cv}^l \leftarrow \mathbf{cv}^l \oplus \{\alpha_{t,t}^l = 0\}$    ▹ Append to conversion sequence
    For all $\{\alpha_{t',t}^l : t' \leq t\}$: $\mathbf{cv}^l[t'] \leftarrow \alpha_{t',t}^l$    ▹ Update conversion sequence with newly received signals
    $\mathbf{cv2} \leftarrow \mathbf{cv}$    ▹ For estimating $\hat{\alpha}_{i,N}^l$'s
    for $j = 1, \cdots, t$ do
        if $\mathbf{ck}[j] = 1 \wedge \mathbf{cv}[j] = 0$ then
            $\mathbf{cv2}[j] \leftarrow e^{-\lambda \Delta t}$    ▹ Prob. of conversion, Equation (5)
        end if
    end for
    $\widehat{cpc}_t^l \leftarrow$ TotalCost$/\sum_{j=1}^{t} \mathbf{cv2}[j]$    ▹ Equation (6)
    $C_t^l \leftarrow \begin{cases} 0 & \text{if } \widehat{cpc}_t^l \leq v^l \cdot (1+\beta) \\ 1 & \text{else} \end{cases}$    ▹ Calculate constraint, Equation (9)
    $r_t^l \leftarrow c_t^l \cdot a_t^l$    ▹ Calculate reward
    $\mathcal{D}^l \leftarrow \mathcal{D}^l \bigcup \{(s_t^l, a_t^l, r_t^l, C_t^l)\}$
end for
return $\mathcal{D}^l$
```

where $r \doteq g^T H^{-1} b$, $S \doteq b^T H^{-1} b$. Then, we can update the policy π_k through duality as follows:

$$\theta_{k+1} = \theta_k + \frac{1}{\lambda^*} H^{-1}(g - bv^*),$$

where λ^* and v^* is a solution to Equation (13). The detailed description is shown in Appendix A.

The bidding and collecting data process can be carried out together which is shown in Algorithm 1.

4.6 Analysis of Bid-DC

Bid-DC provides a new approach to bid by considering the impact of delayed conversions in OCPC. More importantly, Compared with the existing bidding models such as PID, WL, and RSDRL etc, Bid-DC offers several advantages besides enhancing the platform revenue as discussed before.

First, the bidding mechanism provided by Bid-DC will effectively increase the chances of ad exposure. At each step t, Bid-DC uses the true cost-per-conversion $\widehat{cpc}_t^l$ in Equation (6) rather than the original cpc_t^l in Equation (1) to guide the bidding. Due to the delayed conversion nature of the clicked ads, it is easy the know that $\widehat{cpc}_t^l \leq cpc_t^l$, which leads to the bidding models to make more aggressive bids (i.e., higher bidding prices). Since the order and the exposure opportunity of an ad are largely determined by the bid, increasing the bid will raise the number of impressions the ad can get, which will eventually benefit for brand promotions. Moreover, higher exposure indicates higher conversion numbers, therefore, Bid-DC can also improve the advertiser's profit.

Second, Bid-DC increases the bid stability and prevents the high fluctuations during the bidding process. Existing studies have shown that high fluctuations would hurt the stability of the bidding model [64, 82]. (1) One reason that causes the high fluctuations in existing methods is the payment and cost-per-conversion estimation mechanisms. After each bidding, existing methods have to decrease their future bids if observed user clicks. This is because the advertiser pays the clicks immediately while the conversions will occur in the future, resulting overestimation of cost-per-conversion cpc_t^l, and finally increase the bids in the future time steps. In Bid-DC, however, the probability of the conversion is estimated at each step. Therefore, the problem of overestimating the cost-per-conversion is alleviated; (2) Another reason that causes the high fluctuations in existing methods is the uncertain returning of the delayed conversion signals. Suppose that a lot of new conversion signals (i.e., newly observed $\alpha_{t',t}^l = 1$'s) are returned at time t, existing methods have to suddenly decrease their cpc_t^l and thereafter significantly increase the bids, causes the high fluctuations. In Bid-DC, the fluctuations will be suppressed because it uses $\hat{\alpha}_{t',N}^l = 1$ to estimate the $\widehat{cpc}_t^l$. Even there is a large number of new conversion signals, the decreasing of $\widehat{cpc}_t^l$ will be much smaller because Bid-DC has estimated their conversion probability (i.e., change the conversion signals from 0 to their conversion probability before the real conversions).

Third, Bid-DC can be easily adapted to different conversion goals. It has been widely observed that in real applications, different conversion goals usually have different delay time (e.g., user registration usually have shorter delay time than user purchase). Bid-DC learns the optimal parameter $\lambda(x) \geq 0$ in Equation (5) over history to characterize the delayed time.

Please note that If the ad's conversion goal has a very short delay, i.e., if the delayed conversions are always received before the updating process, or if the conversion rate is very low, resulting in few conversions, Bid-DC's effectiveness will be reduced due to the limited number of delayed conversions.

5 Experimental Study

We conducted both online and offline experiments to verify the effectiveness of Bid-DC in OCPC bidding. The Bid-DC and experiments' implement details are shown in Appendix B.

5.1 Experimental Setting

5.1.1 Dataset. The experiments were conducted based on two publicly available datasets iPinYou [31, 44], Criteo [20] and two industrial-scale OCPC datasets collected from a commercial OCPC ad platform. We use these datasets to test the performance of Bid-DC over different ads.

iPinYou dataset consists of 64.75M bid record, 19.50M impressions, 14.79K clicks and 1.1K conversions from some real advertising campaigns [8]. Each click/conversion record includes the bid and time. We use the information and transform into a dataset that can be used for OCPC. Specifically, we extracted the clicks and conversions log for each advertising campaign from the iPinYou dataset, and then sorted them according to the time stamp to simulate the user feedback. Additionally, we also use the bidding prices in the log data as the expected cost-per-conversion, which were pre-set by the advertisers. According to the information released by the data publisher [44], the last three-day data of each campaign is split as the test data, and the rest as the training data. To test the robust performance of Bid-DC (the performance over different advertisers), we further split 8 advertisers in the iPinYou dataset into two categories: (1) *Four normal advertisers*: advertisers who received at least 20 conversions; (2) *Four special advertisers*: advertisers who received less than 20 conversions (most advertisers in this category received only 0 or 1 conversion).

Criteo includes 30 days of Criteo live traffic data which contains 16.5M bidding records [20]. Each auction includes whether this auction leads to the click/conversion, the highest bid price and value corresponding to the conversion. We use the information and transform it into an OCPC bidding dataset. We sorted the auctions based on the timestamp, and used the conversion value to denote the expected CPA. Following the previous bidding methods [82, 83], in each auction, if the bid generates by the bidding method upper than the highest bid price, the advertiser will win this auction and will get charge/conversion based on the click signal/conversion signal respectively. Since here we use the first-price auction [15], the platform will charge the advertiser the bid. According to the publisher [20], the 21 previous days are considered as training data, while the last 7 days of logs are considered as test data.

The industrial OCPC dataset 1 consists of 8 days log data corresponding to 10 randomly selected OCPC advertisers from an online advertising platform for mobile apps. The dataset in total consists of about 1.7M clicks (downloads) and 82K conversions (activation). We used the first 7-day log data as the training set and the last day as the testing set.

The industrial OCPC dataset 2 settings are identical to the industrial OCPC dataset 1, except the conversion goal is set to purchase. The dataset consists of about 2M clicks (downloads) and 32K conversions (purchases), which contains more sparse conversions than that of the industrial OCPC dataset 1.

The Expanded industrial OCPC dataset 1 is a larger version of *The industrial OCPC dataset 1*, which consists of 8 days log data corresponding to 100 randomly selected OCPC advertisers.

