Optimal Real Time Bidding in Online Advertising

David Rådberg

LUND UNIVERSITY

Department of Automatic Control

MSc Thesis
TFRT-6060
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Tryckeriet i E-huset
Lund 2018

Abstract

This thesis explores some of the possibilities of demand side optimization in online advertising, specifically how to evaluate and bid optimally in real time bidding. Theory for many types of optimizations is discussed. The thesis evaluates auctions from a game theory and control theory perspective. It also discusses how big data sets can be used in real time, and how agents can explore unknown stochastic environments.

All items are valued through an estimated action probability, and a control system is designed to minimize the cost for these actions. The control system aims to find the lowest possible price per item while spending the entire budget. Periodic market changes and censored data makes this task hard and imposes low pass characteristics on the closed system. Using data to evaluate items is a high dimensional problem with very small probabilities. When data is limited the algorithm is forced to choose between low variance and precision. The choice between exploring and exploiting the unknown environment is crucial for long and short term results.

An optimization algorithm was implemented and run in a live environment. The algorithm was able to control the spend optimally, but distributed it suboptimally.

Acknowledgements

I want to thank Martina Maggio and Karl-Erik Årzén at the Department of Automatic Control at Lund University for all their help, tips, and comments during the work on this thesis. I would like to express my sincere gratitude to Rasmus Larsson and Carl-Johan Grund at Emerse for giving me the opportunity to do this thesis, and to access their technology and knowledge. I am grateful to the entire team at Emerse for their help, support and good company. A special thanks to Niklas Karlsson.

Contents

1

Introduction

This thesis is about technology in online advertising. Specifically how algorithms can be used to improve results for advertisers. The thesis is done for and with Emerse Sverige AB which has the technology and connections needed to run these kind of algorithms. Some parts of the work in this thesis have been implemented, tested, and integrated in the Emerse platform. Other parts are purely analytical, but can give a better understanding of the possibilities and challenges of online advertising, and make new algorithms possible. Table 1.1 introduces some abbreviatiosn and terms used in the rest of this document.

Table 1.1 Terminology and abbreviations.

Term	Explanation
Creative	A file containing an advertisement that is ready to be put online.
Ad slot	An open spot for a creative on a website or app with a user.
Impression	A view of a creative, which happens after an ad slot is purchased.
Advertiser	A company with advertisements that they want to distribute.
Publisher	A company with ad slots that they can fill with advertisements.
CPM	Cost per mille, the cost of an impression times 1000.
CPC	Cost per click, the amount of money that is spent for every click
RTB	Real time bidding, the real time auction for buying impressions.
DSP	Demand side platform, the bidder in RTB that purchases ad slots for advertisers.
SSP	Supply side platform, the seller in RTB that sells ad slots for publishers.

1.1 Online advertising

Advertising online means putting advertisement in online media. This can be on websites or apps, on videos articles or social media. The growth of the internet creates many new possibilities, and new ways to advertise. When advertising online the advertiser has the ability to choose their audience.

Some of the terminology used in online advertising has been presented in Table 1.1 and will be explained here. When an advertisement has been converted to a file, this is called a creative. The creative is some type, for example a image or video, and a size, for example 300 times 250 pixels. Two creatives can have the same ad, but have different sizes, and will be treated differently. When a user is on a web page or app, the open places for ads are called ad-slots.

After the ad slots are sold they can be filled with creatives. The act of one person viewing one creative is called an impression. Generally an impression is counted when the ad slot has been purchased although there is no certain way to check if the ad was actually ever viewed.

1.2 Programmatic advertising

Programmatic advertising is based on the idea of using programs in the exchange of ad slots. The programs complete the purchases without any human to human interaction or negotiation. Today it is the most common way to trade ad slots in online advertising. As in any transaction of goods there exists a demand side and a supply side. The supply side is based on a publisher with a supply of ad slots that they want to sell to the highest bidders.

The demand side is based on an advertiser which has ads (creatives) and a demand to distribute them. The transaction is performed by a demand side platform (DSP) and a supply side platform (SSP). The platforms have the technology and connections to make the transactions possible but generally only represent the advertisers and publisher, and do not purchase or sell their own inventory. This is an important distinction to make in order to understand how a DSP works with optimization. They are generally not able or allowed to change the content in the creative and can only optimize the results by showing the creative to the correct audience.

1.3 Real time bidding

Real time bidding is the most common way ad slots are sold through programmatic advertising. It is a form of auction that is performed in real time over the internet. When an SSP has an ad slot available, it will send out bid requests to several DSP's. If the DSP is interested in the ad slot it has around 100 ms to respond with a bid. The DSP with the highest bid gets to put their creative on the ad slot.

The short time period makes manual bidding performed by humans impossible, and puts requirements on algorithms, code infrastructure and servers. Every ad slot is sold with an individual real time auction meaning that an advertiser has the opportunity to be very specific about what impressions they want to purchase and for what cost.

Table 1.2 shows an approximate time line of the real time bidding for one ad slot. This is partly based on the RTB framwork described in [IAB, 2017]. The invitation to bid on an ad slot is called a bid request. This is sent from the SSP to the DSP and will contain relevant information about the ad slot. The DSP will respond with a bid response which will contain information about the bid price and the chosen creative. Bidding agents are the components that decide the actual bid price. A bidding agent can represent one or more creatives and will only receive bid request for ad slots that fit the requirements for the creatives.

Table 1.2 Timeline for real time bidding

Approximate time (ms)	**Action**
t = 0	A user loads a website or app
t = 5	Bid requests are sent out to several DSP's
t = 10	A DSP receives a bid request
t = 11	The DSP forwards the bid request to a suitable bidding agent
t = 12	The bidding agent decides a bid and creative for the ad slot
t = 30	The DSP sends a bid response containing the bid price
t = 35	The SSP receives the bid response
t = 100	The SSP chooses the creative of the highest bidder
t = 105	The creative is uploaded on the website/app
t = 110	The creative is viewed by the user

1.4 Demand side optimization

This paper will only focus on the optimization from the demand side, even though there is possibility for optimization algorithms on both sides. The company Emerse has a DSP in which they run algorithms to optimize the targets of their clients, the advertisers. In some campaigns the advertiser is simply interested in getting the advertisement seen as many times as possible, making every ad slot equally valuable. For such purposes a normal bidding strategy is to bid an equal amount on every ad slot. This bidding strategy is called fixed price bidding. The result of these ads could be increased brand awareness, or increased long term sales, but these products (or the cause of them) are generally hard to measure.

This thesis will focus on algorithms that put value in actions. An action is here a general term for a viewer of an ad, doing a trackable action that will be used as optimization target. An action can be a click, an online registration, purchase or booking. Every ad slot is seen by one person and this person can, at most, perform one specific action. In practice, the value of every ad slot is binary when optimizing towards an action. Either the user will perform the action, or not. It is not currently possible to predict precisely which impression will perform an action, so an action probability is used instead.

Campaigns are generally dealing with large numbers, so it is fair to assume that the advertisers are risk-neutral, meaning that they are not affected by the uncertainty [Başar and Olsder, 1998]. Being risk neutral, the value of an item is equal to the expectancy of the value, and so the probability can be used for evaluation. This thesis will focus on using click as an action, but the theory can be generalized to any action. The reason why clicks are more suitable for optimization, is that the probability of a user clicking on an ad is in the order of 10^{-3}, while the probability of, for example ordering the shoes on the ad, might be in the order of 10^{-6}. Very small probabilities are hard to estimate, and to deal with.

1.5 RTB infrastructure

This section gives a simple description of the RTB infrastructure. Figure 1.1 shows a simple model of real time bidding. This is from the point of view of a DSP so the SSP is basically a black box. The DSP is also very simplified since this paper is mostly focused on the actual bidding. The process begins with an SSP sending a bid request which says that there is an ad slot available for bidding, and also contains information of the ad slot. An exchange connector forwards the bid request to a bidding agent. A bidding agent is the part of a DSP which performs the actual bidding.

A bidding agent is a quite complicated program which should handle creative, campaign and budget states. In this representation the bidding logic has been put outside the bidding agent to separate the optimization part, which is new, from the base of the DSP which has already been created. There is however no reason why the bidding logic could not be put directly in the bidding agents. The bidding agent sends the bidding logic a bid request, which holds information about the available ad slot, and a creative object, which hold information about the chosen creative for this ad slot. With the help of data collected in the servers, the bidding logic will decide on an optimal bid price for the impression. The bid price is sent back to the SSP in a bid response which contains the bid price and details about the chosen creative. In order for the bidding logic to work it is essential that it has large amounts of relevant data which should be collected continuously from the SSP and Internet activity. The data contains sets of features of impressions that have, and have not, received clicks.

