A PSO-based algorithm for reserve price optimization in online ad auctions

A PSO-based Algorithm for Reserve Price Optimization in Online Ad Auctions

Jason Rhuggenaath, Alp Akcay, Yingqian Zhang and Uzay Kaymak
School of Industrial Engineering, Eindhoven University of Technology
Eindhoven, The Netherlands
Email: {j.s.rhuggenaath, a.e.akcay, yqzhang, u.kaymak}@tue.nl

***Abstract*—One of the main mechanisms that online publishers use in online advertising in order to sell their advertisement space is the real-time bidding (RTB) mechanism. In RTB the publisher sells advertisement space via a second-price auction. Publishers can set a reserve price for their inventory in the second-price auction. In this paper we consider an online publisher that sells advertisement space and propose a method for learning optimal reserve prices in second-price auctions. We study a limited information setting where the values of the bids are not revealed and no historical information about the values of the bids is available. Our proposed method leverages the dynamics of particles in particle swarm optimization (PSO) to set reserve prices and is suitable for non-stationary environments. We also show that, taking the gap between the winning bid and second highest bid into account leads to better decisions for the reserve prices. Experiments using real-life ad auction data show that the proposed method outperforms popular bandit algorithms.**

I. Introduction

One of the main mechanisms that web publishers use in online advertising in order to sell their advertisement space is the real-time bidding (RTB) mechanism [1]. In RTB there are three main platforms: supply side platforms (SSPs), demand side platforms (DSPs) and an ad exchange (ADX) which connects SSPs and DSPs. The SSPs collect inventory of different publishers and thus serve the supply side of the market. Advertisers which are interested in showing online advertisements are connected to DSPs. When a user visits a webpage with an advertisement (ad) slot, the publisher sends a request to the ADX (via an SSP) indicating that an impression can potentially be displayed in this particular ad slot. At the same time, advertisers that are connected to DSPs send bid requests to the ADX indicating that they are willing to bid for this impression. A real-time auction then decides which advertiser is allowed to display its ad and the amount that the advertiser needs to pay. The most popular auction mechanism is the second-price auction, where the winning advertiser pays the second highest bid in the auction.

Publishers can set a reserve price for their inventory in the second-price auction. Due to the reserve price, all bids below the reserve price are disregarded, and as a consequence, there is a possibility that the ad slot is not sold. If the auction does have a winner, the winner pays the maximum of the second highest bid and the reserve price. In this paper we take the perspective of an online publisher that submits his inventory of advertisement space to an SSP and needs to decide on the optimal value of the reserve price. We assume that the publisher has limited information about the winning bid and second highest bid in the auction. More specifically, the publisher does not observe the actual values of the winning bid and second highest bid. After each sale attempt on the RTB market, the publisher only knows whether the sale was successful and the revenue that is received from that sale. This setting is relevant for publishers that are small and medium size enterprises (SMEs), since the ADX and the connected SSPs typically do not reveal the actual bids placed in the auction but only the result of the auction. Due to the limited feedback, the publisher faces an exploration-exploitation trade-off. He needs to experiment with different reserve prices to figure out which one works best, but at the same time, he does not want to explore too much since he wants to use the best reserve price as much as possible (exploitation). In this paper we present a method called PSO-ETC-RAP (PSO-based explore-then-commit algorithm with risk-aware pricing) that addresses the problem of the publisher. Compared to previous studies our method does not assume that the publisher observes the actual values of the winning bid and second highest bid. Our method is based on method from computational intelligence called particle swarm optimization (PSO). In PSO-ETC-RAP we cycle through four steps where, the first two steps are exploration phases, the third step is a commit phase, and the fourth step is a test phase. In the first step we estimate the value of the second highest bid and in the second step we estimate a reference value of the winning bid that represents a critical boundary: prices above this reference value tend to be too high and prices below the reference value tend to lead to successful sales of impressions. Our method exploits the evolution dynamics of particles in PSO to learn this critical boundary. In the third step the publisher uses the estimates for the winning and second highest bid to determine a reserve price. He subsequently commits to using this reserve price for a fixed amount of periods. In the fourth step, a test is done to see whether we should start another commit phase, or, go back and start with another exploration phase.

We summarize the main contributions of this paper as follows:

- We propose a method for learning optimal reserve prices in a limited information setting where the values of the bids are not revealed and no historical information about the values of the bids is available. Our proposed method is suitable for non-stationary environments.

- We show that taking the gap between the winning bid and second highest bid into account leads to better decisions for the reserve prices (we refer to this as risk-aware pricing).
- To the best of our knowledge, we are the first to consider leveraging techniques from PSO in order to set reserve prices in online ad auctions.
- Experiments using real-life ad auction data show that the proposed method outperforms popular bandit algorithms.

The remainder of this paper is organized as follows. In Section II we discuss the related literature. Section III provides a formal formulation of the problem. In Section IV we present the our proposed method for setting reserve prices. In Section V we perform experiments and compare our method with baseline strategies in order to assess the quality of our proposed method. Section VI concludes our work and provides some interesting directions for further research.

II. Related literature

The problem of maximizing revenues in online advertising has received increasing attention in the machine learning literature over the last decade. In [2] an online learning approach is used to derive a policy for setting optimal reserve prices, but the analysis makes the assumption that the environment is stationary. Other works such as [3], [4], [5], [6] use historical data to directly predict the optimal reserve price or the winning bid (which can indirectly be used to set a reserve price). A drawback of using models based on historical data is that they may not perform well in non-stationary environments. Most of the studies mentioned above do not set reserve prices in an adaptive way that adjusts to changing environments. Some studies such as [5], [7], [8] do study adaptive reserve prices, but they assume that the winning bid and/ or second highest bid are observed. In this paper we do not make this assumption. In [9] a different but related problem is studied, namely optimizing revenues for an SSP using a header-bidding strategy. Finally, we note that [10] studied a problem in a similar setting and proposed a parametric method based on Thompson sampling. In this paper we focus on reserve price optimization wihtin a single SSP, but there are also studies (e.g. [11], [12], [13], [14]) that consider optimization problems with multiple SSPs and guaranteed contracts.