The Expanded industrial OCPC dataset 2 is a larger version of *The industrial OCPC dataset 2*, which also consists of 8 days log data corresponding to 100 randomly selected OCPC advertisers.

In the experiments, the iPinYou dataset and the criteo dataset serves as the standard public bidding datasets that facilitate the reproduction and comparison of other methods with Bid-DC. The two industrial OCPC datasets enable us to effectively present and analyze the performance of Bid-DC in real-world OCPC settings, and test whether Bid-DC is capable of handling different conversion goals as mentioned in Section 4.6. The two expanded OCPC dataset is used to to validate the effectiveness and robustness of Bid-DC.

5.1.2 Compared Methods and Evaluation Metric. We compare the Bid-DC with the following baselines in the experiments to show the performance of Bid-DC:

PID [6]: PID controller is a classic feedback control mechanism which has been widely used in the bidding of Real-Time Display Advertising [82]. PID controller uses proportional, integral, and derivative items to adjust the bids according to the estimated cost-per-conversion.

WL [19]: WL controller is another feedback control mechanism. Compared to PID, the WL controller only uses the proportional item.

LP [63]: LP is used to maximize an object while still meet some constraints. Maximizing platform revenue in OCPC can also be treated as an LP problem, where the future clicks and conversion numbers are modeled by a Gaussian distribution, and solve an LP problem at the each time step to obtain the optimal bid for the next time step.

RSDRL [59]: RSDRL uses RL and an ROI-sensitive agent to bid in OCPM setting. In this article, we used it to bid in OCPC.

USCB [27]: USCB is a DRL bidding method which uses hyper-parameters to non-linearly control the constraint while maximize the objective.

TEE [42]: TEE uses offline RL to train the optimal policy from data. It also proposes Safe Exploration by Adaptive Action Selection to ensure the safety while online.

Moreover, to test Bid-DC can automatically find the optimal bidding strategy for each advertiser, here, we tune the hyper-parameters of each baseline bidding strategy using the training data set.

As for evaluation metrics, following the previous work [9, 22, 59], we mainly use the platform revenue to test the performance improvements over the corresponding baselines.

Table 3 reports the revenue of the proposed Bid-DC and the baselines on iPinYou dataset, Criteo dataset, industrial OCPC dataset 1 and industrial OCPC dataset 2, respectively. The compensation bound donates the 120% of the expected conversion values (i.e., the platform should compensate advertiser if the total charges exceed it) are listed in the last column. Each line represents an ad and their IDs in the dataset are listed in the first column (denoted as Campaign No.). We found that on all four datasets, Bid-DC outperformed all the baselines while ensuing the total charges are lower than the compensation bound. The results verified the effectiveness of Bid-DC in safely improving the revenue, on both types of advertisements. Also, we can see LP performs poor on those datasets, it is because LP is a static bidding method that cannot handle dynamic bidding environments.

As mentioned in Section 5.1.1, the iPinYou dataset contains two types of advertisers: Normal advertisers (the first 4 lines in the table) and Special advertisers (the last 4 lines in the table). We found that on both types of the campaigns, Bid-DC can outperform all of the baselines, which demonstrates the robustness of Bid-DC. Compared to special advertisers which has much sparse clicks and conversions, the revenue achieved by Bid-DC on normal campaigns is much (relatively) closer to their upper bounds. The results is easy to explain: with more observed clicks and conversions, Bid-DC can predict the future conversions more accurately, leading to more accurate bids.

We also found Bid-DC outperforms the baselines on advertisers with different conversion goals (i.e., OCPC dataset 1 and OCPC dataset 2) in terms of revenue. The results show, through constraint RL, Bid-DC can automatically learn the optimal bidding strategy for different advertisers without any human effort which makes Bid-DC easily adapts to other advertising scenario.

Moreover, we test the performance of Bid-DC on the Expanded industrial OCPC dataset 1 and OCPC dataset 2. Table 4 reports the revenue and standard deviation of the proposed Bid-DC and the baselines on the expanded OCPC dataset. We found Bid-DC outperforms all baselines, which demonstrates the effectiveness and robustness of Bid-DC.

5.2 Empirical Analysis

In this section, we conducted empirical analysis for the proposed Bid-DC. All of the analysis were conducted on the campaigns from the OCPC industrial dataset 1, which accurately reflect the bidding effects in real bidding environment. We mainly compared Bid-DC with PID because PID has been widely used in the real online bidding environment.

5.2.1 How Bid-DC Alleviates the Problem of Overestimating Cost-per-Conversion? Bid-DC is motivated from the observation that existing methods make over-conservative bids due to the overestimation of cost-per-conversion, as shown in Figure 4(a). We conducted experiments to show whether Bid-DC addressed the issue.

Figure 4 shows the bids and cost-per-conversion curves generated by Bid-DC for the same campaign as that of used in Figure 4(a). Comparing to the results in Figure 4(a), we can see that Bid-DC conducts more aggressively bids (curve “Bid-DC’s bid”), making the “true cost-per-conversion” curve much closer to the line “expected cost-per-conversion.” The results clearly indicate that the prediction of the future conventions in Bid-DC effectively improved the bids, and finally enhanced the revenue.

Though Bid-DC makes aggressive bids, the curve “true cost-per-conversion” is still lower than the compensation bound at most of the time steps. At the end of the episode, “true cost-per-conversion”

Table 3. Revenues of Bid-DC and Baselines Over Each Advertisement on iPinYou Test Set (Sums of 3 Days' Revenues), Criteo Test Set (Sums of 7 Days' Revenues), Industrial OCPC Test Set 1 (1 Day Revenue) and Industrial OCPC Test Set 2 (1 Day Revenue)

Dataset	Campaign No.	PID	WL	LP	RSDRL	USCB	TEE	Bid-DC	Compensation bound
iPinYou	3476	3,940.45	3,928.12	3,814.55	4,216.56	4,104.78	4,006.31	**4,285.15**	5,018.4
	2259	17,712.27	17,683.71	14,275.51	18,591.39	18,464.39	18,601.33	**18,943.45**	22,041.6
	2821	69,344.52	69,332.16	60,256.39	72,411.61	72,512.98	70,145.62	**73,295.75**	84,506.4
	3358	27,360.99	27,356.81	27,648.03	27,926.11	28,014.32	28,127.59	**28,328.31**	32,001.6
	3386	27.74	27.74	-	43.66	43.7	44.1	**44.8**	300
	3427	21.65	21.64	-	33.69	32.59	33.04	**34.08**	234
	1458	73.87	73.87	1,159.41	97.35	97.42	98.51	**105.38**	360.0
	2997	24.87	24.87	-	35.07	35.68	35.17	**36.02**	277
Criteo	1	355.67	362.13	298.72	374.29	385.96	373.21	**399.25**	412.4
	2	302.74	303.12	263.65	359.81	354.12	361.18	**362.98**	397.21
	3	284.31	284.13	213.74	322.78	317.25	319.41	**343.75**	356.51
	4	449.43	444.15	401.98	489.97	491.45	487.27	**501.23**	514.64
	5	131.62	130.42	101.41	116.24	118.92	105.64	**129.82**	132.86
	6	121.97	122.02	102.81	135.61	136.08	132.11	**141.06**	149.97
	7	92.21	91.98	90.03	111.21	106.28	108.59	**124.88**	129.98
OCPC 1	1	1	0.99	0.87	1.02	1.04	1.01	**1.12**	1.24
	2	1	0.99	0.81	1.12	1.14	1.09	**1.18**	1.31
	3	1	1	0.77	1.05	1.07	1.04	**1.09**	1.18
	4	1	1.01	0.8	1.01	1.03	1.01	**1.07**	1.17
	5	1	0.99	0.65	1.03	1.04	1.05	**1.09**	1.17
	6	1	0.98	0.78	1.08	1.04	1.05	**1.13**	1.29
	7	1	1	0.92	1.02	1.01	1.03	**1.07**	1.15
	8	1	1	0.73	1.04	1.07	1.06	**1.12**	1.19
	9	1	0.98	0.91	1.09	1.05	1.11	**1.13**	1.16
	10	1	0.99	0.87	1.06	1.06	1.06	**1.15**	1.23
OCPC 2	1	1	0.98	0.72	1.07	1.08	1.07	**1.08**	1.19
	2	1	0.99	0.91	1.05	1.04	1.07	**1.07**	1.18
	3	1	1	0.73	1.07	1.07	1.02	**1.09**	1.17
	4	1	0.98	0.65	1.16	1.14	1.12	**1.19**	1.33
	5	1	1.01	0.76	1.05	1.06	1.02	**1.08**	1.12
	6	1	0.99	0.93	1.07	1.08	1.04	**1.09**	1.14
	7	1	1	0.79	1.08	1.05	1.02	**1.10**	1.18
	8	1	0.98	0.72	1.09	1.08	1.08	**1.12**	1.17
	9	1	1.01	0.79	1.04	1.05	1.02	**1.06**	1.09
	10	1	1.01	0.87	1.06	1.04	1.07	**1.09**	1.11

The compensation bound is defined as 120% × v^l × #final conversions. The platform will compensate to the advertiser if the revenue is larger than the compensation bound. '-' means the value is not available because the conversions are too sparse, causing numerical errors in LP. To avoid privacy issues, the results on industrial OCPC test set 1 and test set 2 are displayed as the relative values over the revenue of PID. Note that less than 1 means under-performing PID. The best results are bolded.

