# Learning to Design Coupons in Online Advertising Markets

Weiran Shen
Tsinghua University
Beijing, China
emersonswr@gmail.com

Pingzhong Tang
Tsinghua University
Beijing, China
kenshinping@gmail.com

Xun Wang
Tsinghua University
Beijing, China
wxhelloworld@outlook.com

Yadong Xu
Tsinghua University
Beijing, China
xuyd17@mails.tsinghua.edu.cn

Xiwang Yang
ByteDance
Beijing, China
yangxiwang@bytedance.com

## ABSTRACT

Coupon has been a major marketing tool that promotes sales for new and repeated buyers and proven effective in numerous realistic scenarios. In this paper, we mainly focus on the problem of designing coupons to maximize the revenue in second-price auction.

Firstly, we derive the dominant strategies of bidders if they are provided with coupons in second-price auction and prove that the revenue optimization problem with coupons for all the bidders is NP-complete. Secondly, we cast the problem of designing coupons to maximize revenue into a learning framework. With well-designed loss functions, we perform theoretical analysis of its properties and propose corresponding algorithms to solve the problem. Finally, with both synthetic data and industrial data, extensive experiments are conducted to demonstrate their effectiveness.

## KEYWORDS

Coupon; Online advertising; Second-price auction

**ACM Reference Format:**
Weiran Shen, Pingzhong Tang, Xun Wang, Yadong Xu, and Xiwang Yang. 2020. Learning to Design Coupons in Online Advertising Markets. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 9 pages.

## 1 INTRODUCTION

Advertising has become a dominant source of revenue for Internet companies such as Google, eBay and Baidu. The ads are usually sold through auctions, where advertisers bid and compete with one another for ad slots. In practice, both the VCG (Vickrey–Clarke–Groves) [10, 15, 38] mechanism and the GSP (generalized second-price) [12] mechanism are widely used. In fact, as argued by Akbarpour and Li [1], most real-world auctions are variants on just a few canonical formats: the first-price auction, the ascending auction, and (more recently) the second-price auction. The popularity of second-price auction relies on the fact that it is incentive compatible, i.e., bidders bid exactly what they are willing to pay. Revenue maximization in second-price-based auctions has been intensively studies recently[17, 23, 30]. However, in this paper, we study this problem from a different dimension by designing coupon policies.

Coupons have been a major marketing tool that promotes sales for new and repeated buyers and proven effective in numerous realistic scenarios [5, 24, 31]. It works by offering a financial discount off the regular price when purchasing a product. Coupon can be viewed as a price discrimination device actually [7, 39]. By offering a coupon, the seller can attract some buyer who are otherwise more inclined to buy a competing brand. By setting different prices for customers of his own and that of a competing brand, the seller can increase his own market share, i.e., lose small to win big. Similarly, the intuition behind using coupon in second-price auctions is to incentives low-values advertisers to overbid. Thus, by pricing high-valued advertisers and low-valued ones differently, the platform can increase his revenue. As far as we know, this work is the first to study revenue maximization in second-price auctions using coupons.

There are many devices to maximize revenue. For example, by setting reserve prices [17]. A reserve price is the minimum amount that an advertiser has to pay if he wins. It works by excluding some conditions where a high-valued advertiser pays a low price; Another well-known method is squashing [20]. In actual auctions, advertiser $i$'s bid would be adjusted by a "quality score" $w_i$ (e.g., Click-through rate or CTR). Squashing mechanisms set a parameter $\alpha$ which transforms $w_i$ into $w_i^\alpha$; Recently, the boost mechanism is also proposed by Golrezaei et al. [14]. A boost is a transformation from original bids to boosted bids. With a slight modification on payments, it ensures that no advertisers would pay more than their original bids. It works by boosting low-valued advertisers so the winner has to pay more.

However, coupon mechanisms are quite different from these methods. It can be regarded as a decentralized revenue improvement tool and hand the right to use coupons over to advertisers. While for the other methods, they can be regarded as centralized tools since the platform can decide whether to use them. The former is more advertiser-friendly. Some recent works also consider auctions from the angle of the advertisers, e.g., [1] studies credible auction where the auctioneer has the incentive to follow the rules and does not want to cheat on advertisers.

Although finding revenue optimal auctions in theory has been widely studied [25, 29], we propose a learning approach to handle the problem of determining coupons to optimize revenue for practical reasons. On the one hand, current theoretic results usually rely

*Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand.* 

on knowing each advertisers' value distribution while even good approximations of such distributions are not known in practice; On the other hand, learning methods usually generalize well to new advertisers.

Our contributions can be summarized as follows:

- We model the problem of designing coupons in second-price auctions and cast the problem into a learning framework;
- We derive the dominant strategy for each bidder and study the properties of optimal coupons;
- In the no-feature case, we specify the loss function and propose a heuristic algorithm successively optimize the coupon of selected bidder while fixing other coupons;
- In the general case, we design a surrogate loss function for optimizing the coupons. When the predictor is linear, we propose an algorithm that combines the DC programming and the heuristic algorithm to train the predictor;
- Using both synthetic data and industrial data, extensive experiments are conducted to demonstrate the effectiveness of our algorithms.

### 1.1 Additional related works

There is rich literature on revenue maximization in advertising system [22, 25, 28, 36]. However, these results heavily depend on the knowledge of the bidders' valuation distributions and could be very sensitive to estimation errors [29]. Simpler mechanisms have been widely used in industry, such as second-price auctions and the GSP auctions. To increase the revenue in these auctions, many methods have been proposed, i.e., reserve pricing [17], squashing [20], and boost [14] (see [19] for a comprehensive survey). However, reserve price has a strong negative effect on the number of advertisements shown. Increasing the reserve price may make the auction less attractive and fewer advertisers will bid which lead to lower revenue [27]. The other methods make a transformation on bids. When advertisers have to pay more (e.g., increase the revenue of platform), they may doubt the platform is "cheating" and refuse these transformation if they can.

Another closely related line of works include using machine learning in mechanism design. Balcan et al. [4] formulates mechanism design problems as algorithm design problems; Golowich et al. [13], Shen et al. [34] try to design revenue maximization auctions via deep learning; Medina and Mohri [23] and Shen et al. [32] provide learning methods for setting reserve prices in second-price auctions. A special case of our analysis coincides with the no-feature scenario considered by Cesa-Bianchi et al. [9], Derakhshan et al. [11]. Besides, reinforcement learning is also used to build behavior model from data and then applied in automated mechanism design [8, 33, 35]. Many other works also belong to this category [2, 6, 26].

## 2 PRELIMINARIES

Let $N$ be the set of bidders with $|N| = n$. Each bidder $i$ has a private value $v_i$, which is drawn from a publicly known distribution $F_i$, with the corresponding density function $f_i$. For convenience, let $\boldsymbol{v} = (v_1, v_2, \ldots, v_n)$ be the value profile, and $v_{-i}$ be the value profile of all bidders except $i$. Besides, we use $\boldsymbol{F} = F_1 \times F_2 \times \cdots \times F_n$ to represent the joint distribution of value profile $\boldsymbol{v}$.