This work is also related to a strand of the literature that combines PSO and evolutionary algorithms (EAs) to solve dynamic pricing problems, dynamic optimization problems in non-stationary environments, and problems with an exploration-exploitation trade-off. There are studies that use PSO in a dynamic pricing context to tune forecasting techniques in order to price stocks or options [15], [16], or to optimize a theoretical model [17]. In this paper we study a different problem, namely determining the right price when there is an exploration-exploitation trade-off. In [18] PSO is used for a stylized dynamic pricing problem with an exploration-exploitation trade-off. Compared with [18], we study a different setting (online auctions) and use real-life data to assess performance in non-stationary environments. Furthermore, in this paper, PSO is used in a different way: we use PSO as part of an exploration strategy in order to learn an unobserved quantity. In [19] an adaptive dynamic pricing strategy is designed for a duopoly where each firm has a finite inventory and wants to maximize revenue over a finite sales horizon. In [19] an EA is used to optimize the parameters of the pricing strategy offline. In our problem there is no offline phase and so the techniques developed in [19] are not applicable to our problem. In [20] a framework for EAs in online dynamic optimization problems is discussed and the framework is illustrated on an inventory management problem. In contrast to this paper, the framework requires the availability of a simulator in order to optimize the policies and so the methods presented in [20] are not directly applicable to our problem. Some studies [21], [22], [23] focus on optimization problems where the optima can shift over time. The main differences between our problem and the problems considered in these studies are that: (i) only one function evaluation is allowed in our setting (one reserve price each time period); (ii) after each function evaluation we get a noisy estimate of the objective value; (iii) in standard PSO problem settings the objective function is known and may shift over time, whereas in our setting there is no explicitly known objective function that can be used in order to directly maximize revenue.

To summarize, the main differences between this paper and previous works are that: (i) we show how to set adaptive reserve prices in possibly non-stationary environments; (ii) we do not assume that the publisher observes the top two bids in the ad auction, but only observes the revenue of each auction; (iii) we leverage techniques from PSO and use them as part of an exploration strategy in order to set reserve prices in online ad auctions.

III. Problem Statement

We consider a publisher that owns a single advertisement slot and that there is a sequence of impressions (corresponding to this advertisement slot) arriving over time. Time is discretized, and time periods are denoted by $t \in \mathbb{N}$. At the beginning of each time period (that is, upon arrival of an impression) the publisher has to decide on a reserve price $p_t \in [p_l, p_h] = [0, p_h]$. The prices p_l, p_h are the minimum and maximum reserve prices that are acceptable to the publisher. After deciding on a reserve price, the impression is offered for sale on the RTB-market via a Supply Side Platform (SSP). The SSP runs a second-price auction for the impression and the revenue of the publisher depends on the outcome of this auction. Let X_t and Y_t denote the highest and second highest bid respectively in the auction for the impression at time t. Then the revenue (or return) of the publisher at time t is given by $R_t = \mathbb{I}\{p_t \leq X_t\} \cdot \max\{Y_t, p_t\}$. Here $\mathbb{I}\{A\} = 1$ if A is true and $\mathbb{I}\{A\} = 0$ otherwise. The expression for R_t says that if the reserve price p_t is higher than the winning bid ($p_t > X_t$) then the publisher receives zero revenue. If the reserve price does not exceed the winning bid ($p_t \leq X_t$) then the revenue equals the maximum of the second highest bid and the reserve price. Note that, in general, the publisher does not observe the value of X_t and Y_t after a (successful) sale.

Assumption 1: We assume that all bids are non-negative, that is, $X_t > Y_t \geq 0$. If a sale was not successful, that is, if $p_t > X_t$, then the publisher does not observe X_t and Y_t. If a sale was successful, that is, if $p_t \leq X_t$, then the publisher observes $\max\{Y_t, p_t\}$. ■

Assumption 1 formalizes the setting that is relevant for SME publishers, since the ADX and the connected SSPs typically do not reveal the actual bids placed in the auction but only the result of the auction. The objective of the publisher is to maximize the cumulative revenue over the sales horizon of length T. Thus the revenue optimization problem over T time periods or impressions can be expressed as follows:

$$\max_{p_1,\ldots,p_T} \mathbb{E}\left\{\sum_{t=1}^{T} \mathbb{I}\{p_t \leq X_t\} \cdot \max\{Y_t, p_t\}\right\} \qquad (1)$$

Remark 1: In the literature on online advertising and the RTB-market, the reserve price is sometimes also referred to as the *floor price*. In the remainder of this paper we will use the term *top bid* to refer to the winning bid in the online auction and we will use the term *second bid* to refer to the second highest bid. ■

Remark 2: In order to simplify the exposition of our method, we focus on the case where there is a single ad slot. However, in practice, the publisher may want to set a reserve price depending on the characteristics of the user and the ad slot. Our method can also be applied in such a setting by, for example, making segments of users and applying our method for each segment. ■

IV. Proposed approach: PSO-ETC-RAP

In this section we present a method to sequentially set the reserve price p_t. We will refer to the method developed in this section as PSO-ETC-RAP (PSO based explore-then-commit algorithm with risk-aware pricing). It exploits the dynamic movements of particles in PSO to learn an estimate of the top bid. This estimate is then combined with an estimate of the second bid to set a reserve price.

Our method follows the following main steps:

1) Determine a reference value for the second bid.
2) Using the reference value for the second bid as a reference point, determine the reference value for of the top bid.
3) Apply a *risk-aware* reserve price for E periods.
4) If the environment is favorable, apply the risk-aware reserve price for another E periods. Otherwise, go to step 1.

Our method cycles through 4 steps. The first two steps are called exploration phases, the third step is called a commit phase. In the fourth step, a test is done to see whether we should start another commit phase, or, go back and start with another exploration phase. We now elaborate on each of these phases.

A. Exploration phase I: Reference value for the second bid

In the first exploration phase, the goal is to determine a *reference value* for the second bid. The reference value represents a value that is supposed to be close to the second bid. In order to determine the reference value for the second bid we can exploit the structure of the second-price auction. More specifically, there is a risk-free way to actually observe the second bid. It can be accomplished by setting a reserve price equal to zero: $p_t = 0$. If $p_t = 0$, then (by Assumption 1) the observed revenue will be equal to the second bid since $R_t = \max\{Y_t, p_t\}$. In order to estimate the second bid, we apply a reserve price of zero for M periods and use the average of the observed revenues as our reference value. That is, we set $p_k = 0$ for $k = t, t+1, \ldots, t+M-1$ and use $p^Y = \sum_{k=t}^{t+M-1} R_k/M$ as our reference value. Here M is a parameter of the algorithm that is chosen by the publisher.