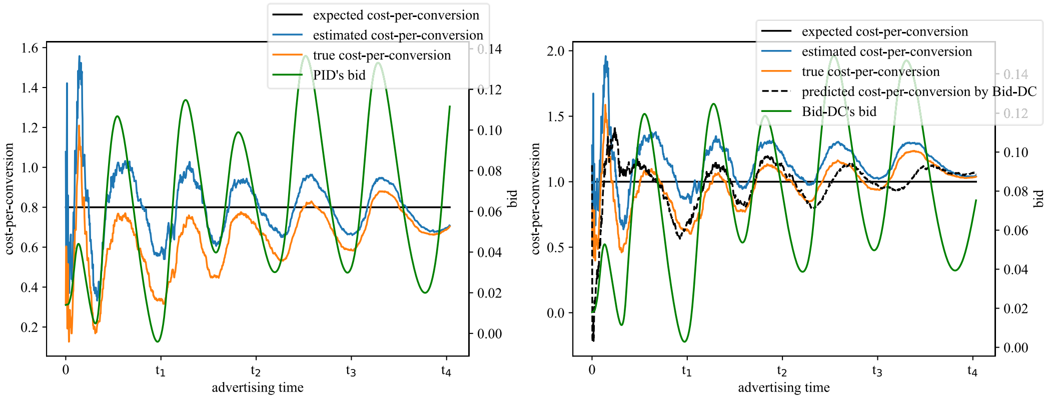
is very close to the "expected cost-per-conversion," indicating that the constraints in CMDP safely make bids so that the final cost-per-conversion is under control and compensation is avoided.

Figure 4 shows a new curve denoted "predicted cost-per-conversion by Bid-DC," which is calculated with the predicted delayed conversions $\hat{\alpha}_t^l$ (i.e., $\widehat{cpc}_t^l$ in Equation (6)). We can see that the curve is much closer to the "true cost-per-conversion" than the curve "estimated cost-per-conversion."

Table 4. The Average and Standard Deviation Revenue of Bid-DC and Other Baselines Over Two Expanded OCPC Datasets

Expanded Dataset	PID	WL	LP	RSDRL	USCB	TEE	Bid-DC	Bound
Expanded OCPC 1	1 ± 0.0	0.99 ± 0.0	0.87 ± 0.0	1.14 ± 0.007	1.13 ± 0.0	1.15 ± 0.006	**1.21 ± 0.005**	1.30
Expanded OCPC 2	1 ± 0.0	1 ± 0.0	0.76 ± 0.0	1.10 ± 0.008	1.09 ± 0.0	1.11 ± 0.007	**1.15 ± 0.008**	1.24

The compensation bound (Bound for short) is defined as 120% × v^l × #final conversions. Here we carefully select baselines' hyper-parameters on the training set to ensure the baselines' revenue does not exceed the compensation bound. To avoid privacy issues, the results are displayed as the relative values over the revenue of PID. Please note that PID, WL, LP and USCB are deterministic strategies, which means their standard deviations are 0. The best results are bolded.



(a) PID's Bidding Curve

(b) Bid-DC's bidding Curve

Fig. 4. Cost-per-convention curves and bid curves generated by Bid-DC and PID on an ad from a real industrial OCPC platform. For both Figure 4(a) and (b), the curve "true cost-per-conversion" denotes the cost-per-conversion which takes the future conversions of the clicked ads into consideration (i.e., $\widetilde{cpc}^l_t$ in Equation (3)) and line "expected cost-per-conversion" represents the expected cost-per-conversion set by the advertiser which does not change over the time. For Figure 4(a), the curve "estimated cost-per-conversion" denotes the cost-per-convention curve generated by PID, i.e., Equation (1), and the curve "PID's bid" denotes the PID's bidding curve; For Figure 4(b), the curve "estimated cost-per-conversion" denotes the cost-per-conversion estimated without taking delayed conversions into consideration (i.e., cpc^l_t in Equation (1), which is the same curve in Figure 4(a), but with Bid-DC as the bidding controller.). The "Bid-DC's bid" denotes the bidding curve by the proposed Bid-DC. The curve "predicted cost-per-conversion by Bid-DC" denotes the cost-per-conversion calculated based on the predicted $\hat{\alpha}^l_{t,N}$ in Bid-DC (i.e., $\widehat{cpc}^l_t$ in Equation (6)). All values have been re-scaled for avoiding privacy issues.

To quantitatively verify Bid-DC can more accurately estimate the cost-per-conversion, we randomly select 24 time steps during the advertising period for 100 OCPC advertisers. We then use the average **Mean Absolute Percentage Error (MAPE)** [29] as a metrics to measure the difference between the predicted and true cost-per-conversion: MAPE $= \frac{1}{100}\sum_{l=1}^{100}\frac{1}{24}\sum_{i=1}^{24}\frac{|\widehat{cpc}^l_{t_i}-\widetilde{cpc}^l_{t_i}|}{\widetilde{cpc}^l_{t_i}}$, where $\widehat{cpc}^l_{t_i}$ is the predicted cost-per-conversion at t_i for ad l, $\widetilde{cpc}_{t_i}$ is the true cost-per-conversion at t_i for ad l and $t_1, t_2, \ldots, t_{24}$ denote 24 time points. We also added two baselines, which are commonly used to estimate delay, to demonstrate Bid-DC's delay part's effectiveness in estimating the true cost-per-conversion:

Table 5. The MAPE Value (Smaller Means Better) of Bid-DC and Other Baselines for 24 Time Points Over 100 Randomly Selected Ads on a Real OCPC Platform

Methods	Not-predict	FSIW	Bayesian	Bid-DC
MAPE	7.45	3.45	2.71	**1.02**

Not-predict means we do not consider the delayed conversions when estimating the cost-per-conversion. The best result is bolded

- —*Bayesian methods* are widely used to estimating the delay time [21, 60]. The core idea of bayesian methods is we first set a prior relation between the feature and the delay time and then update the relation through Bayes' theorem. Although they are widely used in estimating delay time, in OCPC, what we need to predict is the probability of a conversion occurs within a certain period rather than a specific delay time, which is inconsistent with goal of bayesian methods.