### 2.1 Second-price auction with coupons

We first describe how coupons work. Before collecting bids from the bidders, the auctioneer announces to each bidder *privately* that he can provide coupon $c_i \geq 0$ to the bidder. This coupon can get $c_i$ off the original payment if bidder $i$ wins. We can set $c_i = 0$ if the auctioneer does not provide bidder $i$ with any coupons. In such situations, bidder $i$ may change his original bidding strategy and bid $b_i$ instead. Let $\boldsymbol{b}$ and $b_{-i}$ be the bid profile of all bidders and all bidders except $i$, respectively. In response to these bids, the auctioneer determines which bidder is the winner and how much he needs to pay according to some allocation rule and payment rule. Specifically, the allocation rule is a function $a$ that maps the bid profile $\boldsymbol{b}$ to an $n$-dimensional vector indicating the quantity of items allocated to each bidder, i.e. $a : \mathbb{R}^n \mapsto [0,1]^n$. In addition, the payment rule is a function $p : \mathbb{R}^n \mapsto \mathbb{R}^n$ that takes the bid profile $\boldsymbol{b}$ as input and outputs an $n$-dimensional non-negative vector specifying the payment for each bidder. For simplicity, we use $a(\boldsymbol{b})$ and $a$, $p(\boldsymbol{b})$ and $p$ interchangeably. In the vanilla second-price auction, the allocation rule and the payment rule are:

$$a(\boldsymbol{b}) \in \arg\max_{\bar{a}} \sum_{i=1}^{n} \bar{a}_i b_i, \tag{1}$$

$$p_i(\boldsymbol{b}) = \max_{\bar{a}} \sum_{j \neq i} \bar{a}_j b_j - \sum_{j \neq i} a_j b_j, \tag{2}$$

where $\bar{a}_i$ is a binary variable which satisfies that $\bar{a}_i \in \{0,1\}$, $\sum_{i=1}^{n} \bar{a}_i \leq 1$. These equations indicate that the advertiser with the highest bid wins and he pays the second highest bid. As for the second-price auction with coupons, the allocation rule is still Equation (1) while the payment rule becomes Equation (3) instead.

$$p_i^c(\boldsymbol{b}) = p_i(\boldsymbol{b}) - c_i a_i(\boldsymbol{b}) \tag{3}$$

Therefore, the utility of bidder $i$ is:

$$u_i(\boldsymbol{b}) = v_i a_i(\boldsymbol{b}) - p_i^c(\boldsymbol{b}) = (v_i + c_i) a_i(\boldsymbol{b}) - p_i(\boldsymbol{b}) \tag{4}$$

Finally, note that the bidding strategy of bidder $i$ depends on his value and the coupon provided to him, i.e. $b_i = b_i(v_i; c_i)$, the revenue of the auctioneer is ultimately decided by the value profile and the coupon, that is,

$$Rev(\boldsymbol{v}; \boldsymbol{c}) = \sum_{i=1}^{n} p_i(\boldsymbol{b}) - c_i a_i(\boldsymbol{b}).$$

*The intuition behind the coupon idea is that the bidder with the second highest bid could place a higher bid when presented with a coupon.* In what follows, let $(j)$ denote the order of bidders that satisfies: if $j < l$, then $[b_{(j)} > b_{(l)}] \vee [(b_{(j)} = b_{(l)}) \wedge (c_{(j)} \leq c_{(l)})]$ holds, i.e., $(j)$ denotes the bidder with the $j$-th highest bid (if there are more than one, then select the one with smallest coupon). Let $[j]$ denote the index of the bidder that satisfies: if $j < l$, then $v_{[j]} > v_{[l]}$ holds, i.e., $[j]$ denotes the bidder with the $j$-th highest value.

### 2.2 Learning formulation

Our goal is to learn coupons to maximize the expected revenue. To implement the second-price auction with coupons, we apply a data-driven approach to optimize the coupon given history data. Since the second-price auction is a truthful mechanism, we can directly use these historical bids as bidders' values. Let $v_i^t$ be the

value of bidder $i$ in auction $t$, where $v_i^t$ is drawn from $F_i$. Besides, in auction $t$, let $\boldsymbol{v}^t$ and $v_{-i}^t$ denote the value profile of all bidders and the value profile of all bidders except bidder $i$. Furthermore, we consider a generic feature space $\mathcal{X}$ with a label space $\mathcal{B} = \mathbb{R}_+^n$ consisting of the value profile $\boldsymbol{v}$. The corresponding feature vector with value profile $\boldsymbol{v}^t$ in auction $t$ is $\boldsymbol{x}^t$ and $(\boldsymbol{x}^t, \boldsymbol{v}^t)$ is drawn from some known joint distribution on $\mathcal{X} \times \mathcal{B}$. To formulate the coupon design problem in a learning context, we optimize the coupons in the training dataset to achieve good revenue performance on separate testing data which is drawn from the same distribution as the training data. Let $c_i^t$ represent the coupon for bidder $i$ in auction $t$, we consider the following two cases:

*The no-feature case.* In this case, there are no features associated with the auction and the bidders. Thus the coupons provided to the bidders in each auction remain the same, i.e. $c_i^t = c_i, \forall t$. In such case, the objective is to calculate the coupons $\boldsymbol{c} = (c_1, \ldots, c_n)$ that minimize the empirical loss:

$$\mathcal{L}_S(\boldsymbol{c}) = -\frac{1}{T}\sum_{t=1}^{T} Rev\left(\boldsymbol{v}^t; \boldsymbol{c}\right), \tag{5}$$

where the training data is $S = \left(\boldsymbol{v}^1, \ldots, \boldsymbol{v}^T\right)$.

*The general case.* In this case, the coupons provided to the bidders in auction $t$ depend on the feature vector $\boldsymbol{x}^t$. We use a hypothesis function $h : \mathcal{X} \mapsto \mathbb{R}^n$ to set the coupon $\boldsymbol{c} = h(\boldsymbol{x})$, hence the objective is to select a hypothesis function $h$ out of some hypothesis set $H$ to minimize the corresponding empirical loss:

$$\mathcal{L}_S(h) = -\frac{1}{T}\sum_{t=1}^{T} Rev\left(\boldsymbol{v}^t; h(\boldsymbol{x}^t)\right), \tag{6}$$

where $S = \left((\boldsymbol{x}^1, \boldsymbol{v}^1), \ldots, (\boldsymbol{x}^T, \boldsymbol{v}^T)\right)$ is the training data.

It is clear that Equation (5) is in fact a special case of Equation (6) since we can generate identical feature vectors $\boldsymbol{x}^t$ over the course of $T$ auctions. As a concrete example, the training data could consist of value profiles, and features about the bidders like their budgets, ad context, industry categories and so on. Then the coupon design problem is equivalent to predicting a coupon for each bidder in order to improve the revenue of the auctioneer.

# 3 THEORETICAL ANALYSIS

In this section, we analyze the problem from the perspective of the mechanism design theory. We first analyze the dominant bidding strategy for each bidder. Then, we give some simple but intuitive properties of the coupon mechanism.

## 3.1 Dominant bidding strategies

In this section, we analyze the equilibrium of the game induced by the coupon mechanism, and show that there is a dominant strategy for each bidder. Here, a dominant strategy is a strategy that always provides at least the same utility (i.e., Equation (4)) to the bidder, no matter what the other bidders' strategies are.

Theorem 3.1. *In the second-price auction with coupons, the dominant strategy of bidder $i$ is to use the coupon $c_i$ and bid $v_i + c_i$.*

Proof. Since $c_i \geq 0$, bidder $i$ always gets a positive value off his payment compared with not using the coupon. Thus using $c_i$ is always better.

As for "bidding $v_i + c_i$", we prove it by contradiction. Let $\boldsymbol{b}$ be the bid profile where $b_i = v_i + c_i$ and $\bar{\boldsymbol{b}}$ be another bid profile where $b_i \neq \bar{b}_i$ while $b_{i'} = \bar{b}_{i'}$ for all $i' \neq i$. Suppose that $u_i(\boldsymbol{b}) < u_i(\bar{\boldsymbol{b}})$, then we have:

$$u_i(\boldsymbol{b}) = (v_i + c_i)a_i(\boldsymbol{b}) + \sum_{j\neq i} a_j(\boldsymbol{b})b_j - \max_{\bar{a}} \sum_{j\neq i} \bar{a}_j b_j$$

$$u_i(\bar{\boldsymbol{b}}) = (v_i + c_i)a_i(\bar{\boldsymbol{b}}) + \sum_{j\neq i} a_j(\bar{\boldsymbol{b}})b_j - \max_{\bar{a}} \sum_{j\neq i} \bar{a}_j b_j$$

According to Equation (1), we have:

$$a(\boldsymbol{b}) \in \arg\max_{\bar{a}} \left\{ (v_i + c_i)\bar{a}_i(\boldsymbol{b}) + \sum_{j\neq i} \bar{a}_j(\boldsymbol{b})b_j \right\}.$$

Thus:

$$(v_i + c_i)a_i(\boldsymbol{b}) + \sum_{j\neq i} a_j(\boldsymbol{b})b_j \geq (v_i + c_i)a_i(\bar{\boldsymbol{b}}) + \sum_{j\neq i} a_j(\bar{\boldsymbol{b}})b_j.$$

A Contradiction. □

## 3.2 Properties of the coupon mechanism

The properties stated by Proposition 3.2 make the coupon design problem challenging.