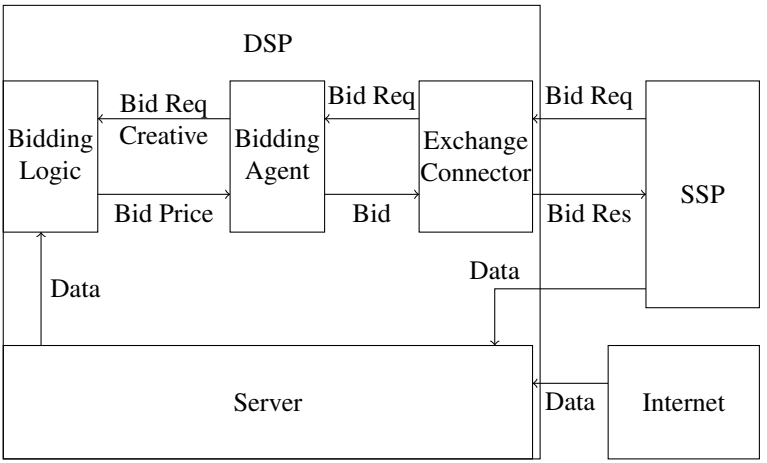


Figure 1.1 Overview of real time bidding

1.6 Thesis outline

The goal of the thesis is to design and implement optimization algorithms for real time bidding. The algorithms should help advertisers increase clicks and/or decrease budget spend. Chapter 2 will introduce the auction and theory behind it. Chapter 3 will discuss how to estimate the value of an ad slot. Chapters 4 and 5 will introduce two possible optimization functions. Chapters 6 and 7 will discuss how these can be implemented. Chapters 8 and 9 will describe two possible algorithms. Chapter 10 will present the results, and Chapter 11 will bring some conclusions. Generally Chapters 1-3 presents the background of the thesis, while the rest describe new work.

2

Auction Theory

2.1 Background

As mentioned, the SSP handles the bids with an auction. An auction is an operation that, given a set of bids, chooses a winner, and a price for an item. Every RTB auction is a closed auction meaning that every bidder submits one sealed bid and the highest bidder wins the item. The bidders do not get a chance to see any other bids, and no chance of changing or putting multiple bids.

In RTB there is also no information given about the number of bidders, and the winning price is only revealed to the highest bidder. A loosing bidder gets no feedback, except that there is at least one bid higher than theirs. This lack of information makes algorithm design harder. Data like historical bids and number of active bidders could be used to create an accurate model of the auction. This model would be a powerful tool in optimization. The remaining part of the auction is deciding the price that the winner pays. The two most common auctions types are first, and second price auction. Before these are presented there will be a simple introduction to game theory and its impact on auction theory. A lot of the game theory is presented in [Başar and Olsder, 1998], and the details behind the auctions are discussed in [Fernandez-Tapia, 2015]

2.2 Game theory in auctions

Game theory is a part of mathematics that studies the cooperation and conflict between decision makers. A game is here a broad term for a situation where several partners take actions to maximize their utility. The actions of other players may affect their utility and what actions are best for them. John Nash did extensive work on Game theory including auctions. Some useful concepts from game theory will be presented in this section.

A strategy is a set of rules that dictates what actions a player will make. It can be, but is not necessarily, dependent on the actions of other players. A strategy can be as simple as always making the same action, but can also be more complex. A

dominant strategy is a strategy that is optimal regardless of the strategies of other players. If a dominant strategy is found, the player does not have to worry about the actions of other players.

A Nash equilibrium is a strategy that is optimal, given that every other player also uses this strategy. If all players would use this strategy in a game, there would be no reason for any player to divert from this strategy.

2.3 First price auction

A first price auction is what most people consider a standard auction. The bidder with the highest bid pays his bid. In an auction with highest rival bid b^* the bidder will pay a price equal to

$$c(b) = \begin{cases} b, & \text{if } b \geq b^*. \\ 0, & \text{if } b < b^*. \end{cases} \tag{2.1}$$

for bid b. Theoretically, an optimal bid in a first price auction would be $b = b^* + \varepsilon$ (given that the bidder valuated the item higher than b^*), but this would require precise knowledge of the rival bidders. There is no general optimal bidding strategy for first price auctions. John Nash showed that there exists equilibria in first price auctions. For a first price auction with n bidders with independent valuations v_i drawn evenly from 0 to 1 there exist an equilibrium in $b_i = \frac{n-1}{n} v_i$. Note that this equilibrium is not a general dominant strategy, only a suggestion for a static equilibrium. In practice we do not know anything about the number of rival bidders, or the distribution of their valuations so this result can not be used directly.

2.4 Second price auction

In a second price auction the highest bidder pays the bid of the second highest bidder. In an auction with highest rival bid b^* the bidder will pay a price equal to

$$c(b) = \begin{cases} b^*, & \text{if } b \geq b^*. \\ 0, & \text{if } b < b^* \end{cases} \tag{2.2}$$

for bid b. This auction is also called a Vickrey auction. One strength of the Vickrey auction is that it encourages bidders to bid truthfully since there is no extra cost for bidding very high. A truthful bidder always bids their true value which will be denoted v.

THEOREM 2.4.1
A bidder with with a valuation v maximizes the expected profit for one second price auction by bidding $b = v$. □

Theorem 2.4.1 can be illustrated by the following three examples:

- If $b > b^*$, the auction is won and the truthful bidder pays b^* which is less than their value.
- A lower bid would result in the same cost if won, but might lose the auction.
- A higher bid would only increase the wins when $b^* > v$, which results in a larger cost than the value.

Since this proof is not dependent on the actions or number of rival bidders this strategy is a dominant strategy. It is an important to note that this strategy is only optimal in a true second price auction, and for only one auction.

2.5 Auction floors

Floors are set by the seller of an auction and changes the characteristics of the auction. There are two types of floors, hard floors and soft floors. A hard floor is a bid limit for an item. If the highest bid is lower than the hard floor the item will not be sold to anyone. This floor introduces a abrupt discontinuity in the auction at the floor, which affects the dynamics of the bidders. The seller sets the hard floor to a price point, at which it is not beneficial for them to sell the item and might be used to push up prices.

The soft floor is a limit between first and second price auction and is a bit more complicated. If the winning bid is larger than the bid floor, the auction is a second price auction, and if the winning bid is lower than the bid floor, the auction is a first price auction. Let's name the highest and second highest bids $b^* \geq b^{**}$, the soft floor f and the winning price c. The cost for the winner will be

$$c = \begin{cases} b^{**}, & \text{if } b^{**} \geq f \\ f, & \text{if } b^* \geq f \geq b^{**} \\ b^*, & \text{if } f \geq b^*. \end{cases} \tag{2.3}$$

It is important to note that if the bid floor is between the highest and second highest bid, the floor will act as the second highest bid and the winner will pay the price of the soft floor. An auction with soft floors cannot be considered a true second, or first price auction, but will be something in between. There is no clear rules for setting floor prices, so the seller can shape the auction into whatever they want using floors.

A seller can change the bidding floors continuously while selling their inventory. This is called a dynamical bid floor. A dynamical bid floor can be used to make sure the floors are set to match bids. The sellers could also use the bid floors to optimize some utility function, like maximize total value or profit. The seller can use optimization algorithms for dynamical floors. These algorithms will have access to all bid data, and could be very precise.

2.6 RTB auction

From the start the auctions in RTB were supposed to be second price in order to encourage truthful bidding. Lately some have claimed that most second price auctions in RTB are not true second price auctions. This could be caused by bid floors, dynamic bid floors, or dishonest auctions. It is hard to verify if a second price auction is legitimate or not, therefore a lot of SSPs have made the choice to move towards first price auctions since they are easier to verify [Sluis, 2017].

Some people claim that all auctions will be first price in the future of RTB so it might be unwise to base optimization algorithms on the assumption of second price auctions. Right now there might be a mix of first price auctions, second price auctions, and something in between. In a "something in between" auction the winning price is dependent of the winning bid, but it is not equal to it.

2.7 Auction theory for RTB

Most work on auction theory is based on game theory. However there are some significant differences between this system and the ones discussed in auction theory. The first one is that auction theory generally focuses on the full auction system, discussing equilibria, fairness and global utility. In representing the buyer side, this report will focus on selfish maximization of one players utility without necessarily worrying about other players actions.

The second difference is that real time bidding auctions are very many, with very small values. An agent might receive several bid requests per second, and so the resource of available auctions can be well represented as a continuous flow. Most auction theory is based on a single auction for an object with defined value. This thesis will claim that continuous bidding with a fixed budget is inherently different from any auction theory and so new theory is needed for the application of RTB.

2.8 Auction as a plant

The probabilistic description for an auction only has meaning if looking at a series of auctions. The bidding algorithm will divide the available impressions into sections. The size of the sections quantizises the precision of the algorithm since all bid request from a section will be treated equally. The bid requests for each section can be modelled as a continuous and possibly time varying flow of available impressions F. At the current bid level b the auction can be modelled as a plant which outputs the flow of won impressions $w(b)F$ with $w(b)$ being the probability of winning an item at bid b. Even though the section might contain impressions with varying win rate functions, the auction can be fully described by the plant $w(b)F$. By interpreting the auction process as plant, the problem formulation takes one big leap from game theory, towards control theory.