- —*FSIW* is usually used to dealing with delayed feedback problem in the CVR prediction [76]. However, in OCPC, we only consider the conversions arrive before the end of advertising, clicks at different time has different attribution window, e.g., click arrives early during the advertising period has more probability to convert than which arrives later, which is inconsistent with FSIW's assumption.

Table 5 shows the result. We found the MAPE value of Bid-DC is **1.02** which is much smaller than the baselines, which means predicted cost-per-conversion by Bid-DC is closer to true cost-per-conversion. The results indicated that Bid-DC effectively alleviated the problem of overestimating the cost-per-conversion that is caused by the delayed conversions.

Comparing the "estimated cost-per-conversion" curves in Figure 4(a) and Figure 4 (both are calculated with Equation (1), we found the aggressive bids made by Bid-DC greatly enhanced the originally estimated cost-per-conversion. This is because increasing bids directly increases the total costs. However, the "estimated cost-per-conversion," "true cost-per-converstion," and "predicted cost-per-conversion" merge at the end of the episode. This is because the episode ends at time N with all conversion signals returned. That is, at the time N, there is no delayed conversion problem anymore.

Also note that at the beginning of the episode, Figure 4(a) and Figure 4 showed high fluctuations on bids and cost-per-conversion. This is because at the beginning, the number of clicks and conversions are small. A small number of clicks and conversions have high impacts on the estimations of cost-per-conversion. This also confirmed the results reported in Table 3: campaigns with sparse clicks and conversions are relatively difficult to improve.

5.2.2 The Effects of the Constraints in Bid-DC.

Bid-DC uses the constraints in CMDP for avoiding compensation. Figure 5 shows the results of Bid-DC with and without the constraints, i.e., setting all $C_t^l = 0$ and directly updating the policy to maximize the revenue through trust region methods [57], on the same campaign used in Section 5.2.1. We also show the compensation bound (denoted as "compensation bound") for reference. We found that without the constraint, it is easy for Bid-DC to bid too high, making cost-per-conversion exceed the compensation bound and cause compensation at the end of the episode. After adding the constraint, the CMDP module in Bid-DC effectively controlled the cost-per-conversion values and avoided the compensation.

Moreover, to avoid reward sparsity [25], i.e., reward is only received at the end, potentially causing the bidding model to exceed the advertiser's constraints, as shown in Equation (9), Bid-DC applies constraints at each bidding step, which is stricter than the advertiser's constraint. Table 6

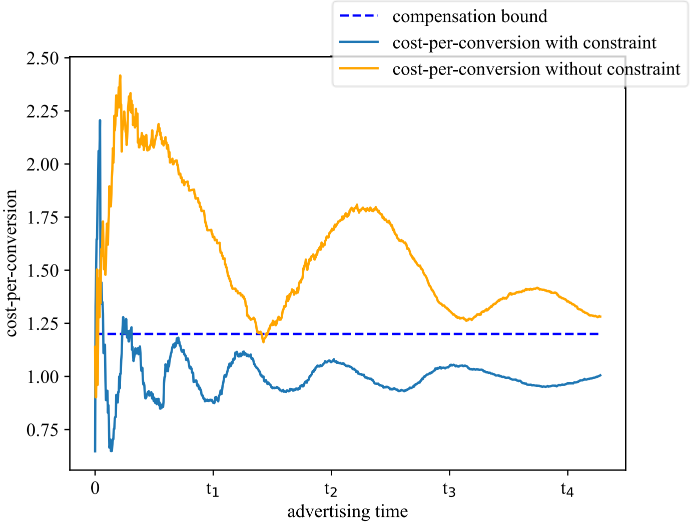


Fig. 5. The cost-per-conversion curves of Bid-DC with and without constraints, respectively. Curve "cost-per-conversion with constraint" denotes the curve of cpc_t^l / v^l for Bid-DC. Curve "cost-per-conversion without constraint" denotes the cpc_t^l / v^l for variation of Bid-DC without the constraint (i.e., by setting all $C_t^l = 0$ in Algorithm 1). The curve "compensation bound" donates the relative compensation bound (i.e., 120%).

Table 6. The Average Revenue and Compensation Numbers (i.e., How Many Ads Result in Compensation?) of Bid-DC and Bid-DC Which Only Has $C_N = 1$ if $\widehat{cpc}_N^l > v^l \times (1 + \beta)$ (Named as Bid-DC$_N$) on the Expanded OCPC Dataset 1

Methods	Average Revenue	Compensation Number
Bid-DC$_N$	1.28	8
Bid-DC	1.21	0

To avoid privacy issues, the results are displayed as the relative values over the revenue of PID.

reports the results of experiments on the expanded OCPC dataset 1. From the results, we found that if we only apply the constraint at the end, Bid-DC will cause some ads to compensate, which is the platform aims to avoid. Therefore, we apply constraints at every bidding step to ensure that Bid-DC results in no compensation.

5.2.3 Can Bid-DC Improve the Bid Stability? In Section 4.6, we show that Bid-DC has more stable bids. In this section, we conducted experiments to verify the conclusion. Figure 6 shows the bidding curves of Bid-DC and PID, respectively, based on the same campaign used in Section 5.2.1. Comparing these two curves, it is easy to find that "PID's bid" fluctuates more violently than "Bid-DC's bid". We also calculated the variances over all the bids (smaller is better). The variance of the bids made by PID is 1.29, which is much larger than the variance of the bids made by Bid-DC is **0.57**. The results confirmed the conclusion that Bid-DC has the ability of making more stable bids.

We note that at the beginning of the episode, Bid-DC has a short time high fluctuates. This is caused by the unstable estimation of $\widehat{cpc}_t^l$ due to the too small number of clicks and conversions. In Bid-DC, the high fluctuates will be alleviated quickly after receiving more clicks and conversions.

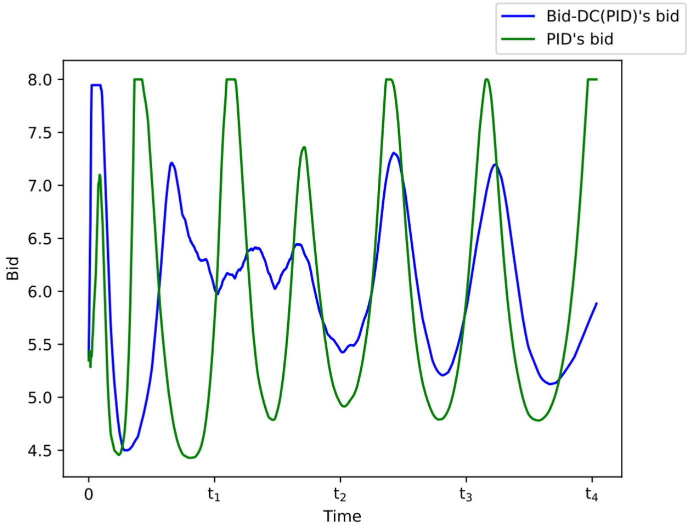


Fig. 6. Bidding curves generated by the PID and Bid-DC.

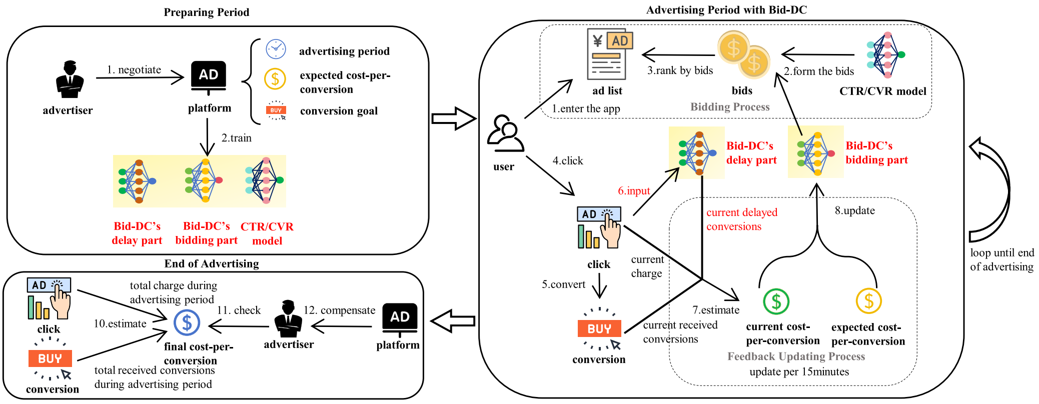


Fig. 7. The deployment of Bid-DC in OCPC. Please note that the yellow background with red name is the difference between Bid-DC's bidding system and the original system, i.e., the difference compared with Figure 1.