Proposition 3.2. *Given the value profile $\boldsymbol{v}$, $Rev(\boldsymbol{v}; \boldsymbol{c})$, as a function of coupons $\boldsymbol{c}$, has the following properties:*

- *It is not continuous, and is neither convex nor concave;*
- *It has infinitely many optimal solutions.*

Proof. Consider the example with two bidders. The value of bidder 1 is 1 while the value of bidder 2 is 0. Then $Rev(\boldsymbol{v}; \boldsymbol{c})$ is not continuous and is neither convex nor concave.

Let $\boldsymbol{c}^*$ be the optimal coupons of the above example. Then $\boldsymbol{c}^* + y$ is also optimal where $y > 0$ and $\boldsymbol{c}^* + y = (c_1^* + y, c_2^* + y)$. Thus there are infinite optimal solutions. □

However, it is actually without loss of generality to focus on the case with at least one coupon being 0:

Proposition 3.3. *For any value profile, there exists an optimal coupon profile $\boldsymbol{c}^*$ that satisfies $\exists i, c_i^* = 0$.*

# 4 THE NO-FEATURE CASE

In this section, we focus on the no-feature case. We first show that finding the optimal coupons in this case is computationally hard. Then, we propose a heuristic algorithm for solving the problem. We also give some structural results in this setting.

## 4.1 Hardness results

Based on Theorem 3.1, the revenue of the auctioneer for a single auction is:

$$Rev(\boldsymbol{v}; \boldsymbol{c}) = v_{(2)} + c_{(2)} - c_{(1)}. \tag{7}$$

Combining Equation (5) and (7) gives:

$$\mathcal{L}_S(\boldsymbol{c}) = -\frac{1}{T}\sum_{t=1}^{T} v_{(2)}^t + c_{(2)}^t - c_{(1)}^t. \tag{8}$$

Theorem 4.1. *In the second-price auction with coupons, the coupon optimization problem in the no-feature case is NP-complete.*

Proof. The result follows from a reduction from the edge bipartization problem, which is NP-complete [16]. The edge bipartization problem is the problem of deleting as few edges as possible to make a graph bipartite. Given a graph $G = (V, E)$, we build the following instance of the coupon optimization problem based on the edge bipartization problem: There are $n$ bidders where $n-1 = |V|$ and $T = 4|E|$ auctions. Let $H$ and $L$ be constants such that $L < H < \frac{2|E|}{2|E|-1}L$. And each edge $e = (i, j)$ represents 4 auctions:

- In the first two auctions: $v_i = v_j = L, v_n = H$. The values of other bidders are all 0.
- In the third auction: $v_i = v_n = H$. The values of other bidders are all 0.
- In the fourth auction: $v_j = v_n = H$. The values of other bidders are all 0.

In this instance, coupons must belong to $\{0, H - L\}$. Before proving that the solution of the coupon optimization problem is equivalent to finding as few edges as possible to make $G$ bipartite, we first describe the structure of the solution of the coupon optimization problem. Given an edge $e = (i, j)$. If bidder $i$ and bidder $j$ are provided with the same coupon, then the revenue will be $2H + 2L$; If one of them has a coupon with value of $H - L$ and the other has a coupon with value of 0, then the revenue will be $3H + L > 2H + 2L$.

Now, we provide different bidders with different coupons arbitrarily. Let $S_A$ be the set of bidders with coupon whose value is $H - L$ and $S_B$ be the rest. Assume that there is no edge within $S_A$ and no edge within $S_B$. That is, each edge connects a vertex in $S_A$ and a vertex in $S_B$. Thus, $G$ is a bipartite graph whose partition has the parts $S_A$ and $S_B$. In that case, the total revenue is $|E|(3H + L)$, which is the maximum possible revenue. When there are $k$ edges within $S_A$ and $S_B$, i.e., there are $k$ edges do not connect $S_A$ and $S_B$, which denotes that we have $k$ edges $(i, j)$ such that bidder $i$ and bidder $j$ are provided with coupons with the same value. In that case, the revenue is $(|E| - k)(3H + L) + k(2H + 2L)$, so we prefer to minimize $k$. In other words, the solution of the coupon optimization problem is obtained by solving the edge bipartization problem. □

## 4.2 Heuristic algorithm

Since there exists no algorithm that can compute the optimal coupons in polynomial time even in the no-feature case, we propose a heuristic algorithm. In this algorithm, instead of optimizing all the coupons simultaneously, we successively optimize one of the coupons while fixing all other coupons. To be specific, we optimize $c_i$ while fixing $c_{-i}$ in each iteration.

**Algorithm 1** Coupon optimization in the no-feature case

1: Initialize $\boldsymbol{c} \leftarrow \mathbf{0}$.
2: **while** not terminated **do**
3:  **for** $i \in N$ **do**
4:  $c_i \leftarrow \arg\min_y \mathcal{L}_S(y; c_{-i}) + \lambda(y - c_i)^2$.
5:  **end for**
6:  Update $c_i$ simultaneously to $c_i - \min_j c_j$.
7: **end while**

Here the algorithm terminates if convergence is reached or after some fixed number of iterations. The term $\lambda(y - c_i)^2$ in line 4 is used as a regularizer to guarantee that the coupon does not change too much in each iteration. Besides, line 6 is used to ensure that the minimum value of the coupons $\boldsymbol{c}$ is zero, which ultimately imposes restriction on the divergence of coupons according to the third property in Proposition 3.3.

In fact, Algorithm 1 can be extended to become a more general solution framework:

- In line 3, we can select a subset of bidders to optimize.
- $c_i$ in line 4 can be updated according to other functions.

## 4.3 Solution for the no-feature case

Algorithm 1 updates the coupon for one bidder at a time. Thus the original problem is reduced to the following one-dimensional optimization problem that minimizes:

$$\mathcal{L}_S(y; c_{-i}) + \lambda(y - c_i)^2. \tag{9}$$

To begin with, we introduce some new notations. For a single auction instance, let $L(y; \boldsymbol{v}, c_{-i}) = -Rev(\boldsymbol{v}; (y, c_{-i}))$. Similar to the aforementioned notation, we use $b_{-i,j} = v_{-i,j} + c_{-i,j}$ to denote the reported bid of bidder $j$ and use $(j)$ to indicate the order of bidders except bidder $i$. Again this order ensures that if $j < l$, then $[b_{-i,(j)} > b_{-i,(l)}] \vee [(b_{-i,(j)} = b_{-i,(l)}) \wedge (c_{-i,(j)} \leq c_{-i,(l)})]$ holds. Therefore, $L(y; \boldsymbol{v}, c_{-i})$ can be written as:

$$L(y; \boldsymbol{v}, c_{-i}) = \begin{cases} -b_{-i,(2)} + c_{-i,(1)} & y \leq b_{-i,(2)} - v_i \\ -v_i - y + c_{-i,(1)} & b_{-i,(2)} - v_i < y < b_{-i,(1)} - v_i \\ \min\left\{-v_{-i,(1)}, -v_i\right\} & y = b_{-i,(1)} - v_i \\ -b_{-i,(1)} + y & y > b_{-i,(1)} - v_i \end{cases} \tag{10}$$

If $b_{-i,(1)} > v_i$, then $L(y; \boldsymbol{v}, c_{-i})$ achieves its minimum at $y = b_{-i,(1)} - v_i$. Otherwise, $L(y; \boldsymbol{v}, c_{-i})$ achieves its minimum at $y = 0$. We use $(b_{-i,(1)} - v_i)^+ = \max\{b_{-i,(1)} - v_i, 0\}$ to denote this point in both cases. Furthermore, in terms of Equation (9), Lemma 4.2 indicates that the optimal coupon for bidder $i$ lies in some finite set with cardinality $O(T)$.