2.9 Win rate in auction

It is interesting to know the probability of winning an auction at a certain bid. This will be denoted $w(b)$ and will be viewed as a cumulative distribution function (cdf). A cdf can be written as

$$w(b) = P(b > b^*) = \int_0^b f(x)dx \tag{2.4}$$

with $f(b)$ the probability density function from which b^* is drawn. The probability function for the auction tells us the distribution of highest rival bids. The following limits are put on the model

$$\begin{aligned} 0 \leq w(b) \leq 1 \\ 0 \leq f(b). \end{aligned} \tag{2.5}$$

For some applications, it will be important that these inequalities are strict. This can be imposed by the application.

2.10 Auction section plant

Suppose that the flow of bid requests are divided into sections, where the impressions of the bid requests all have equal, or indistinguishable value to the bidder. It would be logical for the bidder to bid at a constant bid level b. The section has some flow of bid request F and will receive a flow

$$\begin{aligned} F_{impressions}(b) = Fw(b) \\ F_{cost}(b) = Fw(b)c(b) \end{aligned} \tag{2.6}$$

of impressions and cost, where $w(b)$ is the win rate for the current bid level, and $c(b)$ is the average cost per impression, which is normally denoted by CPM. CPM has a factor change of 1000.

If the bidding level is in an active region raising bid will always increase win ratio, and $\frac{dw}{db}$ will be positive, finite and non zero. Any discontinuities in $w(b)$ can be handled by imposing randomized bidding, which will be discussed in detail later in this report. Making every bid price the result of a stochastic process ensures that $w(b)$ is continuous.

$\frac{dc}{db}$ will also be positive and finite, but depending on the type of auction it might be infinitesimal. In a true second price auction raising the bid level by db only affects the average cost by introducing new impressions. The derivative $\frac{dc}{db}$ will be proportional to db, i.e, infinitesimal. From this we can conclude that for an auction section plant, a very small increase of bid level db will result in a very small increase in the flow of impression and cost. The CPM will also increase but the increase might be very close to zero if the auction is a true second price auction.

2.11 Logarithmic and linear scale

This section addresses the general problem of how to model bid space. This may seem very abstract so let us consider an example: Originating at the point $b = 8$ dollars, we move one unit step in the negative direction and end up at the point $b = 4$ dollars. What point would we reach if we, from $b = 8$ dollars, move two unit steps in the negative direction? If bid space is linear we would end up in $b = 0$ dollars which would technically not be a valid bid. Logarithmic scale would give us $b = 2$ dollars. Additional steps would give, $b = -2$ dollars for linear scale, and $b = 1$ dollars for logarithmic. In logarithmic space, the zero is always infinitely far away. This makes sure that bids are always strictly positive. The choice of scale could be reflected in using the log normal distribution for bid perturbation and/or weight function. It can also be reflected as exponential actuation instead of linear.

The first order Taylor expansion is a linear approximation of a function $f(x)$ at a point a and is given by

$$T(x) = f(a) + (x-a)f'(a). \tag{2.7}$$

From this we find that

$$e^x \approx 1 + x \tag{2.8}$$

for small x. This fact can be used when comparing linear and logarithmic scales.

2.12 Efficient market hypothesis

The efficient market hypothesis claims that the market always reflects true value. The theory is generally applied in finance as a claim that no investor can expect to make consistent profit. Applying this to real time bidding could indicate that the prices of impressions reflect their true value and that expensive impressions should always be extra valuable.

In the case of second price bidding you would always pay the true value what ever you would bid (given that the bid is high enough to win the auction) and no evaluation would be needed. However RTB is different from stock markets in that impressions for different clients have different value. The idea of true value does not exist in RTB but you could identify that some impressions have more general quality than others when they are, for example on a website with good reputation.

2.13 Winner's curse

The winner's curse is a game theoretic phenomenon in auctions that says that the winner of an auction tends to have overestimated the value of the item. We can model this as the value estimation for every bidder i consists of a true value and

some error noise e_i. Without making any assumptions on bidding strategies we could set up a hypothesis that the winning bidder k tends to have a positive error e_k i.e the winners tend to have overestimated the value. The error noise can also be added by the bidder as randomized bidding. When using randomized bidding, the bidder tends to win auctions more often when the randomized bid is higher than the bid level.

2.14 Tragedy of the commons

Tragedy of the commons is a game theoretic problem discussing the problems of cooperation [Hardin, 2010]. It describes a common land and some farmers. Every farmer has the opportunity to let his cows use more of the common land. This will increase their profit but be bad for the land. If every farmer acts rational, by a game theoretic description, every farmer will use more of the common land and the land will be destroyed. This theory has been applied to markets, and Zhang suggest that the auction market that is RTB is a case of the tragedy of the commons, for the bidder, if no cooperation is made between them [Zhang et al., 2014]. Every rational bidder will increase their bids until the profit of every bidder is zero.

3

Action probability

3.1 Value

When speaking of auctions, or auction theory the most central variable is the value of the item to be purchased. When deciding on a bid, the bidder would like to have a well defined personal valuation of the item it is bidding on. It has already been stated that the bidder's actions should be risk neutral which means evaluating an item I based on the expected value.

$$Value(I) = E(I) = pv. \tag{3.1}$$

Here p is the probability of the action for this item and v is the value of the item, given that the action will happen. v can be interpreted as the value that an advertiser puts on one action, and can be a parameter, but it could also be a state in a control system. If used in a control system, an algorithm could decide what the value for one action is. This will be discussed in detail in Chapter 4. From now on clicks will be used instead of actions. Then we can say that p is the probability of clicking denoted by the click through rate (CTR). v could be a goal cost per click (CPC). The next problem is that the CTR for one impression is unknown. It has to be estimated through historical data which will further be discussed in this chapter. The details of the probability estimation will be limited since this has been subject to previous research by Emerse.

3.2 Features

A feature is something that describes an object. Features are used to define objects for computers. An object described by n features can be realized as a point in n-dimensional feature space. It could then be desirable to calculate the distance between two points, or in some way transform the space into scalar values. Distances could be used to evaluate if two objects are similar. The scalar could be used to evaluate some kind of quality for objects in different parts of the feature space.

In order to estimate the CTR of an impression, the features of the impression need to be observed. A feature of an impression might be something that describes the creative, like if it is a video or image. A feature might also be something that describes the ad slot, like the site it is put on. In total an impression might have up to 100 features. Evaluating an item with 100 features is far from trivial and could make up a separate machine learning thesis. The full potential of the features are outside the scope of this thesis. Using all features would create a 100 dimensional impression space, which would be far from feasible. In this estimation, only a few will be used. The features used is considered company confidential and will not be covered.

3.3 Feature space

Suppose that we define every impression by n features. Every impression can be put into an n-dimensional feature space. The problem of CTR estimation can be defined as finding a transformation from the feature space to the scalar CTR. The problem with the features of impressions is that they generally are categorical variables that can not be generalized. Dealing with categorical variables is hard because they are discrete and unordered.

Knowing that "Adam" is tall and "Bob" is short, tells us nothing about how tall "Clint" is because we do not know if "Clint" is more equal to "Bob" or "Adam". The features can in general not be generalized so every combination of features have to be treated separately, instead of using regression analysis. The impression space is divided into sets which are all evaluated separately. Every impression is placed in a set and evaluated against impressions with the same set of features. Depending on which features are chosen the sets can vary in size. In order to make a sufficient CTR estimation the set must be sufficiently large so that the data is enough to create a low variance estimation.

3.4 CTR estimator

The distribution of clicks in one section can be described with a Binomial distribution since every impression is independent and with equal probability p. When observing n_c clicks, from n_i impressions, the maximum likelihood estimation of the CTR is

$$\hat{p} = \frac{n_c}{n_i} \tag{3.2}$$

In order to derive the variance of this estimator, the number of clicks is written as $n_c = \sum_{i=1}^{n_i} Y_i$, where every Y_i is an independent Bernoulli distribution with probability p. A Bernoulli distribution is a binary distribution which returns 1 with a probability of p, and 0 with a probability of $(1-p)$. The variance of the Bernoulli distribution can be calculated as

$$Var(Y) = (p \cdot 1^2 + (1-p) \cdot 0^2) - (p \cdot 1 + (1-p) \cdot 0)^2 = p(1-p) \tag{3.3}$$

using $Var(Y) = E[Y^2] - E[Y]^2$. This can be used with equation 3.2 to produce

$$\begin{aligned} \sigma^2 = Var(\hat{p}) &= Var(\frac{1}{n_i} \sum_{i=1}^{n_i} Y_i) \\ &= \frac{1}{n_i^2} \sum_{i=1}^{n_i} Var(Y_i) = \frac{p(1-p)}{n_i}. \end{aligned} \tag{3.4}$$

The only problem with the expression in Equation 3.4 is that it involves the unknown p. It would be tempting to insert $p = \frac{n_c}{n_i}$ into 3.4 but this will not be done. This would assume that $\hat{p} = p$ and that the estimator has no variance which defeats the purpose of estimating the variance. Instead presume that the CTR p is close to constant through all sections. The variance of the estimator will be inverse proportional to the number of impressions i.e $Var(\hat{p}) \sim \frac{1}{n_i}$. This will not give any value to the variance, but it can be used to compare the quality of estimations against each other. Another measurement of the estimation is the relative standard deviation, or coefficient of variation. It is defined as the square root of the variance divided by the mean which gives

$$c_v = \frac{\sqrt{Var(\hat{p})}}{p} = \sqrt{\frac{1-p}{pn_i}} \approx \frac{1}{\sqrt{pn_i}}. \tag{3.5}$$

Since p is always much smaller than 1, it is fine to approximate $1-p$ with 1. c_v shows how much the estimation will vary relatively and can be used to see how many impressions will be needed for good estimations. If $c_v = 0.1$ is desired when $p = 10^{-3}$, $n_i = 100000$ impressions are needed. Equation 3.5 shows that the smaller p is, the more impressions are needed to estimate it relatively good.