5.3 Online Experimental Results

Figure 7 shows how to deploy Bid-DC in OCPC. Compared with Figure 1, there are three differences during the preparing period and advertising period.

(1) In the preparing period, rather than just learning the bidding model and the CVR/CTR model, we will learn Bid-DC's delay part (i.e., discussed in Section 4.3), Bid-DC's bidding part (i.e., discussed in Section 4.5) and the CTR/CVR model.
(2) During the advertising period, when a user clicks the ad, Bid-DC's delay part will take the click's feature as input and output the convert probability for the click (i.e., Equation (5)). The platform will cache all convert probabilities to prepare for the feedback updating process.
(3) For every 15 minutes, the feedback updating process starts. The platform estimates the cost-per-conversion through total charge of the advertiser, current received conversions and

Table 7. Online A/B Test Results

Metric	Relative improvement
Revenues	10%
Click numbers	7%
Conversion numbers	6%
Compensation rate	0%

the cached estimated convert probabilities for the unconverted click (i.e., Equation (6)). This estimated cost-per-conversion $\widehat{cpc}_t^l$ will be used to update the state of Bid-DC's bidding part, helping it bidding more precisely.

We have deployed Bid-DC on a real OCPC platform for a week, comparing it with the baseline bidding method: RSDRL method. Table 7 shows the results the 7-day online A/B test. The revenue and compensation rate metrics are related to the platform and the click numbers and conversion numbers metrics are related to the advertisers. We found all of them are improved without incurring any compensation. From the results of the online A/B test, as discussed in Section 4.6, besides improving the platform's revenue, Bid-DC also can improve the ad's click numbers and conversion numbers which indicates that Bid-DC is beneficial for both advertisers and platform.

6 Conclusion

In this article, we analyzed the delayed conversions in OCPC, and found that the delayed conversions make existing methods overestimate the cost-per-conversion values, leading to over conservative bidding policy, and finally hurt the platform revenue and ad's exposures. Moreover, the hyper-parameters in the existing bidding methods require much human effort to determine for different ads. To deal with those issues, this article proposes a novel bidding mechanism called Bid-DC in which the conversion probability of a clicked but not converted ad is predicted and used to re-estimate the click-per-conversion values. To avoid the risk of compensation, constrained MDP is adapted to modeling the bidding process and constrained RL is used to automatically learn the optimal bidding policy without any human effort. Offline experimental results based on the publicly available iPinYou dataset, Criteo dataset and two real industrial OCPC datasets showed that Bid-DC can effectively enhance the revenues of all campaigns. Online A/B test result shows, beside improving platform's revenue, Bid-DC also can improve the ad's click numbers and conversion numbers which means Bid-DC benefits to both advertisers and platform. Empirical analysis also showed that Bid-DC improved existing methods through more accurately estimation of the cost-per-conversion values and can make more stable bids.

Future work includes theoretical analysis of the unbiasedness and variance of the predicted conversions, and personalized and accurate prediction of the conversion probabilities based on the user and advertisement information.

References

[1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. 2017. Constrained policy optimization. In *Proceedings of the International Conference on Machine Learning*. PMLR, 22–31.

[2] Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphaël Marinier, Leonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, et al. 2020. What matters for on-policy deep actor-critic methods? A large-scale study. In *Proceedings of the International Conference on Learning Representations*.

[3] Ioannis Arapakis, Antonio Penta, Hideo Joho, and Luis A. Leiva. 2020. A price-per-attention auction scheme using mouse cursor information. *ACM Transactions on Information Systems* 38, 2 (2020), 1–30.

[4] Kursad Asdemir, Nanda Kumar, and Varghese S. Jacob. 2012. Pricing models for online advertising: CPM Vs. CPC. *Information Systems Research* 23, 3-part-1 (2012), 804–822.

[5] Andreas Bender, David Rügamer, Fabian Scheipl, and Bernd Bischl. 2020. A general machine learning framework for survival analysis. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD '20)*. Springer, 158–173.

[6] Stuart Bennett. 1993. Development of the PID controller. *IEEE Control Systems Magazine* 13, 6 (1993), 58–62.

[7] Yuheng Bu, Shaofeng Zou, Yingbin Liang, and Venugopal V. Veeravalli. 2018. Estimation of KL divergence: Optimal minimax rate. *IEEE Transactions on Information Theory* 64, 4 (2018), 2648–2674.

[8] Han Cai, Kan Ren, Weinan Zhang, Kleanthis Malialis, Jun Wang, Yong Yu, and Defeng Guo. 2017. Real-time bidding by reinforcement learning in display advertising. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*, 661–670.

[9] Pedro Chahuara, Nicolas Grislain, Gregoire Jauvion, and Jean-Michel Renders. 2017. Real-time optimization of Web publisher RTB revenues. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1743–1751.

[10] Olivier Chapelle. 2014. Modeling delayed feedback in display advertising. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1097–1105.

[11] Dagui Chen, Junqi Jin, Weinan Zhang, Fei Pan, Lvyin Niu, Chuan Yu, Jun Wang, Han Li, Jian Xu, and Kun Gai. 2019. Learning to advertise for organic traffic maximization in E-commerce product feeds. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2527–2535.

[12] Ye Chen, Pavel Berkhin, Bo Anderson, and Nikhil R. Devanur. 2011. 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*, 1307–1315.

[13] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. 2018. Risk-constrained reinforcement learning with percentile risk criteria. *Journal of Machine Learning Research* 18, 167 (2018), 1–51.

[14] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. 2018. A lyapunov-based approach to safe reinforcement learning. *Advances in Neural Information Processing Systems* 31 (2018), 8103–8112.

[15] James C. Cox, Vernon L. Smith, and James M. Walker. 1988. Theory and individual behavior of first-price auctions. *Journal of Risk and Uncertainty* 1 (1988), 61–99.

[16] Imre Csiszár and János Körner. 2011. *Information Theory: Coding Theorems for Discrete Memoryless Systems*. Cambridge University Press.

[17] Will Dabney, Georg Ostrovski, David Silver, and Rémi Munos. 2018. Implicit quantile networks for distributional reinforcement learning. In *Proceedings of the International Conference on Machine Learning*. PMLR, 1096–1105.

[18] Sunhao Dai, Yuqi Zhou, Jun Xu, and Ji-Rong Wen. 2023. Dually enhanced delayed feedback modeling for streaming conversion rate prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 390–399.

[19] Bernard W Dezotell. 1936. Water level controller. US Patent 2,043,530.

[20] Eustache Diemert, Julien Meynet, Pierre Galland, and Damien Lefortier. 2017. Attribution modeling increases efficiency of bidding in display advertising. In *Proceedings of the ADKDD '17*, 1–6.

[21] Feilong Ding, Cheng Chi, Yu Li, and Haining Huang. 2023. Variational bayesian inference time delay estimation for passive sonars. *Journal of Marine Science and Engineering* 11, 1 (2023), 194.