Lemma 4.2. *The problem that minimizes Equation* (9) *admits a solution $y^*$ that belongs to the following set*

$$\{0\} \cup \left\{\left(b^t_{-i,(1)} - v^t_i\right)^+\right\}_{t=1}^{T} \cup \boldsymbol{Q},$$

*where $\boldsymbol{Q}$ is the set of stationary points of the objective function, which satisfies $|\boldsymbol{Q}| \leq T + 1$.*

Proof. Equation (9) is piecewise continuous, thus the solution of the objective function belongs to its discontinuity and stationary points. It is worth noting that $\{0\} \cup \{(b^t_{-i,(1)} - v^t_i)^+\}_{t=1}^{T}$ consists of all the discontinuity points. Besides, the derivative of the objective function is piecewise linear and there are at most $T+1$ pieces, hence the objective function has at most $T + 1$ stationary points. □

Proposition 4.3. *According to Lemma 4.2, we can complete Algorithm 1 by computing the minimum of Equation* (9)*, which can be implemented by evaluating its objective function at $O(T)$ points and returning the best one.*

## 5 THE GENERAL CASE

In terms of coupon optimization for the no-feature case, the previous section proposes a solution framework based on heuristic algorithm and specifies the corresponding loss function (Equation (10)). This approach motivates us to optimize the coupon of one bidder at a time in the general case as before. Note that the original feature space represents the feature of all the bidders and the hypothesis function maps this feature space to all the coupons, which is inconsistent with the approach. Thus it is straightforward to divide the original feature space into different parts, and each part is associated with related bidder, that is, the feature vector of bidder $i$ is $\boldsymbol{x}_i$. What's more, we use a new hypothesis function $h_i$ to set the coupon for bidder $i$ as $c_i = h_i(\boldsymbol{x}_i)$. *Since $h_i$ is identical for each bidder, we actually learn the predictor that works for any advertiser with any features (including new advertisers).* Here we only use this notation to specify the hypothesis function for bidder $i$, which can make the subsequent content in this section more explicit.

### 5.1 The loss function

Different from the no-feature case, $c_i^t = h_i(\boldsymbol{x}_i^t)$ varies in different auctions in the general case. Hence the approach that directly evaluates coupons from some candidate set does not work as well. In this section, we analyze the loss function for the general case, which is fundamental to coupon optimization.

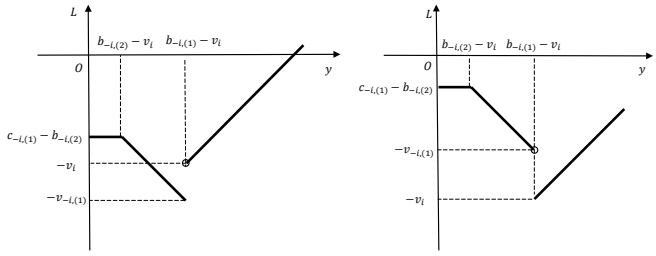


**Figure 1: The loss function for fixed $\boldsymbol{v}$ and $c_{-i}$ where $v_i < b_{-i,(2)}$. (a) case where $v_i < v_{-i,(1)}$; (b) case where $v_{-i,(1)} < v_i$.**

As shown in Figure 1(a), for any fixed $\boldsymbol{v}$ and $c_{-i}$ that satisfy $v_i < b_{-i,(2)}$ and $v_i < v_{-i,(1)}$, $L(y; \boldsymbol{v}, c_{-i})$ is not differentiable at two points, $y = b_{-i,(2)} - v_i$ and $y = b_{-i,(1)} - v_i$, and is discontinuous at $y = b_{-i,(1)} - v_i$. Besides, it is neither convex nor concave, but quasi-convex in fact. Similarly, the above properties still hold in the case where $v_i > v_{-i,(1)}$. Although we can solve quasi-convex optimization problems, a sum of quasi-convex functions (i.e. $\mathcal{L}_S(y; c_{-i})$) does not maintain the quasi-convexity property and can have many local minima. Thus we need to consider some surrogate loss function, which we refer to as $L^Y(y; \boldsymbol{v}, c_{-i})$. *The intuition behind $L^Y$ is to smooth the original discontinuous loss function (Equation* (10)*).* Given $\boldsymbol{v}$ and $c_{-i}$, since the real loss function $L(y; \boldsymbol{v}, c_{-i})$ has different shape in different cases, we define $L^Y(y; \boldsymbol{v}, c_{-i})$ case by case.

(1) When $v_i \geq b_{-i,(1)}$, providing coupon for bidder $i$ yields less revenue. In this case, the loss function is unchanged.

$$L^Y(y; \boldsymbol{v}, c_{-i}) = L(y; \boldsymbol{v}, c_{-i}) = -b_{-i,(1)} + y$$

(2) When $v_i < b_{-i,(1)}$ and $v_i \leq v_{-i,(1)}$, let $d_{i,2} = b_{-i,(2)} - v_i$, $d_{i,1} = b_{-i,(1)} - v_i$ and $e_i = v_{-i,(1)} - v_i$. We simply connect two points

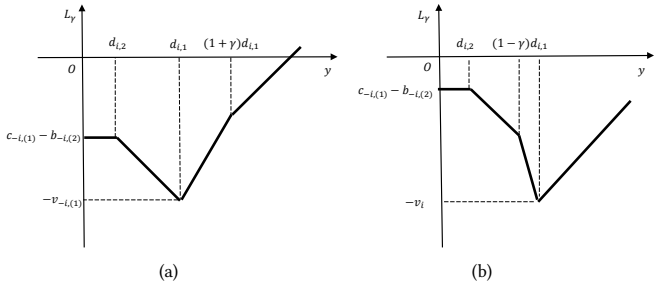


**Figure 2: The surrogate loss function for fixed $\boldsymbol{v}$ and $c_{-i}$ where $v_i < b_{-i,(1)}$. (a) case where $v_i < v_{-i,(1)}$; (b) case where $v_{-i,(1)} < v_i$.**

on $L(y; \boldsymbol{v}, c_{-i})$, i.e. $(d_{i,1}, -v_{-i,(1)})$ and $((1+\gamma)d_{i,1}, -b_{-i,(1)} + (1+\gamma)d_{i,1})$, and define

$$L^Y(y; \boldsymbol{v}, c_{-i}) = \begin{cases} -b_{-i,(2)} + c_{-i,(1)} & y \leq d_{i,2} \\ -v_i - y + c_{-i,(1)} & d_{i,2} < y \leq d_{i,1} \\ \left(1 + \frac{e_i}{\gamma d_{i,1}}\right)(y - d_{i,1}) - v_{-i,(1)} & d_{i,1} < y \leq (1+\gamma)d_{i,1} \\ -b_{-i,(1)} + y & y > (1+\gamma)d_{i,1} \end{cases}$$

Note that when $v_i \geq b_{-i,(2)}$, the curve can be obtained by translation from Figure 2(a).

(3) As shown in Figure 2(b), when $v_i < b_{-i,(1)}$, $v_i > v_{-i,(1)}$ and $(1-\gamma)d_{i,1} \geq d_{i,2}$, we can connect $((1-\gamma)d_{i,1}, c_{-i,(1)} - v_i - (1-\gamma)d_{i,1})$ and $(d_{i,1}, -v_i)$ on $L(y; \boldsymbol{v}, c_{-i})$, and then define

$$L^Y(y; \boldsymbol{v}, c_{-i}) = \begin{cases} -b_{-i,(2)} + c_{-i,(1)} & y \leq d_{i,2} \\ -v_i - y + c_{-i,(1)} & d_{i,2} < y \leq (1-\gamma)d_{i,1} \\ \left(-1 + \frac{e_i}{\gamma d_{i,1}}\right)(y - d_{i,1}) - v_i & (1-\gamma)d_{i,1} < y \leq d_{i,1} \\ -b_{-i,(1)} + y & y > d_{i,1} \end{cases}$$

(4) When $v_i < b_{-i,(1)}$, $v_i > v_{-i,(1)}$ and $(1-\gamma)d_{i,1} < d_{i,2}$, the two points become $((1-\gamma)d_{i,1}, c_{-i,(1)} - b_{-i,(2)})$ and $(d_{i,1}, -v_i)$, and the surrogate function can be defined as

$$L^Y(y; \boldsymbol{v}, c_{-i}) = \begin{cases} -b_{-i,(2)} + c_{-i,(1)} & y \leq (1-\gamma)d_{i,1} \\ \frac{d_{i,2} - c_{-i,(1)}}{\gamma d_{i,1}}(y - d_{i,1}) - v_i & (1-\gamma)d_{i,1} < y \leq d_{i,1} \\ -b_{-i,(1)} + y & y > d_{i,1} \end{cases}$$

Note that $L^Y$ is a lower bound for $L$ in all the cases. After defining $L^Y(y; \boldsymbol{v}, c_{-i})$, we can directly obtain the corresponding empirical loss in the no-feature case, denoted by $\mathcal{L}_S^Y(y, c_{-i})$. We use the following example to show the difference between $L^Y$ and $L$.