3.5 Weighted estimation

Two, or more estimations can be combined into one weighted estimation. This can be done to reduce variance or to add more specific data. Two values x_1 and x_2 are weighted according to

$$x = \frac{w_1 x_1 + w_2 x_2}{w_1 + w_2} \tag{3.6}$$

with the weights w_1 and w_2. The weights could be chosen arbitrarily, but the optimal choice, in least squares, is to choose $w_i = \sigma_i^{-2}$ i.e the inverse of the variance. For the application of weighting CTR estimations the inverse of the variance is equal

to the number of impressions, and the CTR estimation equal to the ratio between clicks and impressions. The weighted CTR estimation will be given by

$$p = \frac{n_i^1 \frac{n_c^1}{n_i^1} + n_i^2 \frac{n_c^2}{n_i^2}}{n_i^1 + n_i^2} = \frac{n_c^1 + n_c^2}{n_i^1 + n_i^2} \tag{3.7}$$

3.6 Exploration versus exploitation

The exploration versus exploitation problem is a classical probability theory problem which has many applications. It is sometimes referred to as the multi-armed bandit [Steven, 2010] problem and consists of spending resources in an unknown environment. The question is do you spend your money on bandits that have good historical returns, or do you try new bandits to find better returns. In real time bidding, you receive no information from losing an auction.

A low valued section will get low bids, which might get no wins, so a low valued section will not get any new data and no chance to recover from a bad start. In a dynamic environment it is crucial that data is updated continuously. A similar problem has been worked on in control theory called the dual control problem [Wittenmark, 1995]. Basically an adaptive controller working in an unknown environment has dual goals. The controller should control the system as good as possible while still keeping the system excited so that the system can be investigated for better control in the future. In both game and control theory complicated optimal solutions have been found but most of them are not practically implementable. It is interesting to note that in a exploration environment a controller that minimizes the error might not necessarily be optimal. Characteristics that are generally avoided, like oscillations, overshooting, slow convergence or random behaviour might be okay in the application of real time bidding. Exploration can be prioritized by creating a bias towards impressions with low data.

3.7 Heisenberg bidding

Heisenberg bidding is a strategy for real time bidding invented by Niklas Karlsson [Karlsson, 2016]. It involves perturbing a nominal bid randomly in order to solve two problems. The first problem is that the win probability is generally not smooth, which gives discontinuous plant gain. A controller working on a plant with a plant with discontinuous gain is likely to be unstable and there might not exist a stable solution.

The second problem that Heisenberg bidding solves is the problem of exploration vs exploitation. Heisenberg bidding will make sure that even low valued impressions have a chance of winning which keeps up the exploration. Karlsson has proposed the use of the Gamma function for perturbation, but almost any probabil-

ity function can be used. Heisenberg bidding resembles the uncertainties that are associated with particles in quantum physics. And just like a low energy particle has a small chance of escaping a steep quantum well, a low value impression has the chance to be purchased.

From a signal processing perspective Heisenberg bidding can be seen as adding white noise to a signal to make sure that the system is sufficiently excited (like in leaky LMS). This sort of artificial excitement will always limit the convergence properties of the system. Like in most white noise applications the normal distribution will be used as bid perturbation. The reason for not using a more logarithmic scaled distribution function like the log normal, or gamma distribution is that these are almost always skewed and non symmetric. It is very desirable that both the mean and median of the final bid follows the bid level. Then the expectation of the final bid is the bid level, and we are as likely to bid over as under the bid level. One problem with using the normal distribution is that there always is a chance to get a negative value.

3.8 CTR estimation through set weighting

This section will illustrate how weighting can be used to improve estimation quality. Figure 3.1 shows two copies of the same space. For CTR estimation this is the feature space, but it can be generalized. The sets have been divided into sections with different precision. The sections in Set 2 are more precise, and each section in this set is a subsection of one in Set 1.

The circle illustrates the impression in the feature space, and the grey areas are the active sections that the impression is put into. The section in Set 2 is more precise, and data from this set would create the best estimation. However, since this section is small there might not exist enough data to create a good estimation.

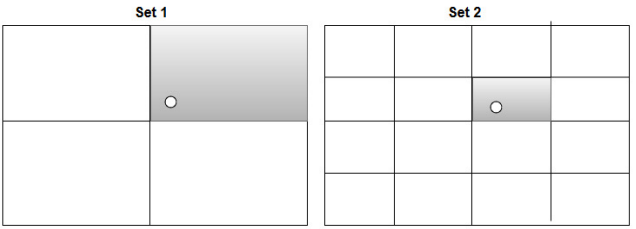


Figure 3.1 Two examples of sets and how they categorize an impression

CTR from both sets can be weighted with Equation 3.6. x_1 and x_2 will be the estimations from sets 1 and 2, which leaves the choice of weights. It is desired that the estimation from Set 2 is weighted highest when there is sufficient data, fading

out the estimation from Set 1. This can be achieved by choosing w_2 to be equal to the number of impressions in this section, which is equivalent to the inverse of the variance. w_1 can be set to a fixed number, which will be insignificant when the number of impressions in Set 2 is large. w_1 can be interpreted as the amounts of impressions needed for a good estimation.

In order to increase exploration rate, it might be beneficial to overestimate impressions with little data a bit. This can be done by adding one or more clicks in the CTR estimation. When there exists much data for this section the added clicks will not have a big effect. If w_1 is chosen large enough the added clicks will only affect the final estimation moderately, since the estimation will be weighted low.

4

Maximum value optimization

4.1 Optimization function

The first three chapters have been focused on background. From this chapter and on, the work is original and new.

Mathematical optimization consists of maximizing, or minimizing some objective function while operating under some constraints. After defining some optimization function, with constraints, an optimal bidding strategy can be deduced and implemented. The first optimization function that will be presented is maximize value, while not spending over budget. The standard value that the algorithm will work towards is clicks, but it could be generalized to any action. The optimization function can be written as

$$\begin{aligned} max &: n_c \\ spend &\leq budget \end{aligned} \tag{4.1}$$

where n_c is the number of clicks. By intuition it is already possible to tell that an optimal solution to this optimization will always spend the entire budget, since there is no penalty for spending if the budget is not reached.

4.2 Section distribution for maximum value

Increasing spend can only increase value, so an optimal solution will always spend the entire budget. Since the spend is constant the optimization problem can be rephrased into minimizing spend/value. For the application of click optimization this means that we want to minimize CPC while spending the entire budget. How should the bids be chosen among the sections? It is reasonable to keep the ratio between the value and bid constant through all sections. The following conjecture

claims that this bid strategy is indeed optimal. The conjecture and the attempt to prove it was inspired by [Fernandez-Tapia, 2017].

CONJECTURE 1
Given a fixed budget and a set of sections, each with CTR p_i, bidding $b_i = vp_i$ for every section i maximizes expected value, where the valuation v is set so that the expected spend is equal to the budget. □

This conjecture will be analyzed with the help of the following theorem.