B. Exploration phase II: Reference value for the top bid

In the second exploration phase, the goal is to determine a good reference value for the top bid. Determining a good reference value for the top bid is much more complicated since (by Assumption 1) the top bid is (in general) never observed. In Algorithm 1 we propose an estimator that performs well in our numerical experiments. The main idea behind the estimator in Algorithm 1 is as follows. The estimator takes as input a sample of K auction outcomes $\{(\mathbb{I}\{p_k \leq X_k\}, p_k)\}_{k=1}^{K}$ and returns a reference value p^X that represents a *critical boundary*. The interpretation of the critical boundary is that it is the highest reserve price such that reserve prices above p^X tend to lead to unsuccessful sales but reserve prices below p^X are successful.

Note that it is not immediately clear how to use Algorithm 1 since the values for p_k need to be chosen and the choices for these values will determine the quality and usefulness of the reference value that is obtained. In order to guide the process of choosing appropriate values for p_k, we exploit the way in which particles behave and evolve in PSO.

Algorithm 1 `Compute-Ref-X`

```
Require: A sample of $K$ reserve prices and outcomes $\{(\mathbb{I}\{p_k \leq X_k\}, p_k)\}_{k=1}^{K}$.
 1: Set $A^1 = \{p_k | \mathbb{I}\{p_k \leq X_k\} = 1,\ 1 \leq k \leq K\}$.
 2: Set $A^2 = \{p_k | \mathbb{I}\{p_k \leq X_k\} = 0,\ 1 \leq k \leq K\}$.
 3: if $A^1 = \varnothing$ then
 4:     Set $p^X = \min\{p |\ p \in A^2\}$.
 5: else if $A^2 = \varnothing$ then
 6:     Set $p^X = \max\{p |\ p \in A^1\}$.
 7: else
 8:     Set $p^X = \min\{\max\{p |\ p \in A^1\}, \min\{p |\ p \in A^2\}\}$.
 9: end if
10: return $p^X$.
```

C. Exploration phase II: Learning the reference value for the top bid using PSO

In this subsection we will show how PSO can be used to learn the reference value for the top bid. First we describe the standard formulation of PSO and then we describe our implementation.

The basic PSO algorithm was introduced by [24] and simulates the process of a flock of birds searching for food. The birds change their position and velocity constantly during the search process. Initially, the bird flock is scattered randomly, and later the flock will gradually gather as a group until they find food. In PSO each bird is represented by a particle and each particle has an associated location vector and velocity vector. The evolution of particle i is defined as follows:

$$V_{i,N} = \kappa V_{i,O} + c_1 r_1 (P_i - L_{i,O}) + c_2 r_2 (G - L_{i,O}) \tag{2}$$
$$L_{i,N} = L_{i,O} + V_{i,N} \tag{3}$$
$$V_{MIN} \leq V_{i,N} \leq V_{MAX} \tag{4}$$

Here $V_{i,N} \in \mathbb{R}^n$ denotes the new velocity of particle i, which is derived from: (i) the old velocity $V_{i,O} \in \mathbb{R}^n$; (ii) the old location $L_{i,O} \in \mathbb{R}^n$; (iii) the best solution found so far by particle i denoted by $P_i \in \mathbb{R}^n$; (iv) and the best global solution (found by any particle) denoted by $G \in \mathbb{R}^n$. The new location of particle i, denoted by $L_{i,N} \in \mathbb{R}^n$, is derived from its old location and new velocity. In Equation (2), c_1 and c_2 are referred to as acceleration coefficients; κ is called the inertia weight to balance global and local search; $r_1 \in \mathbb{R}^n$ and $r_2 \in \mathbb{R}^n$ with elements that are randomly and independently chosen numbers in $(0, 1)$. The movement of the particles is constrained by upper and lower bounds on the locations: $L_{MIN} \leq L_{i,N} \leq L_{MAX}$. The velocities are also constrained by (4).

We now discuss the modifications made to the basic PSO algorithm in our implementation. In our setting, there are S particles (the swarm size) and the location of each particle represents a reserve price. The particles move in a search neighborhood that is determined by the reference value for the second bid and a *search radius*. More specifically, if p_t^Y is the reference value available at time t, then the search neighborhood is:

$$L_{MIN} = p_t^Y \tag{5}$$
$$L_{MAX} = p_t^Y + W \tag{6}$$

Here $W > 0$ is the search radius. The value of G is given by the most recent value of p^X which is based on information from all S particles. The value of P_i is given by the most recent value of p^X but is only based on information from particle i. The update of the velocities is also slightly different in our implementation. The velocities are updated according to:

$$V_{i,N} = \kappa V_{i,O} + c_1 r_1 (P_i - L_{i,O}) + c_2^+ r_2 (G - L_{i,O}) \tag{7}$$
$$V_{i,N} = \kappa V_{i,O} + c_1 r_1 (P_i - L_{i,O}) + c_3^- r_2 (G - L_{i,O}) \tag{8}$$
$$V_{i,N} = \kappa V_{i,O} + c_1 r_1 (P_i - L_{i,O}) + c_4^+ r_2 (G - L_{i,O}) \tag{9}$$
$$V_{i,N} = \kappa V_{i,O} + c_1 r_1 (P_i - L_{i,O}) + c_5^- r_2 (G - L_{i,O}) \tag{10}$$

Since the locations of particles represent reserve prices, we have that $G, L_{i,O}, P_i, V_{i,O} \in \mathbb{R}$. There are four types of updates, and the type of update depends on (i) whether $(G - L_{i,O})$ is positive or negative; (ii) whether the sample used to determine G contained a reserve price $\bar{p}_k$ such that $\mathbb{I}\{\bar{p}_k \leq X_k\} = 0$. That is, the sample used to determine G contains an unsuccessful sale. If there is no unsuccessful sale while determining G, then (7) is used if $(G - L_{i,O}) > 0$ and (8) is used if $(G - L_{i,O}) < 0$ with $c_2^+ > c_3^-$. If there is an unsuccessful sale while determining G, then (9) is used if $(G - L_{i,O}) > 0$ and (10) is used if $(G - L_{i,O}) < 0$ with $c_4^+ < c_5^-$. By specifying appropriate values (the exact values are discussed in Section V) for the acceleration coefficients, the dynamics in (7) - (10) ensure that the locations of the particles get updated in the most informative direction relative to the value of G. In particular, (7) and (8) ensure that the locations $L_{i,O}$ are updated to explore values above G (if there are no unsuccessful sales), and (9) and (10) ensure that the locations $L_{i,O}$ are updated to explore values below G (if there are unsuccessful sales). The motivation for these dynamics comes from the definition of the critical boundary as specified in Algorithm 1. If the sample used to determine G contains an unsuccessful sale, then we know that the next best estimate for the reference value p^X will be at most G, and therefore the particles need to explore reserve prices lower than G. The opposite reasoning holds for the case where there is no unsuccessful sale while determining G.