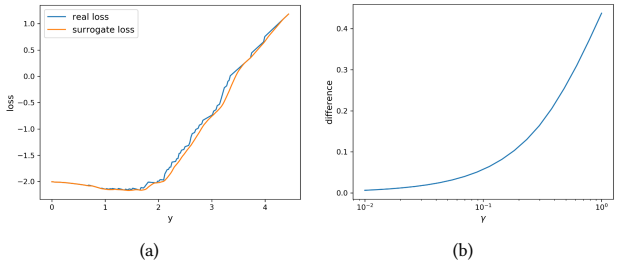


**Figure 3: Difference between $\mathcal{L}_S$ and $\mathcal{L}_S^Y$. (a) $\mathcal{L}_S$ and $\mathcal{L}_S^Y$ as the function of coupon $y$ when $\gamma = 0.09$; (b) average difference between $\mathcal{L}_S$ and $\mathcal{L}_S^Y$ as the function of $\gamma$.**

*Example 5.1.* Suppose there are 5 bidders, and the value of bidder $i$ is drawn from a uniform distribution on $[0, i]$. Let $c_1 = 0.8, c_3 = 0.4, c_4 = 0.2, c_5 = 0$, we simulate $T = 50$ auctions and compute the loss function as the function of the coupon for bidder 2. As Figure 3(a) shows, $\mathcal{L}_S^\gamma$ is a lower bound for $\mathcal{L}_S$, and the difference between two functions is relatively small. Besides, Figure 3(b) indicates that we can approach the real loss function as we select a sufficiently small $\gamma$.

In the general case, since we separate the feature space and hypothesis set, the real empirical loss can be rewritten as

$$\mathcal{L}_S(h_i; h_{-i}) := \frac{1}{T}\sum_{t=1}^{T} L\left(h_i(\boldsymbol{x}_i^t); \boldsymbol{v}^t, h_{-i}(\boldsymbol{x}_{-i}^t)\right),$$

for any $h_i \in H_i$ given $h_{-i}$. Similarly, we can define $\mathcal{L}_S^\gamma(h_i; h_{-i})$ as the empirical loss with respect to $L^\gamma$. We further analyze the difference of the expectations of $L^\gamma$ and $L$. We use

$$\mathcal{L}(h_i; h_{-i}) := \mathbb{E}_{\boldsymbol{x},\boldsymbol{v}}[L(h_i(\boldsymbol{x}_i); \boldsymbol{v}, h_{-i}(\boldsymbol{x}_{-i}))],$$

for any $h_i \in H_i$ given $h_{-i}$, and analogously we can define $\mathcal{L}^\gamma(h_i; h_{-i})$. It is obvious that $\mathcal{L}^\gamma(h_i; h_{-i})$ is still a lower bound for $\mathcal{L}(h_i; h_{-i})$. Similar to the results by Medina and Mohri [23], we get:

Theorem 5.2. *Given $h_{-i}$, let $H_i$ be a closed, convex subset of a linear space of functions containing 0 and $\overline{h}_i^\gamma$ be the solution of $\min_{h_i \in H_i} \mathcal{L}^\gamma(h_i; h_{-i})$. Then if $\sup_{\boldsymbol{v},\boldsymbol{x}_{-i}} |v_i - \max\{v_{-i} + h_{-i}(\boldsymbol{x}_{-i})\}| = Z < \infty$, the difference between $\mathcal{L}^\gamma$ and $\mathcal{L}$ is bounded by:*

$$\mathcal{L}\left(\overline{h}_i^\gamma; h_{-i}\right) - \mathcal{L}^\gamma\left(\overline{h}_i^\gamma; h_{-i}\right) \le O(\gamma)Z$$

Proof sketch. We prove Theorem 5.2 case by case (with respect to the definition of $L^\gamma$). In each case, we partition the whole set into different regions, and the loss function $L$ and $L^\gamma$ are affine in each region. Then using the similar technique in [23], we prove the corresponding inequality by segment amplification and minification in each case. □

Denote by $\overline{h}_i$ the solution of $\min_{h_i \in H_i} \mathcal{L}(h_i; h_{-i})$, the following inequality is straightforward:

$$\begin{aligned}
0 &\le \mathcal{L}\left(\overline{h}_i^\gamma; h_{-i}\right) - \mathcal{L}\left(\overline{h}_i; h_{-i}\right) \\
&= \mathcal{L}\left(\overline{h}_i^\gamma; h_{-i}\right) - \mathcal{L}^\gamma\left(\overline{h}_i^\gamma; h_{-i}\right) + \mathcal{L}^\gamma\left(\overline{h}_i^\gamma; h_{-i}\right) - \mathcal{L}\left(\overline{h}_i; h_{-i}\right) \\
&\le O(\gamma)Z + \mathcal{L}^\gamma\left(\overline{h}_i, h_{-i}\right) - \mathcal{L}\left(\overline{h}_i, h_{-i}\right) \le O(\gamma)Z,
\end{aligned}$$

where the last inequality holds because $\mathcal{L}^\gamma$ is a lower bound for $\mathcal{L}$. This inequality means if we choose a sufficiently small $\gamma$, then we can obtain a solution that is near optimal for $\mathcal{L}(h_i; h_{-i})$ by minimizing the loss function $\mathcal{L}^\gamma(h_i, h_{-i})$.

## 5.2 The DC algorithm

As mentioned in [23], suppose the separate hypothesis set $H_i$ consists of linear functions with bounded norm, i.e. $\boldsymbol{x}_i \mapsto \boldsymbol{\omega}_i \cdot \boldsymbol{x}_i, \|\boldsymbol{\omega}_i\| \le \Lambda$. Then for a fixed $\gamma > 0$, we expect to solve the following optimization problem:

$$\min_{\|\boldsymbol{\omega}_i\| \le \Lambda} \quad \sum_{t=1}^{T} L^\gamma\left(\boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t; \boldsymbol{v}^t, c_{-i}^t\right) \quad \text{s.t. } \boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t \ge 0,\ \forall t, \qquad (11)$$

where $\boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t = h_i(\boldsymbol{x}_i^t)$ and $c_{-i}^t = h_{-i}(\boldsymbol{x}_{-i}^t)$. Constraints $\boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t \ge 0$ are used to ensure that coupons are non-negative. Although each term in the sum may be not convex, Problem (11) can be formulated as a DC programming problem. DC programming problems denote the set of programming problems which can be represented as a difference of two convex functions. It is a widely used technique in the learning literature. Many theoretical results, applications, and algorithms for this interesting and important class of programming problems have been studied [3, 18, 21, 37]. The only thing we need to do is to decompose $L^\gamma(y; \boldsymbol{v}, c_{-i})$ as the difference of two convex functions, i.e. $g_1(y; \boldsymbol{v}, c_{-i}) - g_2(y; \boldsymbol{v}, c_{-i})$. Here we omit the superscript $t$ for simplicity. Since $L^\gamma$ varies in different cases, we derive two convex functions $g_1$ and $g_2$ case by case.

(1) When $v_i \ge b_{-i,(1)}$, it is easy to derive $g_1(y; \boldsymbol{v}, c_{-i}) = -b_{-i,(1)} + y$ and $g_2(y; \boldsymbol{v}, c_{-i}) = 0$.