THEOREM 4.2.1
Given that for every i x_i, $f_i(x) > 0$, $\frac{df_i(x)}{dx} > 0$ and that $f_i(x)$ is continuous.
The solution to the optimization problem $\arg\min \sum_{i=1}^{N} x_i f_i(x_i)$ under the constraint $\sum_{i=1}^{N} x_i = C$ satisfies the condition $f_1(x_1) + x_1 \frac{df_1(x)}{dx} = f_2(x_2) + x_2 \frac{df_2(x)}{dx} = ... = f_N(x_N) + x_N \frac{df_N(x)}{dx}$. □

By interpreting x_i as the spend and $f_i(b_i)$ as the average CPC on section i, and C as the total budget, Theorem 4.2.1 can help to analyze Conjecture 1. This uses the fact that maximizing value is the same thing as minimizing CPC when spending a budget and that the average CPC on a section will increase when increasing demand. To prove the theorem the Lagrange multiplier is used. The Lagrange multiplier for this problem is $\mathscr{L}(x,\lambda) = \sum_{i=1}^{N} x_i f_i(x_i) - \lambda(\sum_{i=1}^{N} x_i - C)$. Any optimal solution to the optimization problem must satisfy $\nabla\mathscr{L} = 0$. This gives

$$\nabla\lambda(\sum_{i=1}^{N} x_i - C) = \nabla \sum_{i=1}^{N} x_i f_i(x_i)$$
$$f_i(x) + x_i \frac{df_i(x)}{dx} = \lambda. \tag{4.2}$$

In order to analyze this solution to the application of CPC minimization $\frac{df_i(x)}{dx}$, i.e, the budget allocation effect on average CPC has to be analyzed. This will be done separately under the assumption of second and first price auctions.

4.2.1 Section distribution in second price auctions

Raising bids in second price auctions only affects the bids that would previously not have been purchased. This means that increasing demand by dx only affects CPC on the section dx. When increasing the allocation by dx the new average CPC will be $f + df = \frac{xf + dxf^*}{x+dx}$ where f^* is the CPC of the new allocation demand and f the old average CPC level. From this we get $df = \frac{f^*dx - fdx}{x+dx}$ and

$$\frac{df}{dx} = \frac{f^* - f}{x + dx} \tag{4.3}$$

Inserting this into 4.2 gives

$$f(1-\frac{x}{x+dx})+\frac{x}{x+dx}f^* = \lambda \tag{4.4}$$

For small dx (which is naturally the case) this gives $f^* = \lambda$, i.e the limit of the effective CPC will be constant at all sections for minimal CPC while spending budget. When bidding $b_i = vp_i$ v is identical to the limit of the effective CPC, which proves Conjecture 1 under the assumption of second price auction.

4.2.2 Section distribution in first price auctions

When raising bids over a section in first price auctions, the expected price of all wins will increase. By increasing demand by dx the CPC will increase over all the section x. For a first price auction the average CPC $f(x)$ for every section will be identical to the valuation v.

Conjecture 1 is true for first price auction if Equation 4.2 implies $f_i(x_i) = k$ for k which is constant for all sections. The analysis becomes more complex under the assumption of first price auction. It is possible that the term $x_i \frac{df_i(x)}{dx}$ could be large in relation to $f_i(x_i)$ and so the term cannot be neglected in general. Instead one could search for the cases when $x_i \frac{df_i(x)}{dx} = kf_i(x_i)$ (again it is crucial that k is constant for all sections). The solution to this is

$$f_i(x) = c_i x^N, \tag{4.5}$$

i.e, if the CPC functions for all sections are functions from demand raised to a common power Conjecture 1 is true for first price auctions. The most interesting special case comes from $N = 1$ when the CPC function is linear. It is tempting to claim that all CPC functions could be modeled as linear and so the Conjecture is true for all auctions. But we generally do not know how the CPC is affected by demand, the functions could even differ between sections and so finding a solution for Equation 4.2 would be impractical or even impossible. We will settle with claiming that bidding $b_i = vp_i$ will produce a good approximation for the optimal distribution.

4.3 Time distribution for maximum value

We now consider how the budget should be distributed over time. The market decides how bid levels translate into spend (and clicks) and is in general time varying. How should budget be distributed in a market that is time varying? If there is no specific need to keep spend even, it would be reasonable to keep the bid level independent of market time variations. The next Conjecture will claim that this strategy is optimal.

CONJECTURE 2
When CPC bidding $b_i(t) = v(t)p_i$ in a time varying market with a fixed budget B for time $0 < t \leq T$, an optimal solution is found as a constant CPC level $v(t) = v$ for which the expected spend is equal to the budget. □

By considering time as small discrete sections, Conjecture 2, is identical to Conjecture 1 which has been proven true. Keeping v static for every section can be translated into keeping v static during all times. There is, however, a difference in implementation, since time only moves one way. It is not feasible to keep v static at all times and still spend the budget perfectly.

The click value, which might be time varying, should be adjusted so that the entire budget gets spent during the campaign time (and not before the end). It is important to understand that the resource of auctions will be time varying, both in the flow of available auctions, and in the price required to win. For example during night we expect less people to be on the internet and so there should be a lower flow of available auctions.

We could also suspect that the low flow of auctions might make the auctions more competitive and drive up prices. How should the algorithm handle such variations? If it is not specifically important to keep spend flow even at all times, it is most profitable to ignore variations and keep v steady. By keeping v steady, the spend will go up and down during a day, but the daily spend will be kept constant (and optimal). The market might also change from day to day, affected by weekdays, holidays, or new campaigns.

4.4 Budget as a soft constraint

For practical implementation the budget constraint will be changed into a soft constraint. This can be done since there are internal blocks preventing any agent from spending money that it does not have. If an agent would run out of budget before the campaign was finished, it would be blocked from spending any more, causing the campaign to end early. This result, while sub optimal, would not cause any real harm. Proper budget spend can be set as a soft constraint while internals function take care of the hard constraint of not spending money that you do not have.

4.5 Reference tracking solution

It is has now been concluded that the optimal solution for value maximization will satisfy:

- All bids are proportional to CTR i.e $b_i(t) = v(t)p_i$.
- The bid level $v(t)$ is held at a constant level that spends the entire budget.

The second point is not possible to satisfy perfectly in a practical environment since we cannot know what bid level corresponds to which spend beforehand. Instead it is proposed that $v(t)$ is adjusted during the run to make sure that the budget is spent correctly.

The optimization problem can be rephrased into a reference tracking problem with optimization function

$$V(v) = K(Spend_{ref} - Spend(v))^2 \tag{4.6}$$

The gradient of this optimization function with respect to bid level v will be

$$g(v) = 2K(Spend_{ref} - Spend(v)). \tag{4.7}$$

Moving in the direction of this gradient is equivalent to moving in the direction of the error. The implementation of gradient descent for the optimization function in Equation 4.6 would be an I controller. An I controller is a controller that sums the error for all times.

4.6 CPC and budget error minimization

It is already concluded that minimizing CPC is equivalent to maximizing the value under budget. The following optimization function combines low CPC and following reference spend. The L2 norm is chosen and both CPC and reference error are divided with reference values in order to become unitless which produces the following optimization which shall be minimized

$$V(b) = \frac{w_1}{CPC_{nom}} CPC(b)^2 + \frac{w_2}{Spend_{ref}} (Spend_{ref} - Spend(b))^2. \tag{4.8}$$

Equation 4.8 has desirable symmetric properties since both terms are unitless. There is a difference in that even though we use a reference for both terms, the CPC does not have to follow the signal CPC_{ref} even though it is evaluated in reference to this signal. The optimization is differentiated in respect to bid to give $\frac{dV}{db} = \frac{2w_1 CPC(b)}{CPC_{nom}} \frac{dCPC}{db} - \frac{2w_2(Spend_{ref} - Spend(b))}{Spend_{ref}} \frac{dSpend}{db}$.

Since both derivatives in the expression are hard to analyze we will once again make the assumption that the CPC is linear with respect to spend i.e $\frac{dCPC}{dSpend} = \lambda$. Using this a gradient function e is created, which is proportional to the negative derivative of the optimization.

$$e = \frac{Spend_{ref} - Spend}{Spend_{ref}} - K\frac{CPC}{CPC_{nom}} \tag{4.9}$$

Minimizing this optimization function can be implemented as a feedback controller with the gradient as an error signal. This error function will focus on minimizing the relative reference error when this is large, and focus on minimizing

CPC when the relative CPC is large. The constant K can be chosen arbitrary but it could be set to make the parameters $Spend_{ref}$ and CPC_{nom} meaningful. The static point is given by $e = 0$ (presumably the controller still has integral action for this application). At this point the relation $\frac{Spend_{ref} - Spend}{Spend_{ref}} = K \frac{CPC}{CPC_{nom}}$ will hold. If $CPC = CPC_{nom}$, the relative reference error will be equal to K, and so K can be set to reflect a general relation between relative CPC and error for a solution to equation 4.8.