[22] Joaquin Fernandez-Tapia, Olivier Guéant, and Jean-Michel Lasry. 2017. Optimal real-time bidding strategies. *Applied Mathematics Research Express* 2017, 1 (2017), 142–183.

[23] Javier Garcia and Fernando Fernández. 2015. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research* 16, 1 (2015), 1437–1480.

[24] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. 2018. Soft actor-critic algorithms and applications. arXiv:1812.05905. Retrieved from https://arxiv.org/abs/1812.05905

[25] Joshua Hare. 2019. Dealing with sparse rewards in reinforcement learning. arXiv:1910.09281. Retrieved from https://arxiv.org/abs/1910.09281

[26] Mohammadhosein Hasanbeig, Alessandro Abate, and Daniel Kroening. 2018. Logically-constrained reinforcement learning. arXiv:1801.08099. Retrieved from https://arxiv.org/abs/1801.08099

[27] Yue He, Xiujun Chen, Di Wu, Junwei Pan, Qing Tan, Chuan Yu, Jian Xu, and Xiaoqiang Zhu. 2021. A unified solution to constrained bidding in online display advertising. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2993–3001.

[28] John R. Hershey and Peder A. Olsen. 2007. Approximating the kullback leibler divergence between Gaussian mixture models. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '07)*, Vol. 4. IEEE, IV–317.

[29] Timothy O. Hodson. 2022. Root-mean-square error (RMSE) or mean absolute error (MAE): When to use them or not. *Geoscientific Model Development* 15, 14 (2022), 5481–5487.

[30] Yu Hu, Jiwoong Shin, and Zhulei Tang. 2010. Pricing of online advertising: Cost-per-click-through vs. cost-per-action. In *Proceedings of the 43rd Hawaii International Conference on System Sciences.* IEEE, 1–9.

[31] Guojing Huang, Qingliang Chen, and Congjian Deng. 2020. A new click-through rates prediction model based on deep & cross network. *Algorithms* 13, 12 (2020), 342.

[32] Aditya Jain and Sahil Khan. 2021. Optimizing cost per click for digital advertising campaigns. arXiv:2108.00747. Retrieved from https://arxiv.org/abs/2108.00747

[33] Ashish K Jayant and Shalabh Bhatnagar. 2022. Model-based safe deep reinforcement learning via a constrained proximal policy optimization algorithm. *Advances in Neural Information Processing Systems* 35 (2022), 24432–24445.

[34] Nan Jiang and Lihong Li. 2016. Doubly robust off-policy value evaluation for reinforcement learning. In *Proceedings of the International Conference on Machine Learning.* PMLR, 652–661.

[35] Sebastian Junges, Nils Jansen, Christian Dehnert, Ufuk Topcu, and Joost-Pieter Katoen. 2016. Safety-constrained reinforcement learning for MDPs. In *Proceedings of the International Conference on Tools and Algorithms for the Construction and Analysis of Systems.* Springer, 130–146.

[36] Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. 1996. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4 (1996), 237–285.

[37] Sham M. Kakade. 2001. A natural policy gradient. *Advances in Neural Information Processing Systems* 14 (2001), 1531–1538.

[38] Ramin Abolghasemi Komleh. 2023. Investigation of relationship between Google cost-per-click and search-volume on keyword of Chicago tours. *Journal of Information Systems and Informatics* 5, 3 (2023), 1002–1019.

[39] Vijay Konda and John Tsitsiklis. 1999. Actor-critic algorithms. *Advances in Neural Information Processing Systems* 12 (1999), 1008–1014.

[40] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. 2021. Offline reinforcement learning with implicit Q-learning. arXiv:2110.06169. Retrieved from https://arxiv.org/abs/2110.06169

[41] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. 2020. Conservative Q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems* 33 (2020), 1179–1191.

[42] Haoming Li, Yusen Huo, Shuai Dou, Zhenzhe Zheng, Zhilin Zhang, Chuan Yu, Jian Xu, and Fan Wu. 2024. Trajectory-wise iterative reinforcement learning framework for auto-bidding. In *Proceedings of the ACM on Web Conference*, 4193–4203.

[43] Sinan Li, Chun Yuan, and Xin Zhu. 2021. Optimizing enhanced cost Per Click via reinforcement learning without exploration. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN).* IEEE, 1–8.

[44] Hairen Liao, Lingxiao Peng, Zhenchuan Liu, and Xuehua Shen. 2014. IPinYou global rtb bidding algorithm competition dataset. In *Proceedings of the 8th International Workshop on Data Mining for Online Advertising*, 1–6.

[45] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. arXiv:1509.02971. Retrieved from https://arxiv.org/abs/1509.02971

[46] Junwei Lu, Chaoqi Yang, Xiaofeng Gao, Liubin Wang, Changcheng Li, and Guihai Chen. 2019. Reinforcement learning with sequential information clustering in real-time bidding. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 1633–1641.

[47] Stacey Matthews, Jasmine Just, Garry Jennings, Janet Bray, Jesse Lewis, and Amanda Buttery. 2023. Can a targeted social media campaign increase reach of, and engagement with, heart failure self-management resources in culturally and linguistically diverse communities in Victoria, Australia? An evaluation including cost. *Heart, Lung and Circulation* 32, 9 (2023), 1089–1095.

[48] Zhiyu Mou, Yusen Huo, Rongquan Bai, Mingzhou Xie, Chuan Yu, Jian Xu, and Bo Zheng. 2022. Sustainable online reinforcement learning for auto-bidding. arXiv:2210.07006. Retrieved from https://arxiv.org/abs/2210.07006

[49] Sami Najafi-Asadolahi and Kristin Fridgeirsdottir. 2014. Cost-per-click pricing for display advertising. *Manufacturing & Service Operations Management* 16, 4 (2014), 482–497.

[50] Weitong Ou, Bo Chen, Xinyi Dai, Weinan Zhang, Weiwen Liu, Ruiming Tang, and Yong Yu. 2023. A survey on bid optimization in real-time bidding display advertising. *ACM Transactions on Knowledge Discovery from Data* 18, 3 (2023), 1–31.

[51] Weitong Ou, Bo Chen, Yingxuan Yang, Xinyi Dai, Weiwen Liu, Weinan Zhang, Ruiming Tang, and Yong Yu. 2023. Deep landscape forecasting in multi-slot real-time bidding. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 4685–4695.

[52] Matteo Pirotta, Marcello Restelli, Alessio Pecorino, and Daniele Calandriello. 2013. Safe policy iteration. In *Proceedings of the International Conference on Machine Learning.* PMLR, 307–315.

[53] Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. 2017. Parameter space noise for exploration. arXiv:1706.01905. Retrieved from https://arxiv.org/abs/1706.01905

[54] David Pollard. 2000. Asymptopia: An Exposition of Statistical Asymptotic Theory. Retrieved from http://www.stat.yale.edu/pollard/Books/Asymptopia

[55] Jiarui Qin, Weinan Zhang, Rong Su, Zhirong Liu, Weiwen Liu, Guangpeng Zhao, Hao Li, Ruiming Tang, Xiuqiang He, and Yong Yu. 2023. Learning to retrieve user behaviors for click-through rate estimation. *ACM Transactions on Information Systems* 41, 4 (2023), 1–31.

[56] Yanru Qu, Bohui Fang, Weinan Zhang, Ruiming Tang, Minzhe Niu, Huifeng Guo, Yong Yu, and Xiuqiang He. 2018. Product-based neural networks for user response prediction over multi-field categorical data. *ACM Transactions on Information Systems* 37, 1 (2018), 1–35.

[57] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International Conference on Machine Learning*. PMLR, 1889–1897.

[58] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. arXiv:1506.02438. Retrieved from https://arxiv.org/abs/1506.02438

[59] Pingzhong Tang, Xun Wang, Zihe Wang, Yadong Xu, and Xiwang Yang. 2020. Optimized cost Per mille in feeds advertising. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, 1359–1367.