(2) When $v_i < b_{-i,(1)}$ and $v_i \le v_{-i,(1)}$, we can define

$$g_1(y; \boldsymbol{v}, c_{-i}) = \begin{cases} -v_i - y + c_{-i,(1)} & y \le d_{i,1} \\ \left(1 + \frac{e_i}{\gamma d_{i,1}}\right)(y - d_{i,1}) - v_{-i,(1)} & y > d_{i,1} \end{cases}$$

$$g_2(y; \boldsymbol{v}, c_{-i}) = \begin{cases} -v_i - y + b_{-i,(2)} & y \le d_{i,2} \\ 0 & d_{i,2} < y \le (1+\gamma)d_{i,1} \\ \frac{e_i}{\gamma d_{i,1}}\left(y - (1+\gamma)d_{i,1}\right) & y > (1+\gamma)d_{i,1} \end{cases}$$

(3) When $v_i < b_{-i,(1)}$, $v_i > v_{-i,(1)}$ and $(1-\gamma)d_{i,1} \ge d_{i,2}$, $g_1$ and $g_2$ are defined as

$$g_1(y; \boldsymbol{v}, c_{-i}) = \begin{cases} \left(-1 + \frac{e_i}{\gamma d_{i,1}}\right)(y - d_{i,1}) - v_i & y \le d_{i,1} \\ -b_{-i,(1)} + y & y > d_{i,1} \end{cases}$$

$$g_2(y; \boldsymbol{v}, c_{-i}) = \begin{cases} g_1(y; \boldsymbol{v}, c_{-i}) + b_{-i,(2)} - c_{-i,(1)} & y \le d_{i,2} \\ \frac{e_i}{\gamma d_{i,1}}\left(y - (1-\gamma)d_{i,1}\right) & d_{i,2} < y \le (1-\gamma)d_{i,1} \\ 0 & y > (1-\gamma)d_{i,1} \end{cases}$$

Since $g_2$ is continuous, we can verify the convexity of $g_2$ by calculating the derivatives. It is straightforward that the derivative is $-1 + \frac{e_i}{\gamma d_{i,1}}$ in $(-\infty, d_{i,2})$, $\frac{e_i}{\gamma d_{i,1}}$ in $(d_{i,2}, (1-\gamma)d_{i,1})$, and 0 in $((1-\gamma)d_{i,1}, +\infty)$, which is monotonically increasing. Hence the convexity property holds.

(4) When $v_i < b_{-i,(1)}$, $v_i > v_{-i,(1)}$ and $(1-\gamma)d_{i,1} < d_{i,2}$, similar to case (3), we have

$$g_1(y; \boldsymbol{v}, c_{-i}) = \begin{cases} \frac{d_{i,2} - c_{-i,(1)}}{\gamma d_{i,1}}(y - d_{i,1}) - v_i & y \le d_{i,1} \\ -b_{-i,(1)} + y & y > d_{i,1} \end{cases}$$

$$g_2(y; \boldsymbol{v}, c_{-i}) = \begin{cases} g_1(y; \boldsymbol{v}, c_{-i}) + b_{-i,(2)} - c_{-i,(1)} & y \le (1-\gamma)d_{i,1} \\ 0 & y > (1-\gamma)d_{i,1} \end{cases}$$

Besides, in order to extend the above definitions to $T$ auctions, we define four disjoint auction set as Equation (12) shows.

$$\begin{aligned}
C_1 &= \left\{t | v_i^t \ge b_{-i,(1)}^t\right\} \\
C_2 &= \left\{t | v_i^t < b_{-i,(1)}^t, v_i^t \le v_{-i,(1)}^t\right\} \\
C_3 &= \left\{t | v_i^t < b_{-i,(1)}^t, v_i^t > v_{-i,(1)}^t, (1-\gamma)d_{i,1}^t \ge d_{i,2}^t\right\} \\
C_4 &= \left\{t | v_i^t < b_{-i,(1)}^t, v_i^t > v_{-i,(1)}^t, (1-\gamma)d_{i,1}^t < d_{i,2}^t\right\}
\end{aligned} \qquad (12)$$

Define $G_1$ and $G_2$ as $G_1(\boldsymbol{\omega}_i) = \sum_t g_1(\boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t; \boldsymbol{v}^t, c_{-i}^t)$ and $G_2(\boldsymbol{\omega}_i) = \sum_t g_2(\boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t; \boldsymbol{v}^t, c_{-i}^t)$. Then the objective of Problem (11) can be rewritten as $G_1(\boldsymbol{\omega}_i) - G_2(\boldsymbol{\omega}_i)$. Based on the approach adopted in [23], we propose Algorithm 2 to solve this problem.

**Algorithm 2** DC algorithm

1: Initialize $\boldsymbol{\omega}_i^0$.
2: **for** $k = 1$ to $K$ **do**
3: $\quad \boldsymbol{\omega}_i^k \leftarrow \text{DCA}(\boldsymbol{\omega}_i^{k-1})$.
4: $\quad \boldsymbol{\omega}_i^k \leftarrow \text{TUNE}(\boldsymbol{\omega}_i^k)$ when $\boldsymbol{\omega}_i^k \neq \mathbf{0}$.
5: $\quad$ Break if $\|\boldsymbol{\omega}_i^k - \boldsymbol{\omega}_i^{k-1}\| < \epsilon$.
6: **end for**

Here $K$ is the maximum number of iterations and $\epsilon$ is used as a threshold indicating convergence. What's more, $\text{DCA}(\boldsymbol{\omega}_i^{k-1})$ and $\text{TUNE}(\boldsymbol{\omega}_i^k)$ are used to update $\boldsymbol{\omega}_i$. To be specific, $\text{DCA}(\boldsymbol{\omega}_i^{k-1})$ means to solve Problem (13), where $\delta G_2(\boldsymbol{\omega}_i^{k-1})$ denotes an arbitrary element of the sub-gradient $\partial G_2(\boldsymbol{\omega}_i^{k-1})$. Thus $\text{DCA}(\boldsymbol{\omega}_i^{k-1})$ belongs to quadratic-programming problems and can be tackled using any standard QP solver.

$$
\begin{aligned}
\min_{\|\boldsymbol{\omega}_i\| \le \Lambda, s} \quad & \sum_{t=1}^{T} s_t - \delta G_2\left(\boldsymbol{\omega}_i^{k-1}\right) \cdot \boldsymbol{\omega}_i \\
\text{s.t.} \quad & \begin{cases}
\boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t \ge 0, & t = 1, 2, \ldots, T \\
s_t \ge -b_{-i,(1)}^t + \boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t, & t \in C_1 \cup C_3 \cup C_4 \\
s_t \ge -v_i^t - \boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t + c_{-i,(1)}^t, & t \in C_2 \\
s_t \ge \left(1 + \frac{e_i^t}{\gamma d_{i,1}^t}\right)\left(\boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t - d_{i,1}^t\right) - v_{-i,(1)}^t, & t \in C_2 \\
s_t \ge \left(-1 + \frac{e_i^t}{\gamma d_{i,1}^t}\right)\left(\boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t - d_{i,1}^t\right) - v_i^t, & t \in C_3 \\
s_t \ge \frac{d_{i,2}^t - c_{-i,(1)}^t}{\gamma d_{i,1}^t}\left(\boldsymbol{\omega}_i \cdot \boldsymbol{x}_i^t - d_{i,1}^t\right) - v_i^t, & t \in C_4
\end{cases}
\end{aligned}
\tag{13}
$$

In Algorithm 2, $\text{DCA}(\boldsymbol{\omega}_i^{k-1})$ (line 3) is used to specify the direction of $\boldsymbol{\omega}_i^k$, we propose $\text{TUNE}(\boldsymbol{\omega}_i^k)$ (line 4) to further determine the norm of $\boldsymbol{\omega}_i^k$, that is,

$$
\min_{0 \le \eta \le \Lambda} \sum_{t=1}^{T} L^{\gamma}\left(\eta \frac{\boldsymbol{\omega}_i^k}{\|\boldsymbol{\omega}_i^k\|} \cdot \boldsymbol{x}_i^t; \boldsymbol{v}^t, c_{-i}^t\right). \tag{14}
$$

However, instead of directly solving Problem (14), which can be done in polynomial time ($O(T \log T)$), we merely evaluate a few discrete values of $\eta$ and select the best. In our empirical studies, this approach still has good performance.

## 5.3 Heuristic algorithm for the general case

We have proposed Algorithm 2 to optimize the coupon of one bidder. Now we solve the coupon design problem for the general case in the solution framework defined in Section 4. Note that $h_i$ has the same formula, i.e. $h_i = h, \forall i$, and we use $\boldsymbol{\omega} = \boldsymbol{\omega}_i$ to denote the weight for each bidder. Our algorithm is presented as Algorithm 3.

Here the algorithm terminates when $\boldsymbol{\omega}$ converges. Notice that the modification of Algorithm 2 (line 6 in Algorithm 3) is reflected in mainly two aspects.