5

Max profit optimization

5.1 Expected profit optimization function

Lets assume that our optimization goal is to maximize our profit for every impression, i.e, $V = E(value - cost)$. We can look at a section i of all available impressions with a relatively homogeneous CTR p and win probability function $w(b)$ which is only dependent on the bid b. Assuming a first price auction, the expected cost of an impression is exactly the bid b. We define the value of the impressions through the expected clicks it will produce and a user defined click value v. The value function for a section with win probability $w(b)$ and click through rate p can be written as

$$V = w(b)(pv - b). \tag{5.1}$$

A question of interest is how V changes with our bid b. The derivative of the value function in bid is obtained through the chain rule.

$$\frac{dV}{db} = \frac{dw(b)}{db}(pv - b) - w(b) = \frac{dw(b)}{db}m - w(b)$$
$$dV = dw(b)m - w(b)db \tag{5.2}$$

Here m stands for the difference between the value pv of the impression and our bid b. In order to have positive profit m must always be positive. The derivative in Equation 5.2 can be interpreted as follows: Raising the bid by db will have two effects on the value function. The win probability function will increase by $dw(b)$ resulting in $dw(b)m$ more expected value. The cost of the impression will increase by db resulting in a decrease of the value function by $w(b)db$. An extreme point may exist in $\frac{dV}{db} = 0$ which is defined by $w(b) = \frac{dw(b)}{db}m$. By proving that V is concave in b, we prove that this extreme point exists, and that it is the global maximum. This will also prove that this extreme point is unique and plausible to find by means of gradient ascent.

5.2 Is the profit optimization function concave?

A function $f(x)$ is concave if $\frac{d^2 f(x)}{dx^2} \leq 0$. The second derivative is calculated from Equation 5.2 by using the chain rule once again.

$$\frac{d^2V}{db^2} = \frac{d^2w(b)}{db^2}(pv-b) - 2\frac{dw(b)}{db} \tag{5.3}$$

Any satisfying solution must satisfy $0 < b < pv$, so it can be concluded that $pv-b$ is strictly positive in profit maximization. $\frac{dw(b)}{db}$ should always be larger or equal to zero since a larger bid should always be expected to win more auctions. All the positive terms are moved to one side to create an expression for V to be concave.

$$\frac{d^2w(b)}{db^2} \leq \frac{2}{pv-b}\frac{dw(b)}{db} \tag{5.4}$$

The problem with proving this is that we do not have any information about the second derivative of a generalized win probability function. It can be positive or negative at different bids and it is not necessarily finite.

Even though the profit optimization function is never proven to be concave, it is assumed to suitable for gradient ascent. Any algorithm can only estimate this function, and so it would be sufficient that the estimated optimization function is concave.

5.3 Profit gradient ascent

Gradient ascent is an iterative method for finding a maximum to a weight function, by always moving in the direction of the gradient. This is implemented with the update equation

$$\begin{aligned} b(k+1) &= b(k) + \mu g(k) \\ g(k) &= \frac{dV}{db} \end{aligned} \tag{5.5}$$

with μ being the step size. How should the gradient be approximated? Perhaps using $dV = V(k) - V(k-1)$, $db = b(k) - b(k-1)$? This would probably not work since V could be time varying and unpredictable. Also $g(k)$ could increase without bounds for $b(k+1) = b(k)$.

Perhaps a better solution would be to use the theory from Section 5.1. In order to use Equation 5.2 to approximate the gradient we need to approximate the click through rate p, the win probability $w(b)$ at the current bid, and the derivative of the win probability $\frac{dw(b)}{db}$ at the current bid. This would require some modeling of the auction. This will be discussed further in Chapter 7.

5.4 Global profit maximization

Maximizing the profit for every impression can be seen as local optimization. The global optimization will be to maximize the profit for the entire campaign. Global maximization will produce positive profit for every impression and will satisfy global optimization. But when there is a budget involved local maximization does not necessarily translate into global.

The optimal global solution would make sure that the budget is spent only on the impressions that have the best profit relative cost. With the right v the local optimization might come close to that. If there is little to gain in an section of an auction the local optimization will bid very low, which will give other sections more budget. It is fair to say that the local optimization might produce good results globally. It could be possible to adjust v with a controller like in maximum value optimization.

Profit maximization is difficult to implement since it requires many models. Creating these models requires much data which can be hard to store. It would also be hard to verify if these models are correct or not.

6

Dynamic Bidding

6.1 Introduction

Dynamic bidding means changing bids during time. This can be done through a control system. The control system might be used for campaigns and agents with a wide range of bids and budgets. It is therefore very important that the control system is always robust and stable in every scale.

And even though the algorithms are generally designed for long campaigns they should also be able to perform okay in short campaigns. Making the algorithm scale invariant can be seen as removing units. A parameter set should be found, that are unitless, and produce good results for every scale and system, without risking instability or undesired oscillations.

6.2 Moving average filter

A moving average filter (MA filter) is a low pass filter that averages the signal it receives. The number of samples it should use in that average is determined by the length N. The filter can be used to reduce variance and/or remove certain frequencies from a signal. It will be used to remove the periodic behaviour of the market.

A MA filter of length N is defined through
$y(n) = 1/N \cdot (x(n) + x(n-1) + ... + x(n-N+1))$. Using the z transform $zx(n) = x(n+1)$ and geometric sums we can write a filter $W(z)$ with length N as

$$W(z) = \frac{1}{N} \sum_{k=0}^{N-1} z^{-k} = \frac{1}{N} \frac{1-z^{-N}}{1-z^{-1}} \tag{6.1}$$

6.3 Control goals

In this section a couple of goals for the control system will be presented. The system is not a classical control system and neither are the control goals.

- No static error between daily spend and reference daily spend.
  An integral (I) controller can be used to make sure that there is no static gain.
- The 24h oscillations from market should not affect bid level. A 24 hour MA filter can be used to remove these oscillations.
- Low bid level overshoot from market or reference changes. In order to keep the overshoot low, the controller gain cannot be too high.
- Reasonable fast bid level adjustment (max 3 days) from market or reference changes. To make sure that the system is not too slow, the controller gain cannot be too low.
- Stable system with no oscillations. A low controller gain ensures a stable system.

The process is time varying and oscillating by nature and the output (spend) will also be. The market oscillations are caused by different amount of people on the internet through the day. The control design will not focus on the spend signal itself but the filtered Daily spend, and control signal (bid level). For dynamic characteristics it is more interesting to look at bid level than daily spend. The daily spend is a filtered signal with high latency and it is not desired to make it fast since that would require serious overshoot in control signal. In a classical control system where the goal is to minimize the square error the filter might have been redundant as the integrator already possesses low pass characteristics. A P part could have been used to handle overshoots and improve the speed of the system. Simulation results suggested that the system worked better without the P controller so this was never implemented. This is partly because the system needs to remain slow, and there is also no obvious starting point to the bid level.

6.4 CPC-control system

Figure 6.1 shows a general control flow diagram of the CPC controller. The regulator adjusts the control signal u, which can be interpreted as the current value of a click, in order to follow the desired daily spend. The spend is summed over 24 hours because we know that the market changes over the day and we want to limit the variance of the signal sent to the regulator. The final bid B is drawn from a bid distribution $f(b)$ which depends on the nominal bid b. The distribution and its relation to b could be chosen arbitrarily but it would be wise to choose $f(b)$ so that $0 = f(b,x)$, for $b > 0$, $x < 0$, $E(f(b)) = b$ and $f(b,x) \to 0$ when $x \to \infty$. There are no real error, or disturbances in this system. The only unknown is the stochastic behaviour of the market.

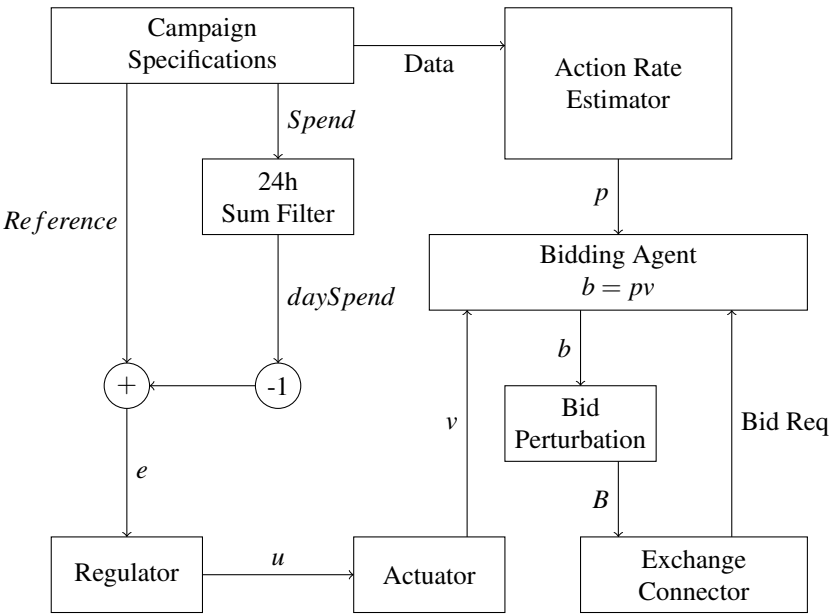


Figure 6.1 Control flow diagram of the CPC-controller

6.5 Control implementation

It is desired to find an analytical expression for controller gain that satisfies the goals at any scale. For this analysis we can simplify the system to a I controller connected to a static plant with gain $f(u) = ku = y$. The output from the plant is filtered through a 24 h sum filter which is the same as an MA filter with length N times a factor of N, where N is the number of periods during 24 hours. Figure 6.2 shows a flow diagram of the simplified system. The z transform of the filter is found with the help of Equation 6.1.