The full procedure for the PSO step is described in Algorithm 2. In our implementation each particle in the swarm is used m times in a cyclic way. The main idea is to first initialize the particles in the search neighborhood and then use the initial locations as reserve prices in order to get in initial estimate for p^X. After each particle has been used once, we direct the particles in the direction that is most promising for learning the critical boundary by updating the velocities and locations

Algorithm 2 `PSO-Ref-X`

```
Require: $S$, $m$, $t_s$, $W$, $p^Y$, $\kappa$, $c_1$, $c_2^+$, $c_3^-$, $c_4^+$, $c_5^-$, $V_{MIN}$, $V_{MAX}$.
1: Set $t = t_s$.
2: Set $p_t^Y = p^Y$.
3: Initialize $L_{i,N}$ for $i = 1, \ldots, S$ according to (4), (5) and (6).
4: Set $B_i = \varnothing$ for $i = 1, \ldots, S$.
5: for $k = 1$ to $m$ do
6:     for $i = 1$ to $S$ do
7:         Set $t = t + 1$.
8:         if $k > 1$ then
9:             Update velocity $V_{i,N}$ and location $L_{i,N}$ of particle $i$ using (3),(4), (7)-(10).
10:        end if
11:        Set reserve price $p_t = L_{i,N}$.
12:        Observe outcome of auction: $\mathbb{I}\{p_t \leq X_t\}$.
13:        Set $B_i = B_i \cup \{(\mathbb{I}\{p_t \leq X_t\}, p_t)\}$.
14:        Run Compute-Ref-X (Algorithm 1) with input $\{(\mathbb{I}\{p_k \leq X_k\}, p_k)\}_{k=t_s}^t$ and $p^X$ as output.
15:        Set $G = p^X$.
16:        Run Compute-Ref-X (Algorithm 1) with input $B_i$ and $p_i^X$ as output.
17:        Set $P_i = p_i^X$.
18:    end for
19: end for
20: return $G$
```

(Line 9). In the end, each particle is used m times. Finally, we remark that in our setting, a function evaluation takes place every time that a particle is used for setting a reserve price, and that only one function evaluation is allowed (since only one reserve price is set) each period.

D. *Commit phase: Risk-aware pricing*

In the *commit phase* the publisher uses the reference values p_t^X and p_t^Y for the top bid and second bid in order to determine a reserve price p^*. The publisher then *commits* to using p^* for E periods, that is, $p_{t+1} = \cdots = p_{t+E} = p^*$. Here E is a parameter of the algorithm that is chosen by the publisher. There are in general many ways to select p^*. In this paper we present a simple scheme that performs well in our numerical experiments. The intuition behind the scheme is as follows: (i) if the gap between Y_t and X_t is believed to be large, then try to set a reserve price above the second bid (but not too high); (ii) if the gap between Y_t and X_t is believed to be small, then choose a reserve price close to Y_t.

In order to quantify the gap between Y_t and X_t we look at the reference values p_t^X and p_t^Y for the top bid and second bid. If $|p_t^X - p_t^Y|/p_t^Y \leq \alpha$ then we consider the gap to be small and the publisher sets the reserve price according to $p^* = p_t^Y$. If on the other hand $|p_t^X - p_t^Y|/p_t^Y > \alpha$, then the gap is considered to be large enough and the publisher sets a reserve price according to $p^* = \omega p_t^X + (1-\omega)p_t^Y$ for some $\omega \in (0,1)$. By controlling the parameters α and ω the publisher can decide the degree to which he wants to exploit the fact that the gap between Y_t and X_t is large.

We refer to this process as *risk-aware pricing* since the publisher explicitly takes information about both Y_t and X_t into account while setting reserve prices in order to reduce the risk of setting a reserve price that is too high. In principle, the publisher can choose to set reserve prices solely based on p_t^X, so why would he be interested in risk-aware pricing? The main motivation for risk-aware pricing is related to estimation error. Recall that the reference values are approximations that are based on realizations from the distribution of Y_t and X_t. The reference value p_t^X is an approximation for the critical boundary for the top bid, but this value still suffers from errors. In particular, it depends on (i) the sample of reserve prices used to determine its value, and (ii) the distribution of X_t. Assuming a fixed distribution for X_t, a different sample of reserve prices will lead to a different value of p_t^X. Furthermore, as the distribution of X_t might change over time, the value of p_t^X might not be representative for future time periods.

E. *Test phase: Monitoring the environment*

After using p^* for E periods, that is, after using $p_{t+1} = \cdots = p_{t+E} = p^*$ the publisher can use the observed revenues in order to determine whether he wants to start another commit phase. Suppose that $\{(\mathbb{I}\{p_k \leq X_k\}, R_k)\}_{k=t+1}^{t+E}$ is observed during the commit phase. The publisher can conduct a test to see whether the environment has changed substantially and use this test to determine whether it is worthwhile to start another commit phase, or, to go back and start another exploration phase. The full procedure for the test is described in Algorithm 3. The idea is to use a fraction $0 < f < 1$ to divide

Algorithm 3 `Test-Commit-Phase`

```
Require: A sample of size $E$ of revenues, auction outcomes
    $(\{\mathbb{I}\{p_k \leq X_k\}, R_k)\}_{k=1}^{E}$, the commit reserve price $p^*$, $f$, $\theta_S$,
    $d_R$, $d_S$, $\theta_L$, $\theta_H$.
    Calculate sample statistics.
 1: Set $n^* = \lfloor f \cdot E \rfloor$.
 2: Set $\mu_G = \sum_{k=1}^{E} R_k/E$. Set $\mu_B = \sum_{k=1}^{n^*} R_k/n^*$.
 3: Set $\mu_A = \sum_{k=n^*+1}^{E} R_k/(E-n^*)$.
 4: Set $\rho_G = \sum_{k=1}^{E} \mathbb{I}\{p_k \leq X_k\}/E$.
 5: Set $\rho_B = \sum_{k=1}^{n^*} \mathbb{I}\{p_k \leq X_k\}/n^*$. Set $\rho_A = 1 - \rho_B$.
 6: if $\rho_G \geq \theta_S$ then
 7:     Set $V$ = true.
 8:     Set $G = |\mu_A - \mu_B|/\min\{\mu_A, \mu_B\}$.
 9:     if $G \leq d_R$ then
10:         if $\mu_G > p^*$ then
11:             Set $V$ = false.
12:         end if
13:     else
14:         if $\mu_A > p^*$ then
15:             Set $V$ = false.
16:         end if
17:     end if
18: else
19:     if $|\rho_A - \rho_B| \leq d_S$ then
20:         Set $V$ = true.
21:         if $\rho_G \leq \theta_L$ then
22:             Set $V$ = false.
23:         end if
24:     else
25:         Set $V$ = false.
26:         if $\rho_A \geq \theta_H$ then
27:             Set $V$ = true.
28:         end if
29:     end if
30: end if
31: return $V$
```