**Algorithm 3** Algorithm for the general case

1: Initialize $\boldsymbol{\omega} \leftarrow \mathbf{0}$.
2: **while** not terminated **do**
3: $\quad$ Generate a random permutation of 1 to $n$ as $\hat{N}$.
4: $\quad$ **for** $i = \hat{N}_1$ to $\hat{N}_n$ **do**
5: $\quad\quad$ Use $\boldsymbol{\omega}$ to get the coupons for bidders except $i$, i.e. $c_j^t = \boldsymbol{\omega} \cdot \boldsymbol{x}_j^t$, and make sure all the coupons are non-negative.
6: $\quad\quad$ Run the modification of Algorithm 2 and use the output to update $\boldsymbol{\omega}$.
7: $\quad$ **end for**
8: $\quad$ Fix the direction of $\boldsymbol{\omega}$, multiply it by different scales, and calculate the corresponding revenue. Then select the scale with the maximum revenue, denoted as $\eta^*$, and use $\eta^* \boldsymbol{\omega}$ to update $\boldsymbol{\omega}$.
9: **end while**

- Since there exists a regularizer $\lambda(y - c_i)^2$ in line 4 of Algorithm1, the objective of $\text{DCA}(\boldsymbol{\omega}_i^{k-1})$ can be modified with

$$
\min_{\|\boldsymbol{\omega}_i\| \le \Lambda, s} \quad \sum_{t=1}^{T} s_t - \delta G_2(\boldsymbol{\omega}_i^{k-1}) \cdot \boldsymbol{\omega}_i + \lambda \|\boldsymbol{\omega}_i - \boldsymbol{\omega}_i^{k-1}\|^2,
$$

  which is still a quadratic-programming problems.
- $\text{TUNE}(\boldsymbol{\omega}_i^k)$ in Algorithm 2 is not necessary. Instead, we fine tune $\boldsymbol{\omega}$ after successively optimizing $\boldsymbol{\omega}$ for all the bidders, as implemented in line 8 of Algorithm 3.

# 6 EXPERIMENT

In this section, we implement our algorithms for coupon design in both no-feature case and general case and conduct extensive experiments. We compare the performance of our algorithms with existing ones in terms of revenue gain. Note that the work in this paper is motivated mainly by algorithms in [23] and [14], we implement these algorithms for comparison. First of all, we briefly introduce these algorithms.

- In the second-price auction with anonymous reserve price, the reserve price is the minimum amount that a bidder has to pay if he wins the auction, and anonymity means that all bidders share the same minimum amount. [23] designs the surrogate loss function and proposes corresponding algorithms to select the anonymous reserve price for both no-feature case and general case. We use ARP-NA and ARP-GA to denote the algorithms for the no-feature case and the general case, respectively.
- In the boosted second-price auction without reserve price, bidder $i$ is assigned with a boost $\beta_i$, the slot is allocated to the bidder with the highest boosted bid, i.e. $i^* \in \arg\max_i \{b_i \beta_i\}$, and he pays $\max_{i \neq i^*} \{b_i \beta_i\} / \beta_{i^*}$. Golrezaei et al. [14] proposes an iterative algorithm, to successively optimize one of the boost values while fixing all other boost values. We use BSP-AM to denote this algorithm as [14] does. It is worth noting that BSP-AM works in the no-feature case.

We then use both synthetic data and industrial data to verify the properties of our algorithms and compare the performance with that of the aforementioned algorithms. It is worth noting that $\gamma$ and $\lambda$ are chosen to be 0.1 and 0.01 respectively.

## 6.1 Synthetic data

We choose two types of value distribution, one is the uniform distribution, and another is the Pareto distribution with density function $p(x) = \frac{am^a}{x^{a+1}}$. There are 5 bidders in our experiments. For the uniform distribution, each time we simulate 1000 auctions. To implement this, we sample 1000 numbers from $U(0, 1)$ for each bidder, and then each bidder is assigned with a random scale size to multiply these 1000 numbers as his values in 1000 auctions. Next we use 700 auctions of these auctions as our training data, and use the remaining as testing data. We implement ARP-NA, BSP-AM and Alg.1 to fit the reserve price, boost values and coupons, respectively on the training data, and calculate the corresponding revenue on the testing data. We use Equation (15) to denote the level of increment in revenue:

$$\rho_a = \frac{Rev_a - Rev_0}{Rev_0} \tag{15}$$

where $Rev_a$ represents the revenue achieved through algorithm $a$ (i.e., ARP-NA, BSP-AM or Alg.1) and $Rev_0$ denotes the revenue obtained without these methods in the second-price auction.

After repeating 50 times, we demonstrate the average level of increment in revenue for each algorithm on both training data and testing data for these two kinds of distributions. In Figure 4, for both (a) and (b), the y-axis denotes the value of $\rho_a$.

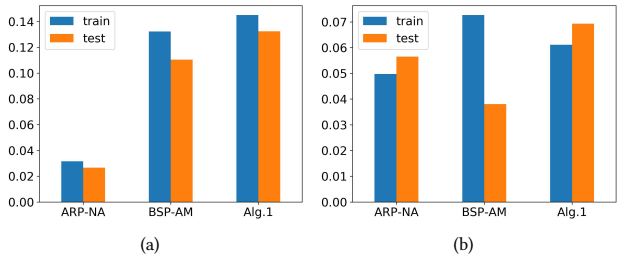


**Figure 4: Revenue comparison for the synthetic data. (a) Uniform distribution; (b) Pareto distribution.**

As shown in Figure 4 (a), Alg.1 improves the revenue on testing data by about 13.2%, while the compared algorithms improves the revenue on testing data by about 2.6% and 11%; In Figure 4 (b), Alg.1 also outperforms the other two algorithms. It improves the revenue on testing data by about 6.9%, while other algorithms improves the revenue on testing data by about 5.6% and 3.8%. These results demonstrate the effectiveness of Alg.1 in the no-feature case.

## 6.2 Industrial data

*Data description.* The industrial data we used comes from Tiktok. This platform reaches 320 million DAU (daily active user) and 500 million MAU (monthly active users) in 2018. We randomly pick 29, 000 advertisers and extract 1, 000 auctions for each advertiser from the log. Each advertiser instance contains 178 features, including labels, industry category, pricing type (i.e., cost per click, cost per mille or cost per action), budget and so on, where the continuous variables are normalized and the category variables are represented with one-hot encoding.

We randomly choose one advertiser and the corresponding 50 auctions. Figure 5 shows the relation between our surrogate loss and the real loss for this advertiser based on these auctions . The

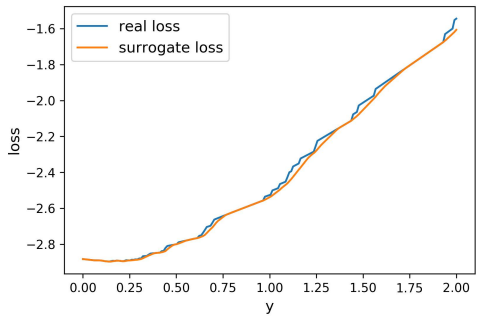


**Figure 5: The relationship between the surrogate loss and the real loss.**

x-axis is the value of coupon provided while the y-axis is the value of loss. Based on Figure 5, the surrogate loss is smoother and close to the real loss. Together with previous theoretical analysis, the experiment results show that the surrogate loss is effective.

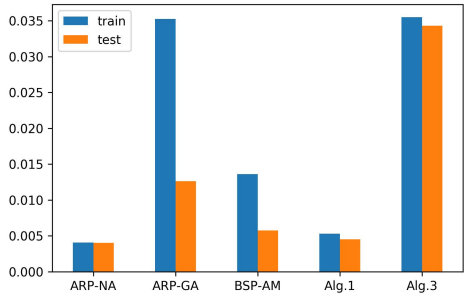


**Figure 6: Revenue comparison for the industrial data.**

Figure 6 shows the results of the industrial data. The y-axis is the value of $\rho_a$. APR-GA and Alg.3 are used for the general case while the other algorithms are used for the no-feature case. APR-GA and Alg.3 are significantly better than the other algorithms. In Figure 6, APR-GA improves the revenue by about 1.2% and Alg.3 improves the revenue by about 3.4%. While for the other algorithms, APR-NA, BSP-AM and Alg.1 improves the revenue by about 0.40%, 0.57% and 0.45%, respectively. And the revenue improvement of Alg.3 is three times that of the ARP-GA.