[60] Magalie Thomassin, Thierry Bastogne, Alain Richard, and Antoine Libaux. 2003. A Bayesian approach for time-delay estimation of a managed river reach in imposed experimental conditions. *IFAC Proceedings Volumes* 36, 19 (2003), 299–304.

[61] Eiji Uchibe and Kenji Doya. 2007. Constrained reinforcement learning from intrinsic and extrinsic rewards. In *Proceedings of the IEEE 6th International Conference on Development and Learning*. IEEE, 163–168.

[62] Muhammad Uzair and Noreen Jamil. 2020. Effects of hidden layers on the efficiency of neural networks. In *Proceedings of the IEEE 23rd International Multitopic Conference (INMIC)*. IEEE, 1–6.

[63] Robert J. Vanderbei. 2020. *Linear Programming*. Springer.

[64] Márcia Peixoto Vega, Gabrielle Fontella de Moraes Oliveira, Lindoval Domiciano Fernandes, and André Leibsohn Martins. 2018. Monitoring and control strategies to manage pressure fluctuations during oil well drilling. *Journal of Petroleum Science and Engineering* 166 (2018), 337–349.

[65] Iryna Voronenko, Maryna Nehrey, Serhiy Kostenko, Iryna Lashchyk, and Viktoriia Niziaieva. 2021. Advertising strategy management in Internet marketing. *Journal of Information Technology Management* 13, Special Issue: Advanced Innovation Topics in Business and Management (2021), 35–47.

[66] Akifumi Wachi and Yanan Sui. 2020. Safe reinforcement learning in constrained Markov decision processes. In *Proceedings of the International Conference on Machine Learning*. PMLR, 9797–9806.

[67] Ping Wang, Yan Li, and Chandan K. Reddy. 2019. Machine learning for survival analysis: A survey. *ACM Computing Surveys* 51, 6 (2019), 1–36.

[68] Yu Wang, Jiayi Liu, Yuxiang Liu, Jun Hao, Yang He, Jinghe Hu, Weipeng P. Yan, and Mantian Li. 2017. Ladder: A human-level bidding agent for large-scale real-time online auctions. arXiv:1708.05565. Retrieved from https://arxiv.org/abs/1708.05565

[69] Zhoufutu Wen, Xinyu Zhao, Zhipeng Jin, Yi Yang, Wei Jia, Xiaodong Chen, Shuanglong Li, and Lin Liu. 2023. Enhancing dynamic image advertising with vision-language pre-training. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 3310–3314.

[70] Di Wu, Xiujun Chen, Xun Yang, Hao Wang, Qing Tan, Xiaoxun Zhang, Jian Xu, and Kun Gai. 2018. Budget constrained bidding by model-free reinforcement learning in display advertising. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 1443–1451.

[71] Youshao Xiao, Shangchun Zhao, Zhenglei Zhou, Zhaoxin Huan, Lin Ju, Xiaolu Zhang, Lin Wang, and Jun Zhou. 2023. G-Meta: Distributed meta learning in GPU clusters for large-scale recommender systems. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 4365–4369.

[72] Tian Xu, Ziniu Li, and Yang Yu. 2020. Error bounds of imitating policies and environments. *Advances in Neural Information Processing Systems* 33 (2020), 15737–15749.

[73] Qisong Yang, Thiago D. Simão, Simon H. Tindemans, and Matthijs T. J. Spaan. 2021. WCSAC: Worst-case soft actor critic for safety-constrained reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, 10639–10646.

[74] Xun Yang, Yasong Li, Hao Wang, Di Wu, Qing Tan, Jian Xu, and Kun Gai. 2019. Bid optimization by multivariable control in display advertising. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1966–1974.

[75] Xiao Yang, Daren Sun, Ruiwei Zhu, Tao Deng, Zhi Guo, Zongyao Ding, Shouke Qin, and Yanfeng Zhu. 2019. Aiads: Automated and intelligent advertising system for sponsored search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1881–1890.

[76] Shota Yasui, Gota Morishita, Fujita Komei, and Masashi Shibata. 2020. A feedback shift correction in predicting conversion rates under delayed feedback. In *Proceedings of the Web Conference*, 2740–2746.

[77] Dongjie Yu, Haitong Ma, Shengbo Li, and Jianyu Chen. 2022. Reachability constrained reinforcement learning. In *Proceedings of the International Conference on Machine Learning*. PMLR, 25636–25655.

[78] Shuai Yuan, Jun Wang, and Xiaoxue Zhao. 2013. Real-time bidding for online advertising: Measurement and analysis. In *Proceedings of the 7th International Workshop on Data Mining for Online Advertising*, 1–8.

[79] Yong Yuan, Feiyue Wang, Juanjuan Li, and Rui Qin. 2014. A survey on real time bidding advertising. In *Proceedings of IEEE International Conference on Service Operations and Logistics, and Informatics*. IEEE, 418–423.

[80] Dongxiang Zhang, Long Guo, Liqiang Nie, Jie Shao, Sai Wu, and Heng Tao Shen. 2017. Targeted advertising in public transportation systems with quantitative evaluation. *ACM Transactions on Information Systems* 35, 3 (2017), 1–29.

[81] Rui Zhang, Chengtian Jiang, Junbo Zhang, Jiteng Fan, Jiayi Ren, and Hui Xia. 2023. Reinvigorating sustainability in Internet of things marketing: Framework for multi-round real-time bidding with game machine learning. *Internet of Things* 24 (2023), 100921.

[82] Weinan Zhang, Yifei Rong, Jun Wang, Tianchi Zhu, and Xiaofan Wang. 2016. Feedback control of real-time display advertising. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining*, 407–416.

[83] Weinan Zhang, Shuai Yuan, and Jun Wang. 2014. Optimal real-time bidding for display advertising. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1077–1086.

[84] Xiao Zhang, Haonan Jia, Hanjing Su, Wenhan Wang, Jun Xu, and Ji-Rong Wen. 2021. Counterfactual reward modification for streaming recommendation with delayed feedback. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 41–50.

[85] Jun Zhao, Guang Qiu, Ziyu Guan, Wei Zhao, and Xiaofei He. 2018. Deep reinforcement learning for sponsored search real-time bidding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1021–1030.

[86] Xiangyu Zhao, Long Xia, Jiliang Tang, and Dawei Yin. 2019. "Deep reinforcement learning for search, recommendation, and online advertising: A survey" by Xiangyu Zhao, Long Xia, Jiliang Tang, and Dawei Yin with Martin Vesely as coordinator. *ACM Sigweb Newsletter* Spring (2019), 1–15.

[87] Fengtao Zhou and Hao Chen. 2023. Cross-modal translation and alignment for survival analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 21485–21494.

[88] Han Zhu, Junqi Jin, Chang Tan, Fei Pan, Yifan Zeng, Han Li, and Kun Gai. 2017. Optimized cost per click in taobao display advertising. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2191–2200.

Appendices

A Details of the Model Updating

In this section, we provide a detailed description of how to replace the objective and constraints with the surrogate functions in Equation (12).

First, we introduce the three bounds which will be used to replace the objective and constraints:

Lemma 1. *Through Pinsker's inequality [16]: for two arbitrary p, q, we have $D_{TV}(p||q) \leq \sqrt{D_{KL}(p||q)/2}$. Combining with Jensen's inequality, we have*:

$$E_{s\sim d^{\pi}}\left[D_{TV}(\pi'||\pi)[s]\right] \leq E_{s\sim d^{\pi}}\left[\sqrt{\frac{1}{2}D_{KL}(\pi'||\pi)[s]}\right] \leq \left[\sqrt{\frac{1}{2}E_{s\sim d^{\pi}}D_{TV}(\pi'||\pi)[s]}\right],$$

where d^{π} denotes the state distribution [72] of policy π, $D_{TV}(\pi'||\pi)[s]$ is the TV-divergence of policy π and π' over state s and $D_{KL}(\pi'||\pi)[s]$ is the KL-divergence of policy π and π' over state s.