the sample of observed results $\{(\mathbb{I}\{p_k \leq X_k\}, R_k)\}_{k=t+1}^{t+E}$ into two groups, where the first group uses the first $n^* = \lfloor f \cdot E \rfloor$ observations and the other group the remaining $E - n^*$ observations. Next, two main cases are distinguished: case (i) where the commit price was not close to the critical boundary and where most sales were successful; case (ii) where most of the sales are not successful and the commit price is most likely too high. In case (i) we start another commit phase, unless the observed revenues exceed the commit price, because this is an indication that the second bid will be higher than the commit price (in particular, it is expected to be higher than the previous reference value p_t^Y) in the future and that this commit price will be too low. In case (ii) we check how frequent the unsuccessful sales are. If the success rate in the two groups are similar and unsuccessful sales are very frequent, then we start another exploration phase. If the success rate in the two groups differ but in the last (most recent) group of observations the success rate meets a minimum requirement, then we start another commit phase. Finally, if there are N_{test} or more commit phases in a row, then we start an exploration phase with probability P_{test} (see Lines 27-42 in Algorithm 4).

F. Full algorithm

The details of the full procedure of PSO-ETC-RAP are described in Algorithm 4.

Algorithm 4 Pseudocode for PSO-ETC-RAP

```
Require: $M$, $E$, $\omega$, $\alpha$, $S$, $m$ , $W$, $\kappa$, $c_1$, $c_2^+$, $c_3^-$, $c_4^+$, $c_5^-$, $V_{MIN}$, $V_{MAX}$, $f$, $\theta_S$, $d_R$, $d_S$, $\theta_L$, $\theta_H$, $N_{test}$, $P_{test}$.
 1: Set $t = 0$. Set $N = 0$.
    Exploration Phase I.
 2: Set $A = \varnothing$.
 3: for  $j = 1$ to $M$  do
 4:     Set $t = t + 1$.
 5:     Apply reserve price $p_t = 0$.
 6:     Observe outcome of auction: $(\{\mathbb{I}\{p_t \leq X_t\}, R_t)$.
 7:     $A = A \cup \{R_t\}$
 8: end for
 9: Set $p^Y = \sum_{R \in A} R/M$.
    Exploration Phase II.
10: Set $t_s = t$.
11: Run PSO-Ref-X (Algorithm 2) with inputs $S$, $m$, $t_s$, $W$, $p^Y$, $\kappa$, $c_1$, $c_2^+$, $c_3^-$, $c_4^+$, $c_5^-$, $V_{MIN}$, $V_{MAX}$ and $G$ as output.
12: Set $p^X = G$.
    Risk-aware pricing.
13: if $|p^X - p^Y|/p^Y \leq \alpha$ then
14:     set $p^* = p^Y$
15: else if $|p^X - p^Y|/p^Y > \alpha$ then
16:     set $p^* = \omega p^X + (1-\omega)p^Y$
17: end if
18: Set $t = t + S \cdot m$.
19: Set $p_t = p^*$.
20: Set $B = \varnothing$.
21: for  $j = 1$ to $E$  do
22:     Set $t = t + 1$.
23:     Set reserve price $p_t$.
24:     Observe outcome of auction: $(\{\mathbb{I}\{p_t \leq X_t\}, R_t)$.
25:     Set $B = B \cup \{(\mathbb{I}\{p_t \leq X_t\}, R_t)\}$.
26: end for
    Test for change in environment.
27: Run Test-Commit-Phase (Algorithm 3) with inputs $B$, $p^*$, $t_s$, $f$, $\theta_S$, $d_R$, $d_S$, $\theta_L$, $\theta_H$ and $Q$ as output.
28: if $Q =$ true then
29:     Set $N = N + 1$.
30:     if  $N < N_{test}$ then
31:         Go to Line 20.
32:     else
33:         Draw $k$ from Bernoulli distribution with parameter $P_{test}$.
34:         if  $k = 1$ then
35:             Set $N = 0$ and go to Line 2.
36:         else
37:             Go to Line 20.
38:         end if
39:     end if
40: else
41:     Go to Line 2.
42: end if
```

V. Numerical Experiments

In this section we conduct experiments to evaluate the effectiveness of our proposed approach. The experiments are implemented in Python 2.7 and run on Intel(R) Core(TM) CPU i5-6300U @ 2.40GHz with 8GB RAM under Windows 7 environment.

A. Dataset Description

In order to evaluate our method we use real-life data from ad auction markets. We use the publicly available iPinYou dataset [25], which contains information from the perspective of nine advertisers on a DSP. It contains information about bids placed by advertisers on a DSP for impressions during a week. For each bid there is information about the ad slot (height, visibility, etc.), time of day, the ad exchange, and the result of the bid. It is important to note that the dataset only contains information about the top bid and the second bid if the advertiser actually wins the auction. As a consequence the bid records represent a biased sample from the distribution of the top bid and second bid. However, the dataset contains information for several advertisers and could still be used to get a general understanding of the dynamics on the ad auction market. We use the iPinYou dataset to construct synthetic data for the top bid and second bid in order to test our proposed approach. Note that the values of the bids are not revealed to any of the algorithms: they are only needed in order to test which methods yield the highest returns.

a) Construction of second bid: For a specific advertiser, we take the values of the second bid for the first 310000 impressions (sorted chronologically). We then divide these 310000 impressions into blocks with length 500. Within each block we sample with replacement 500 values of second bid from the 500 impressions in the block. After we are done with the sampling, we take rolling mean with window length of 25 observations of the resulting time series. Finally, we take the last 300000 values of the resulting time series as the values for the second bid. The main reason for taking a rolling mean is that the advertisers in this dataset are bidding on ad slots from different publishers (with different properties etc.), whereas we are interested in a single publisher that is selling a specific ad slot. By taking a rolling mean we are effectively extracting the general trend in the second bids.

b) Construction of top bid: In order to construct the top bid, we take the time series of the second bid and divide the time series into blocks with length B. Within each block we determine the maximum of the values of the second bid (denote the maximum by MB). The value of the top bid within a block is then equal to $MB \cdot (1+u) \cdot (1+v)$ where $v \sim \mathcal{U}(0.0, z)$. B, z and u are discrete uniformly distributed with $B \in \{50, 100\}$, $z \in \{0, 0.2, 0.4\}$ and $u \in \{0, 0.1, 0.2\}$. The draws of u are identically independently distributed (i.i.d) between blocks and draws of v are i.i.d within blocks. This construction models a situation where the gap between the top bid and second bid varies over time and is independent of the level of the second bid.