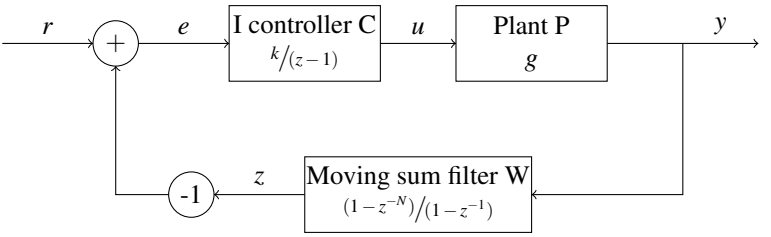


Figure 6.2 Control flow of the simplified system

The transfer function $G(z)$ of the closed system r->y can be written as

$$G(z) = \frac{CP}{1+CPW} = \frac{\frac{gk}{z-1}}{1+\frac{gk(1-z^{-N})}{(z-1)(1-z^{-1})}} = \frac{gk(1-z^{-1})}{(z-1)(1-z^{-1})+gk(1-z^{-N})} \tag{6.2}$$

It can be observed that we are only interested in the product of plant and controller gain which we can use as $gk = c$. In order to analyze the characteristics of the closed loop system we look at the poles of $G(z)$. These poles are found by the z that solves

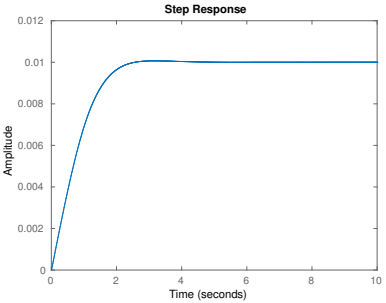
$$(z-1)(1-z^{-1})+c(1-z^{-N}) = 0 \tag{6.3}$$

which is the same as solving

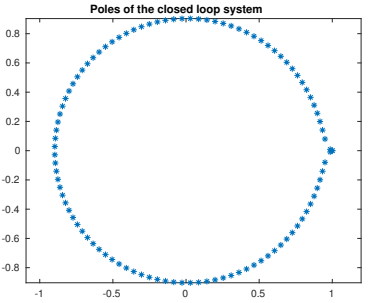
$$1-z^{-1}(2-c)+z^{-2}-cz^{-N-1} = 0. \tag{6.4}$$

Intuitively, and from simulations, we find that the system is more stable and with less oscillations when the combined gain c is small. We might expect a stable discrete time system with no oscillations to have poles p close to the origin. Through simulation it was found that in order to keep system characteristics, the controller gain k had to be proportional to the inverse of the length of the sum filter squared. The controller gain can be written as $k = r/N^2$. r is a constant which can describe the system well, regardless of filter length. Equation 6.4 is hard to analyze by hand. Figure 6.3 shows the poles of the closed loop system, plotted on the complex plane for $N = 100$ and different values of r.

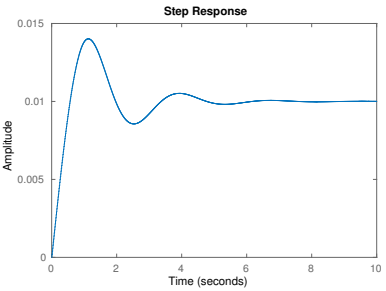
The plot shows 100 poles evenly distributed in a circle which is only marginally smaller than the unit circle. There are a couple of poles very close to 1 which is added by the integrator and/or MA filter. It is these poles that make the system unstable for large r. Smaller r makes the circle smaller, which indicates that the system is more stable. When r is very large, the system becomes unstable, which can be seen in the step responses in Figure 6.3. From the figures it can be seen that r should be set close to 1 for good performance.



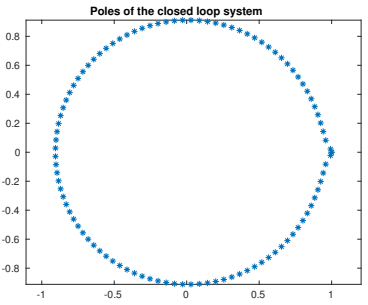
(a) Step response for r = 0.8



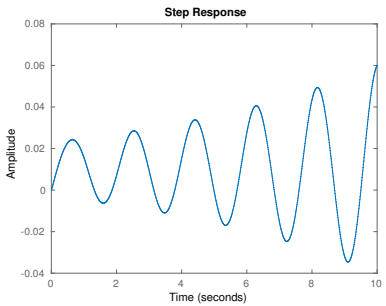
(b) Poles for r = 0.8



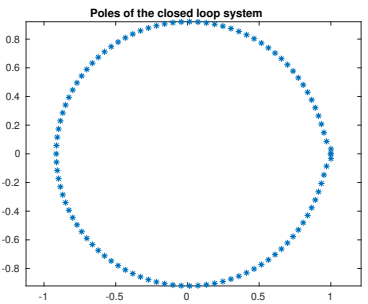
(c) Step response for r = 2



(d) Poles for r = 2



(e) Step response for r = 6



(f) Poles for r = 6

Figure 6.3 Poles and step responses for three different system gains

6.6 Actuator

The actuator converts the control signal into click value. This operation turns the signal into the correct unit. This could be done with a linear actuator $v = Ku$. This choice might not be optimal since only positive values are valid, which means that the control signal will have to be strictly positive. A more suitable choice might be

$$v = v_0(1 + Ku) \tag{6.5}$$

with v_0 being a nominal starting value. This would probably work, but there is still a chance of negative v if u is below $-K$. From Section 2.11 we have made the conclusion that the bidding space is logarithmic so a more suitable actuator might be

$$v = v_0 e^{Ku}. \tag{6.6}$$

It is important that K is sufficiently small so that v does not risk getting extremely high. The first order Taylor series of Equation 6.6 is Equation 6.5 i.e $v_0 e^{Ku} \approx v_0(1 + Ku)$ for small Ku. This means that the two actuators are almost identical for small control signals.

6.7 CPC and error controller

A controller minimizing the optimization function

$$V = \frac{w_1}{CPC_{nom}} CPC^2 + \frac{w_2}{Spend_{ref}} (Spend_{ref} - Spend)^2. \tag{6.7}$$

has very similar characteristics as a controller that minimizes the error squared. The negative gradient will be

$$f = (Spend_{ref} - Spend(b))/Spend_{ref} - Kv/CPC_{nom}. \tag{6.8}$$

CPC has been exchanged with v since it should theoretically decide the effective CPC. K can be set arbitrarily, but $K = 0.1$ has shown good results. Just like in the previous control system the negative gradient will be fed into an I controller with gain k and input into an actuator. This controller is harder to evaluate analytically, but some parallels can be drawn to the original control system. A lower control gain increases stability of the system but reduces speed. When v is not very large, this control system will be very similar to reference tracking system, only differing in the scalar $1/Spend_{ref}$ which is constant.

The analytical theory in Section 6.5 can be used to show that the controller gain k should be roughly proportional to the inverse of the MA filter length N. This gives $k = r/N$, and r will have roughly the same characteristics as seen in Figure 6.3. This is true if the gain g of the plant is equal to $Spend_{ref}/N$. The simulations in

Figure 6.3 used the assumption that $g = 1$, which means that the term $1/Spend_{ref}$ can be ignored in the gradient, and $1/N$ can be removed from the controller gain. We cannot know the true gain of the plant but it would be reasonable to assume that it is in the order of $Spend_{ref}/N$ since that would be its output at a steady state solution. And it is fair to assume that the gain relative to its output should be close to one.

The final control gain was adjusted by the help of simulations. It was true that it should be inversely proportional to the length of the MA filter with $k = r/N$. $r = 0.5$ gave bests simulation results, which resembles the theoretic results.

7

Auction modelling

7.1 Introduction

For some applications we might be interested to know what kind of auction we are participating in. It is necessary in finding the bid that optimizes profit. This section will show how such a model can be made. The idea of modelling an auction is presented in both [Fernandez-Tapia, 2017], [Zhang et al., 2016] and [Cui et al., 2011]. This generally leads to very complicated methods that are difficult to implement.

The bidders in RTB only have access to some data which makes the modeling of an auction much harder. The bidder only knows at which prices they have lost an auction and what prices they have paid for the winning auctions.

The auction is defined by one cdf $w(b)$ which defines a probability of winning at a certain bid, and pdf $f(b)$ which defines the change in win probability at the bid b. From the set of winning prices, and lost bids, the algorithm will try to estimate $w(b)$ and $f(b)$ for the current bid b. Three general modeling methods are presented.

7.1.1 Maximum likelihood method

The maximum likelihood method is a commonly used method in mathematical statistics. Before it can be used a model with some uncertain parameters has to be proposed. The method will find the parameters that are most likely to produce the given realizations.