## 7 CONCLUSION

In this paper, we first analyze the properties of the optimal coupons; Then, we consider the coupon design problem in two cases. In the no-feature case, we provide hardness results and a heuristic algorithm.In the general case, we provide a surrogate loss and prove its effectiveness. A learning algorithm is also proposed based on the DC programming; Finally, extensive experiments are conducted. And our algorithms outperform previous algorithms significantly.

## ACKNOWLEDGMENT

This work is supported by Science and Technology Innovation 2030 — "New Generation Artificial Intelligence" Major Project No. 2018AAA0100904, 2018AAA0101103, Turing AI Institute of Nanjing and Beijing Academy of Artificial Intelligence (BAAI).

# REFERENCES


[1] Mohammad Akbarpour and Shengwu Li. 2018. Credible Mechanisms. In *Proceedings of the 2018 ACM Conference on Economics and Computation, Ithaca, NY, USA, June 18-22, 2018*. 371. https://doi.org/10.1145/3219166.3219169

[2] Kareem Amin, Afshin Rostamizadeh, and Umar Syed. 2013. Learning prices for repeated auctions with strategic buyers. In *Advances in Neural Information Processing Systems*. 1169–1177.

[3] Andreas Argyriou, Raphael Hauser, Charles A Micchelli, and Massimiliano Pontil. 2006. A DC-programming algorithm for kernel selection. In *Proceedings of the 23rd international conference on Machine learning*. ACM, 41–48.

[4] Maria-Florina Balcan, Avrim Blum, Jason D Hartline, and Yishay Mansour. 2008. Reducing mechanism design to algorithm design via machine learning. *J. Comput. System Sci.* 74, 8 (2008), 1245–1270.

[5] Helmut Bester and Emmanuel Petrakis. 1996. Coupons and oligopolistic price discrimination. *International journal of industrial organization* 14, 2 (1996), 227–242.

[6] Avrim Blum, Vijay Kumar, Atri Rudra, and Felix Wu. 2004. Online learning in online auctions. *Theoretical Computer Science* 324, 2-3 (2004), 137–146.

[7] Simon Board and Andrzej Skrzypacz. 2016. Revenue management with forward-looking buyers. *Journal of Political Economy* 124, 4 (2016), 1046–1087.

[8] Qingpeng Cai, Aris Filos-Ratsikas, Pingzhong Tang, and Yiwei Zhang. 2018. Reinforcement mechanism design for fraudulent behaviour in e-commerce. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[9] Nicolò Cesa-Bianchi, Claudio Gentile, and Yishay Mansour. 2015. Regret Minimization for Reserve Prices in Second-Price Auctions. *IEEE Trans. Information Theory* 61, 1 (2015), 549–564. https://doi.org/10.1109/TIT.2014.2365772

[10] Edward H Clarke. 1971. Multipart pricing of public goods. *Public choice* 11, 1 (1971), 17–33.

[11] Mahsa Derakhshan, Negin Golrezaei, and Renato Paes Leme. 2019. LP-based Approximation for Personalized Reserve Prices. *arXiv preprint arXiv:1905.01526* (2019).

[12] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. 2007. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American economic review* 97, 1 (2007), 242–259.

[13] Noah Golowich, Harikrishna Narasimhan, and David C Parkes. 2018. Deep Learning for Multi-Facility Location Mechanism Design.. In *IJCAI*. 261–267.

[14] Negin Golrezaei, Max Lin, Vahab Mirrokni, and Hamid Nazerzadeh. 2017. Boosted second price auctions: Revenue optimization for heterogeneous bidders. *Available at SSRN 3016465* (2017).

[15] Theodore Groves et al. 1973. Incentives in teams. *Econometrica* 41, 4 (1973), 617–631.

[16] Jiong Guo, Jens Gramm, Falk Hüffner, Rolf Niedermeier, and Sebastian Wernicke. 2006. Compression-based fixed-parameter algorithms for feedback vertex set and edge bipartization. *J. Comput. System Sci.* 72, 8 (2006), 1386–1396.

[17] Jason D Hartline and Tim Roughgarden. 2009. Simple versus optimal mechanisms. In *Proceedings of the 10th ACM conference on Electronic commerce*. ACM, 225–234.

[18] Reiner Horst and Nguyen V Thoai. 1999. DC programming: overview. *Journal of Optimization Theory and Applications* 103, 1 (1999), 1–43.

[19] Nitish Korula, Vahab Mirrokni, and Hamid Nazerzadeh. 2015. Optimizing display advertising markets: Challenges and directions. *IEEE Internet Computing* 20, 1 (2015), 28–35.

[20] Sébastien Lahaie and David M Pennock. 2007. Revenue analysis of a family of ranking rules for keyword auctions. In *Proceedings of the 8th ACM conference on Electronic commerce*. ACM, 50–56.

[21] Hoai An Le Thi, Hoai Minh Le, Tao Pham Dinh, et al. 2008. A DC programming approach for feature selection in support vector machines learning. *Advances in Data Analysis and Classification* 2, 3 (2008), 259–278.

[22] Eric Maskin, J Riley, and F Hahn. 1989. Optimal Multi-Unit Auctions. *The Economics of Missing Markets, Information, and Games* (1989).

[23] Andres M Medina and Mehryar Mohri. 2014. Learning theory and algorithms for revenue optimization in second price auctions with reserve. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 262–270.

[24] José Luis Moraga-González and Emmanuel Petrakis. 1999. Coupon advertising under imperfect price information. *Journal of Economics & Management Strategy* 8, 4 (1999), 523–544.

[25] Roger B Myerson. 1981. Optimal auction design. *Mathematics of operations research* 6, 1 (1981), 58–73.

[26] Thomas Nedelec, Noureddine El Karoui, and Vianney Perchet. 2019. Learning to Bid in Revenue Maximizing Auction. In *Companion Proceedings of The 2019 World Wide Web Conference*. ACM, 934–935.

[27] Michael Ostrovsky and Michael Schwarz. 2011. Reserve prices in internet advertising auctions: a field experiment. *EC* 11 (2011), 59–60.

[28] Gregory Pavlov. 2011. Optimal mechanism for selling two goods. *The BE Journal of Theoretical Economics* 11, 1 (2011).

[29] Tim Roughgarden and Okke Schrijvers. 2016. Ironing in the dark. In *Proceedings of the 2016 ACM Conference on Economics and Computation*. ACM, 1–18.

[30] Maja R Rudolph, Joseph G Ellis, and David M Blei. 2016. Objective variables for probabilistic revenue maximization in second-price auctions with reserve. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1113–1122.

[31] Greg Shaffer and Z John Zhang. 2000. Pay to switch or pay to stay: preference-based price discrimination in markets with switching costs. *Journal of Economics & Management Strategy* 9, 3 (2000), 397–424.

[32] Weiran Shen, Sébastien Lahaie, and Renato Paes Leme. 2019. Learning to Clear the Market. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Vol. 97. PMLR, 5710–5718.

[33] Weiran Shen, Binghui Peng, Hanpeng Liu, Michael Zhang, Ruohan Qian, Yan Hong, Zhi Guo, Zongyao Ding, Pengjun Lu, and Pingzhong Tang. 2020. Reinforcement mechanism design, with applications to dynamic pricing in sponsored search auctions. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

[34] Weiran Shen, Pingzhong Tang, and Song Zuo. 2019. Automated mechanism design via neural networks. In *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 215–223.

[35] Pingzhong Tang. 2017. Reinforcement mechanism design.. In *IJCAI*. 5146–5150.

[36] Pingzhong Tang and Zihe Wang. 2017. Optimal mechanisms with simple menus. *Journal of Mathematical Economics* 69 (2017), 54–70.

[37] Pham Dinh Tao and Le Thi Hoai An. 1997. Convex analysis approach to DC programming: theory, algorithms and applications. *Acta mathematica vietnamica* 22, 1 (1997), 289–355.

[38] William Vickrey. 1961. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance* 16, 1 (1961), 8–37.

[39] Rodney B Wallace. 2004. Preference-Based Discrimination and Profit: On the Profitability of Discriminatory Spatial Policy. *Journal of Economics & Management Strategy* 13, 2 (2004), 351–369.