We use data from 4 advertisers and we repeat the above procedure 5 times for each advertiser in order to generate 5 time series for the top bid and second bid.

B. Benchmark Strategies

The MAB framework is a popular framework for decision making under exploration-exploitation trade-offs. In order to judge the quality of our proposed method, we compare its performance with two MAB algorithms: (i) the UCB algorithm [26] and (ii) the EXP3 algorithm [27]. These are popular bandit algorithms that are simple to implement and have satisfactory performance in a broad range of applications. In the case of i.i.d and bounded rewards for each arm, UCB achieves an order-optimal upperbound on cumulative regret. In the adversarial setting with bounded rewards, EXP3 achieves a worst-case order-optimal upperbound on cumulative regret.

C. Settings and Performance Metrics

In addition to PSO-ETC-RAP, we also consider PSO-ETC-TB. PSO-ETC-TB does not use a risk-aware price, instead it uses the reference value for the top bid as the commit price. In order to measure the performance of the methods, we consider four performance metrics. The first performance metric is the cumulative average return, which is defined as $\sum_{t=1}^{T} \hat{R}_t/T$, where $\hat{R}_t$ is the observed return in period t. This is our main metric to determine the profitability of a strategy. The second metric is the success rate, which is defined as $\sum_{t=1}^{T} \mathbb{I}\{p_t \leq X_t\}/T$. This measures how often reserve prices are set too high. The third metric is the revenue rate, which is defined as $\sum_{t=1}^{T} \mathbb{I}\{p_t \leq X_t\}\hat{R}_t/\sum_{t=1}^{T} X_t$. This measures the rate at which the top bid is extracted. The fourth metric is the revenue rate given success, which is defined as $\sum_{t=1}^{T} \mathbb{I}\{p_t \leq X_t\}\hat{R}_t/\sum_{t=1}^{T} \mathbb{I}\{p_t \leq X_t\}X_t$. This measures the rate at which revenue is extracted given that a sale is successful. We average the three performance metrics over the 5 samples constructed for each advertiser.

We use the following settings for PSO-ETC-RAP: $W = 50$, $E = 25$, $M = 15$, $S = 5$, $m = 3$, $\kappa = 0.45$, $c_1 = 1$, $c_2^+ = c_5^- = 2.0$, $c_3^- = c_4^+ = 1.05$, $f = 0.5$, $\theta_S = 0.8$, $d_R = 0.1$, $d_S = 0.1$, $\theta_L = 0.45$, $\theta_H = 0.8$, $V_{MIN} = -\infty$, $V_{MAX} = \infty$, $N_{test} = 3$, $P_{test} = 0.05$. With these choices $M = Sm = 15$ and so both exploration phases have the same duration. The choices for the acceleration coefficients, in combination with the dynamics in (7) - (10), ensure that the locations of the particles get updated in the most informative direction relative to the value of G. Note that, although the general PSO algorithm may also feature velocity clamping, we do not use it in PSO-ETC-RAP and PSO-ETC-TB. The main reason for this is that the locations of the particles are already restricted to a search radius and by imposing velocity clamping we would restrict exploration within this search radius. We use $p_l = 0$, $p_h = 250$ since the resulting time series of the top bid is at most 250.

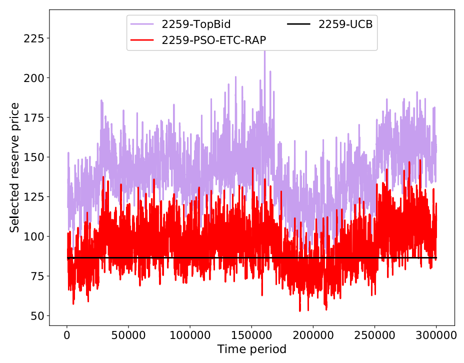
The following settings are used for the risk-aware pricing component in PSO-ETC-RAP: (i) If $|p^X - p^Y|/p^Y \leq 0.1$ then set the reserve price according to $p^* = p^Y$; (ii) If $0.1 < |p^X - p^Y|/p^Y \leq 0.2$, then set the reserve price according to $p^* = 0.5p^X + 0.5p^Y$; (iii) If $|p_t^X - p_t^Y|/p_t^Y > 0.2$, then set the reserve price according to $p^* = 0.7p^X + 0.3p^Y$. This models a situation were the gap between X_t and Y_t can be "small", "medium" and "large", and for larger gaps p^* is closer to p^X. In the UCB and EXP3 algorithms each arm represents a reserve price and we use $N = 100$ arms which are equally spaced in the interval $[p_l, p_h]$.

D. Results: PSO-ETC-RAP versus bandits

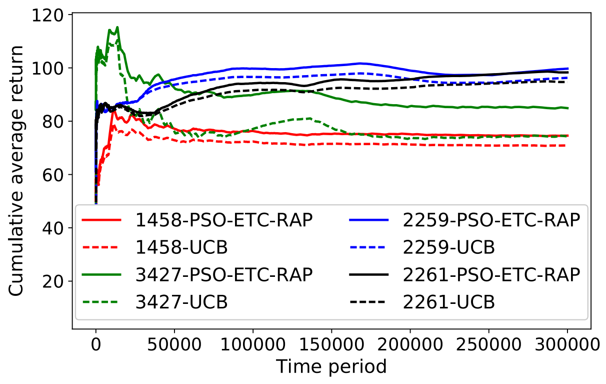
The results for PSO-ETC-RAP, PSO-ETC-TB, UCB and EXP3 are displayed in Table I. The results show that PSO-ETC-RAP generally outperforms the other methods. An interesting observation is that UCB outperforms EXP3, even though EXP3 makes no assumption on the sequence of returns. A similar finding was also reported in [9]. As UCB outperforms EXP3, the rest of this subsection focuses on the differences between UCB and PSO-ETC-RAP. The performance gap differs depending on the specific advertiser, but in general, PSO-ETC-RAP has a cumulative average return that is about 3.0 to 10.0 units higher than UCB. The main explanation for the superior performance of PSO-ETC-RAP compared to UCB is that (i) PSO-ETC-RAP is able to better track changes in the top bid and (ii) PSO-ETC-RAP is better in selecting reserve prices that are closer to the top bid when the gap between the top bid and second bid is large. In Fig. 1a a rolling average with window size 250 of the selected reserve prices by PSO-ETC-RAP and UCB relative to the top bid for a specific advertiser are shown (for a specific sample). From Fig. 1a we see that UCB tends to be more conservative and selects reserve prices that are often low, which results in a higher success rate but a lower revenue rate (see also Table I). Fig. 1a shows that UCB generally does not react very quickly to changes in the top bid. On the other hand, the reserve prices selected by PSO-ETC-RAP are generally higher than those selected by UCB and they tend to do a better job at tracking the changes in the top bid. Finally, in Fig. 1b we can see that it does not take long for PSO-ETC-RAP to outperform UCB within the sales horizon.