Lemma 2. *Based on [1], for any two policies π' and π, with $\epsilon^{\pi'} \doteq \max_s |E_{a\sim\pi'}\left[A^{\pi}(s,a)\right]|$, we have*:

$$\begin{aligned} J(\pi') - J(\pi) &\geq \frac{1}{1-\gamma}E_{s\sim d^{\pi},a\sim\pi'}\left[A^{\pi}(s,a) - \frac{2\gamma\epsilon^{\pi'}}{1-\gamma}D_{TV}(\pi'||\pi)[s]\right] \\ &\geq \frac{1}{1-\gamma}E_{s\sim d^{\pi},a\sim\pi'}\left[A^{\pi}(s,a) - \frac{2\gamma\epsilon^{\pi'}}{1-\gamma}\sqrt{\frac{1}{2}E_{s\sim d^{\pi}}D_{TV}(\pi'||\pi)[s]}\right], \end{aligned}$$

where γ is the discount factor [36] and $A^{\pi}(s,a)$ denotes the advantage function [58] over state-action pair (s,a).

Lemma 3. *Based on [1], for any two policies π', π and cost function C, with $\epsilon_C^{\pi'} \doteq \max_s |E_{a\sim\pi'}[A_C^{\pi}(s,a)]|$, we have:*

$$J_C(\pi') - J_C(\pi) \leq \frac{1}{1-\gamma} E_{s\sim d^{\pi}, a\sim\pi'}\left[A_C^{\pi}(s,a) + \frac{2\gamma\epsilon_C^{\pi'}}{1-\gamma} D_{TV}(\pi'||\pi)[s]\right]$$
$$\leq \frac{1}{1-\gamma} E_{s\sim d^{\pi}, a\sim\pi'}\left[A_C^{\pi}(s,a) + \frac{2\gamma\epsilon_C^{\pi'}}{1-\gamma} \sqrt{\frac{1}{2} E_{s\sim d^{\pi}} D_{TV}(\pi'||\pi)[s]}\right],$$

where $A_{C_i}^{\pi}(s,a)$ denotes the advantage function of the cost function C_i [58] over state-action pair (s,a).

Based on the above lemmas, we can transform Equation (12) to

$$\pi_{k+1} = \arg\max_{\pi} \frac{1}{1-\gamma} E_{s\sim d^{\pi_k}, a\sim\pi}\left[A^{\pi_k}(s,a) - \frac{2\gamma\epsilon^{\pi_k}}{1-\gamma}\sqrt{\frac{1}{2} E_{s\sim d^{\pi_k}} D_{TV}(\pi||\pi_k)[s]}\right]$$
$$s.t. \quad J_C(\pi_k) + \frac{1}{1-\gamma} E_{s\sim d^{\pi_k}, a\sim\pi}\left[A_C^{\pi}(s,a) + \frac{2\gamma\epsilon_C^{\pi_k}}{1-\gamma}\sqrt{\frac{1}{2} E_{s\sim d^{\pi}} D_{TV}(\pi||\pi_k)[s]}\right] \leq 0$$
$$D_{KL}(\pi, \pi_k) \leq \delta.$$

Similar to trust region method [57], we set δ as a very small number, i.e., we update the policy in a small trust region, where we can assume $\delta \approx 0$ and γ is fixed discount factor, we can have:

$$\begin{aligned} \pi_{k+1} &= \arg\max_{\pi} E_{s\sim d^{\pi_k}, a\sim\pi}\left[A^{\pi_k}(s,a)\right] \\ s.t. \quad & J_C(\pi_k) + \frac{1}{1-\gamma} E_{s\sim d^{\pi_k}, a\sim\pi}\left[A_C^{\pi}(s,a)\right] \leq 0 \\ & D_{KL}(\pi, \pi_k) \leq \delta. \end{aligned} \tag{A1}$$

Then as shown in CPO method [1], if γ is small, the objective and cost function can be well-approximated by linearizing around π_k, and the KL-divergence constraint is well-approximated by second order expansion. The approximation to Equation (A1) is:

$$\begin{aligned} \theta_{k+1} &= \arg\max_{\theta} g^T(\theta - \theta_k) \\ s.t. \quad & c + b^T(\theta - \theta_k) \leq 0 \\ & \frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k) \leq \delta, \end{aligned} \tag{A2}$$

where g denotes the gradient of the objective, b denotes the gradient of the constraint, H is the Hessian of the KL-divergence which assumes to be positive-definite as in [1] and $c \doteq J_C(\pi_k)$. A dual to Equation (A2) can be

$$\max_{\nu\succeq 0, \lambda\geq 0} \quad \frac{-1}{2\lambda}(g^T H^{-1} g - 2r^T\nu + \nu^T S\nu) + \nu^T c - \frac{\lambda\delta}{2}, \tag{A3}$$

where $r \doteq g^T H^{-1} b$, $S \doteq b^T H^{-1} b$.

If there is at least one strictly feasible point, the optimal point θ satisfies

$$\theta_{k+1} = \theta_k + \frac{1}{\lambda^*} H^{-1}(g - b\nu^*),$$

where λ^* and ν^* is a solution to Equation (A3).

If there does not exist a feasible point, we directly update θ_k to decrease the cost function J_C under the KL-divergence constraint by:

$$\theta_{k+1} = \theta_k - \sqrt{\frac{2\delta}{b^T H^{-1} b}} H^{-1} b,$$

which is calculated based on [37] and [1].

B Implement Details

B.1 Bid-DC's Implementation

Bid-DC's bidding part adopts an actor-critic framework [2, 24, 39] and uses CPO to train the policy. The policy π and the critic networks V and V_C in CPO are all implemented as fully connected neural networks with 3 hidden layers [62]. The batch size is set to 128, discounted factor $\gamma = 1$, and the size of replay memory is set to 10,000. The learning rate of actor network is set to 3e-4. The learning rate for learning two critic networks are set to 1e-3, and the penalty is set to 5e-2. The target KL divergence [28, 54] in CPO is set to $\delta = 0.01$. The Bid-DC's delay part is a three-layer fully connected network, which takes the click's feature as its input and output the value of λ. The training batch size is set to 256 with a 1e-3 learning rate. The implementation of the baselines follows the original papers [27, 42, 59].

B.2 Offline Experiment's Implementation

The offline experiment is similar to the one used in USCB [27]. We use ad's data log to construct the offline experiments, which can be described in the following seven steps.

(1) Select an ad and collect all the bidding logs of the ad, i.e., all the bidding the ad participates in during its advertising.
(2) Sort all bidding logs based on its bidding time.
(3) For each bidding log, retrieve the bid value log corresponding to each position, i.e., How much bid is required to place the ad in each position in the ad list shown in Figure 1.
(4) Simulate the advertising process for the ad. For each bidding log, the ad submits a bid to determine the position observed by the user based on the bid log. Then, we simulate the user's click action based on the click model[9] and simulate the convert action based on the CVR prediction [18].
(5) We use the click log and conversion log to simulate the delay. For each bidding log, if both a click log and a conversion log are present, we set the interval time between them as the delay time for the corresponding click. If there is no conversion log for a click, we set the delay time to 0, indicating no delay.
(6) For every 15 minutes, the bidding model will change its state based on the current advertising state, e.g., current cost-per-conversion.
(7) At the end of advertising, we will check the performance of bidding as shown in Figure 1.

Received 26 April 2024; revised 18 September 2024; accepted 19 November 2024

[9] https://github.com/varepsilon/clickmodels