E. Results: Impact of risk-aware pricing

If we compare PSO-ETC-RAP with PSO-ETC-TB, then we see that PSO-ETC-RAP significantly outperforms PSO-ETC-TB. The performance gap differs depending on the specific advertiser, but in general, PSO-ETC-RAP has a cumulative average return that is at least 5.0 higher than PSO-ETC-TB. PSO-ETC-TB tends to select reserve prices that are higher then those selected by PSO-ETC-RAP, and this results in a lower success rate and a higher revenue rate given success (see Table I). The results show that estimation errors relating to the top bid can have a big impact on performance. The main reason for this is that, in this application, exceeding the top bid by an amount $\Delta > 0$ is more costly than selecting a reserve price Δ below the top bid. Taking the gap between the top and second bid into account leads to better performance.



(a) Reserve prices selected by PSO-ETC-RAP and UCB for advertiser 2259.



(b) Results for advertisers 1458, 3427, 2259 and 2261.

Fig. 1. Cumulative average returns and selected reserve prices.

TABLE I
PERFORMANCE OF PSO-ETC-RAP, PSO-ETC-TB, UCB AND EXP3 ON SYNTHETIC DATASETS.

	PSO-ETC-RAP			PSO-ETC-TB			UCB			EXP3		
advertiser 1458	min	max	mean	min	max	mean	min	max	mean	min	max	mean
cumulative average return	74.48	74.74	74.58	67.93	68.32	68.13	69.07	72.01	70.90	63.50	63.75	63.63
success rate	0.89	0.89	0.89	0.78	0.78	0.78	0.91	1.00	0.97	0.86	0.86	0.86
revenue rate	0.70	0.70	0.70	0.64	0.64	0.64	0.65	0.68	0.67	0.60	0.60	0.60
revenue rate success	0.77	0.77	0.77	0.79	0.79	0.79	0.65	0.72	0.68	0.68	0.68	0.68
advertiser 3427	min	max	mean	min	max	mean	min	max	mean	min	max	mean
cumulative average return	84.68	85.25	84.90	77.54	78.69	77.98	68.99	77.50	74.31	72.71	73.07	72.96
success rate	0.89	0.90	0.90	0.79	0.80	0.79	0.66	0.93	0.80	0.86	0.87	0.87
revenue rate	0.71	0.71	0.71	0.65	0.66	0.65	0.58	0.65	0.62	0.61	0.61	0.61
revenue rate success	0.78	0.78	0.78	0.79	0.79	0.79	0.68	0.77	0.73	0.69	0.69	0.69
advertiser 2259	min	max	mean	min	max	mean	min	max	mean	min	max	mean
cumulative average return	99.45	100.16	99.71	93.35	93.93	93.58	95.43	97.23	96.28	88.05	88.59	88.33
success rate	0.91	0.91	0.91	0.82	0.82	0.82	0.95	0.98	0.97	0.88	0.89	0.89
revenue rate	0.71	0.71	0.71	0.66	0.67	0.67	0.68	0.69	0.69	0.63	0.63	0.63
revenue rate success	0.77	0.77	0.77	0.78	0.79	0.79	0.68	0.72	0.70	0.69	0.70	0.69
advertiser 2261	min	max	mean	min	max	mean	min	max	mean	min	max	mean
cumulative average return	98.11	98.45	98.29	91.84	92.67	92.32	92.17	96.08	94.70	86.95	87.25	87.13
success rate	0.90	0.91	0.90	0.81	0.82	0.82	0.94	0.99	0.97	0.89	0.90	0.90
revenue rate	0.70	0.70	0.70	0.66	0.66	0.66	0.66	0.69	0.68	0.62	0.62	0.62
revenue rate success	0.76	0.76	0.76	0.78	0.78	0.78	0.67	0.71	0.69	0.68	0.68	0.68

Remark 3: We have presented results based on reasonable settings that performed well across all advertisers. The results for the other 5 advertisers in the iPinYou dataset are very similar to those reported in Table I and are omitted due to space limitations. Results with $S \in \{3, 8\}$, $W = 75$, $M = 25$ are also similar. Performance can be improved by tuning the parameters for each advertiser separately. Also, the risk-aware pricing component can be refined to improve performance. In our experiments we did not conduct extensive parameter tuning. One approach to conduct paramter tuning is to use a percentage of the impressions (say the first 10%) to tune parameters and then evaluate the performance on the remaining impressions. ■

VI. Discussion and Conclusion

We proposed a method for learning reserve prices in second-price auctions. We studied a limited information setting where the values of the bids are not revealed and no historical information about the values of the bids is available. Our method leverages the movement of particles in PSO to set reserve prices. To the best of our knowledge, we are the first to consider leveraging techniques from PSO in order to set reserve prices in online ad auctions.

Our results indicate that it is important to properly deal with non-stationarity in the distribution of the bids. Although there are algorithms based on the theory of multi-armed bandits that have performance guarantees, our results indicate that using multi-armed bandit strategies is not enough to guarantee high performance. Another key insight is that incorporating knowledge about the structure of the problem (e.g. in the form of risk-aware pricing) can lead to improved performance.

Our approach can be improved in various ways. One drawback of our appoach is that is takes a number of control parameters as input. Future work can be directed towards using evolutionary algorithms and swarm intelligence to optimize these parameters. For example, evolutionary algorithms and swarm intelligence can be employed to optimize the parameters of the risk-aware pricing component and to make it adaptive and self-regulatory.

References

[1] J. Wang, W. Zhang, and S. Yuan, "Display advertising with real-time bidding (RTB) and behavioural targeting," *Foundations and Trends® in Information Retrieval*, vol. 11, no. 4-5, pp. 297–435, 2017. [Online]. Available: http://dx.doi.org/10.1561/1500000049

[2] N. Cesa-Bianchi, C. Gentile, and Y. Mansour, "Regret minimization for reserve prices in second-price auctions," *IEEE Transactions on Information Theory*, vol. 61, no. 1, pp. 549–564, Jan 2015.

[3] Z. Xie, K.-C. Lee, and L. Wang, "Optimal reserve price for online ads trading based on inventory identification," in *Proceedings of the ADKDD'17*, ser. ADKDD'17. New York, NY, USA: ACM, 2017, pp. 6:1–6:7. [Online]. Available: http://doi.acm.org/10.1145/3124749.3124760

[4] M. R. Rudolph, J. G. Ellis, and D. M. Blei, "Objective variables for probabilistic revenue maximization in second-price auctions with reserve," in *Proceedings of the 25th International Conference on World Wide Web*, ser. WWW '16. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2016, pp. 1113–1122. [Online]. Available: https://doi.org/10.1145/2872427.2883051

[5] S. Yuan, J. Wang, B. Chen, P. Mason, and S. Seljan, "An empirical study of reserve price optimisation in real-time bidding," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '14. New York, NY, USA: ACM, 2014, pp. 1897–1906. [Online]. Available: http://doi.acm.org/10.1145/2623330.2623357

[6] M. Mohri and A. M. n. Medina, "Learning algorithms for second-price auctions with reserve," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2632–2656, Jan. 2016. [Online]. Available: http://dl.acm.org/citation.cfm?id=2946645.3007027

[7] D. Austin, S. Seljan, J. Monello, and S. Tzeng, "Reserve price optimization at scale," in *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, Oct 2016, pp. 528–536.

[8] P. Chahuara, N. Grislain, G. Jauvion, and J.-M. Renders, "Real-time optimization of web publisher RTB revenues," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '17. New York, NY, USA: ACM, 2017, pp. 1743–1751.

[9] G. Jauvion, N. Grislain, P. Dkengne Sielenou, A. Garivier, and S. Gerchinovitz, "Optimization of a SSP's header bidding strategy using Thompson sampling," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '18. New York, NY, USA: ACM, 2018, pp. 425–432. [Online]. Available: http://doi.acm.org/10.1145/3219819.3219917

[10] J. Rhuggenaath, A. Akcay, Y. Zhang, and U. Kaymak, "Optimizing reserve prices for publishers in online ad auctions," in *IEEE Conference on Computational Intelligence for Financial Engineering and Economics (CIFEr)*, 2019, to appear.

[11] K. Salomatin, T.-Y. Liu, and Y. Yang, "A unified optimization framework for auction and guaranteed delivery in online advertising," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, ser. CIKM '12. New York, NY, USA: ACM, 2012, pp. 2005–2009.

[12] R. R. Afshar., Y. Zhang., M. Firat., and U. Kaymak., "A reinforcement learning method to select ad networks in waterfall strategy," in *Proceedings of the 11th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART,*, INSTICC. SciTePress, 2019, pp. 256–265.

[13] R. Refaei Afshar, Y. Zhang, M. Firat, and U. Kaymak, "A decision support method to increase the revenue of ad publishers in waterfall strategy," in *IEEE Conference on Computational Intelligence for Financial Engineering and Economics (CIFEr)*, 2019, to appear.

[14] S. R. Balseiro, J. Feldman, V. Mirrokni, and S. Muthukrishnan, "Yield optimization of display advertising with ad exchange," *Management Science*, vol. 60, no. 12, pp. 2886–2907, 2014.

[15] M. E. Abdual-Salam, H. M. Abdul-Kader, and W. F. Abdel-Wahed, "Comparative study between differential evolution and particle swarm optimization algorithms in training of feed-forward neural network for stock price prediction," in *2010 The 7th International Conference on Informatics and Systems (INFOS)*, March 2010, pp. 1–8.

[16] F. Wang, P. L. H. Yu, and D. W. Cheung, "Complex stock trading strategy based on particle swarm optimization," in *2012 IEEE Conference on Computational Intelligence for Financial Engineering Economics (CIFEr)*, March 2012, pp. 1–6.

[17] B. Wang, "Dynamic pricing model for flight based on passenger behavior choice and its solution of particle swarm optimization," in *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing*, Oct 2008, pp. 1–4.

[18] P. B. Mullen, C. K. Monson, K. D. Seppi, and S. C. Warnick, "Particle swarm optimization in dynamic pricing," in *2006 IEEE International Conference on Evolutionary Computation*, July 2006, pp. 1232–1239.

[19] S. Ramezani, P. A. N. Bosman, and H. La Poutre, "Adaptive strategies for dynamic pricing agents," in *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Volume 02*, ser. WI-IAT '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 323–328. [Online]. Available: https://doi.org/10.1109/WI-IAT.2011.193

[20] P. A. N. Bosman and H. La Poutré, "Learning and anticipation in online dynamic optimization with evolutionary algorithms: The stochastic case," in *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '07. New York, NY, USA: ACM, 2007, pp. 1165–1172. [Online]. Available: http://doi.acm.org/10.1145/1276958.1277187

[21] T. T. Nguyen, S. Yang, and J. Branke, "Evolutionary dynamic optimization: A survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 6, pp. 1 – 24, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2210650212000363

[22] R. Liu, J. Li, J. fan, C. Mu, and L. Jiao, "A coevolutionary technique based on multi-swarm particle swarm optimization for dynamic multi-objective optimization," *European Journal of Operational Research*, vol. 261, no. 3, pp. 1028 – 1051, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221717302709

[23] C. Cruz, J. R. González, and D. A. Pelta, "Optimization in dynamic environments: a survey on problems, methods and measures," *Soft Computing*, vol. 15, no. 7, pp. 1427–1448, Jul 2011. [Online]. Available: https://doi.org/10.1007/s00500-010-0681-0

[24] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, Nov 1995, pp. 1942–1948 vol.4.

[25] W. Zhang, S. Yuan, J. Wang, and X. Shen, "Real-time bidding benchmarking with iPinYou dataset," *arXiv preprint arXiv:1407.7073*, 2014.

[26] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, no. 2, pp. 235–256, May 2002. [Online]. Available: https://doi.org/10.1023/A:1013689704352

[27] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. Schapire, "The nonstochastic multiarmed bandit problem," *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2002. [Online]. Available: https://doi.org/10.1137/S0097539701398375