One disadvantage with this method is that we do not know what kind of distribution describes the auction dynamics best, and we might lose generality when assuming a distribution. Another disadvantage may be that we are not interested in the entire function $w(b)$, only the value and derivative at the current bid b. The estimation of the entire function might be unnecessary expensive and may compromise precision at current b.

7.1.2 Direct observation

Given that there is enough data around the bid b we can estimate the win probability by $w(b) = \frac{\sum_W \delta(p_i - b)}{\sum_W \delta(p_i - b) + \sum_L \delta(B_i - b)}$, i.e the rate of wins and total bids at b. The deriva-

tive could be approximated with $\frac{dw(b)}{db} = \frac{w(b+db)-w(b-db)}{2db}$ where db is a very small change in bid. Since bids can have almost unlimited precision there is little chance we could ever have enough data to make this approach work. We would have to modify the δ function to have finite precision i.e. $\delta(\varepsilon) = 1$ for a small ε. The next method is a version of this idea.

7.1.3 Weighted observation

This solution uses the same idea as the previous solution. In order to make it work with limited data we look at all data points. The data points are weighted with bid difference. The closer the bid is to the current bid, the more relevant the data is. We exchange $\delta(x)$ with a weighting function $f(x)$ which has its maximum at $f(0)$, is symmetric and is strictly decreasing for x (and minus x). $f(x)$ is always strictly positive. The new estimation will be $w(b) = \frac{\sum_{wins} f(p_i-b)}{\sum_{wins} f(p_i-b)+\sum_{losses} f(B_i-b)}$.

7.2 Using the normal distribution for identification

The weighted observation method is chosen since it is general and does not require a lot of data. The normal distribution is chosen for weighting. The strength in the normal distribution is that it is well defined, smooth and differentiable. It is defined by its mean μ and standard deviation σ. For a value x, the pdf is given by

$$\varphi(x,\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{7.1}$$

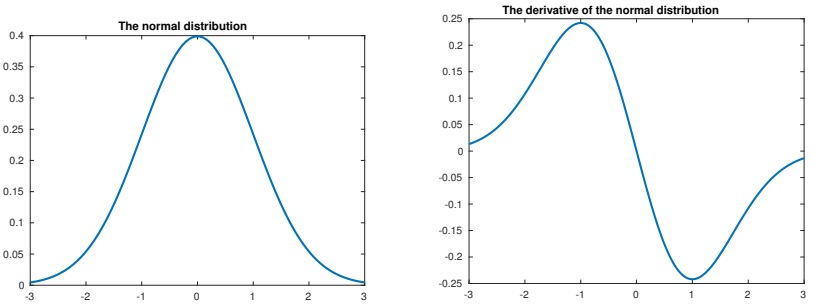
The win probability estimation will be given by $w(b,\sigma) = \frac{W(b,\sigma)}{W(b,\sigma)+L(b,\sigma)}$, where $W(b,\sigma) = \sum_{wins} \varphi(b,p_i,\sigma)$ and $L(b,\sigma) = \sum_{losses} \varphi(b,B_i,\sigma)$. σ is now included into the definition of the win probability estimation to highlight the fact that the estimation is dependent on the width of the weight function. Ideally we want a very low variance so that we get good local precision, but this only works if we have sufficient local data. Using a small variance in a area with little data could result in a spiky win estimation with potentially negative pdf. Using weight with very high variance will result in a very uniform probability estimation with little details.

In order to estimate the gradient of the win probability we can use the relation

$$\varphi'(x,\mu,\sigma) = -\frac{(x-\mu)}{\sigma^2}\varphi(x,\mu,\sigma). \tag{7.2}$$

Here the dot in a function $f'(x,y)$ is used to define the derivative of the function in the first variable. In this case it represents the operation d/dx. Figure 7.1 shows the pdf and derivative of the unity normal function i.e. $\varphi(x,0,1)$.

We can define the derivatives
$W'(b,\sigma) = \sum_{wins} \varphi'(b,p_i,\sigma) = \frac{1}{\sigma^2}\sum_{wins}(p_i-b)\varphi(b,p_i,\sigma)$ and



(a) The probability density function of the normal distribution

(b) The derivative of the normal distribution probability density function

Figure 7.1 The unity normal distribution and its derivative

$L'(b,\sigma) = \sum_{losses} \varphi'(b,B_i,\sigma) = \frac{1}{\sigma^2}\sum_{losses}(B_i - b)\varphi(b,B_i,\sigma)$. Using the quotient derivation rule on $w(b,\sigma)$ we get

$$w'(b,\sigma) = \frac{W'(b,\sigma)(W(b,\sigma)+L(b,\sigma)) - W(b,\sigma)(W'(b,\sigma)+L'(b,\sigma))}{(W(b,\sigma)+L(b,\sigma))^2}. \quad (7.3)$$

From this estimation we can estimate the gradient of the value function, and a suitable bid change from equations 5.2 and 5.5.

8

Max value algorithm

8.1 Introduction

This chapter presents an algorithm that maximizes the total value. This minimizes the optimization function

$$V(b) = \frac{w_1}{CPC_{nom}} CPC(b)^2 + \frac{w_2}{Spend_{ref}} (Spend_{ref} - Spend(b))^2. \quad (8.1)$$

while distributing the spend proportionally to the estimated CTR. The optimization function is a weighted sum of the squares of relative error and CPC. The same algorithm could be used for only maximum value optimization in which the optimization function can be reduced to relative error squared. This is achieved by setting w_1 to 0.

The reason for choosing Equation 8.1 as optimization function is that it limits the size of the CPC. Figure 8.1 shows the structure of the algorithm. Arrows indicate functions/programs being called with the inputs/outputs named. The algorithm utilizes 5 static and 3 dynamic core variables. The step size is always 0.5/96 (96 is the number of controller calls per day), and the relative standard deviation (RSTD) is always set to 0.1. These parameters have proven effective in simulations. nomCPC, and reference is set for every agent to suit the goals. The bidLevel is initialized to nomCPC and the integrator and Spend start off at 0. The agent gets a set of available creatives for which it will buy ad slots.

The algorithm consists of two main parts. The controller part is called once every 15 minutes. This adjusts the bid level to optimize performance. The bidding is the second main part of the algorithm. This is called every time a bid request is received, which happens at a rate of approximately 10 times per second. This part calculates and returns an optimal bid for each bid request. The calculations are generally fast, so the biggest real time problem is server communication.

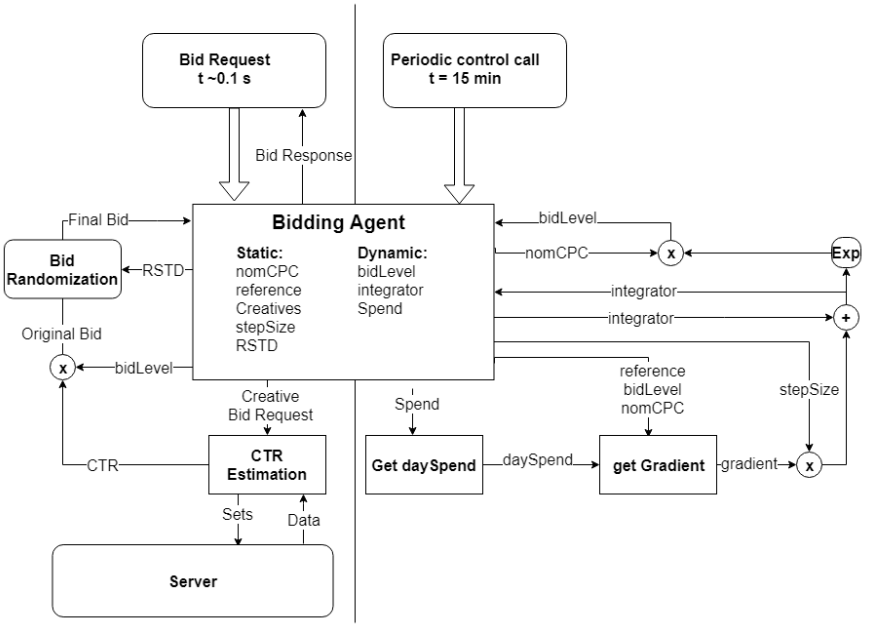


Figure 8.1 Diagram of the value optimization algorithm

8.2 CPC Controller

The controller is a simple integrator with exponential actuator. The controller seeks to minimize the Function 4.8 which is very close to a reference follow problem. The control system can also be interpreted as exponential gradient descent for this function. The controller uses 3 states, or dynamic variables: Spend, integrator and bidLevel. Spend is total spend by agent, integrator sums up the gradient, and bidLevel is the current level at which the bidding function will bid.

The first part of the controller calculates the spend for the last 24 hours, called daySpend. This is calculated by Spend(k)-Spend(k-96). This is essentially a MA filter, described in Section 6.2, times a factor. If the agent is younger than 24 hours this operation is not possible and is replaced by (Spend(k)-Spend(k-1))*96, which essentially removes the MA filtering for the first 24 hours. The daySpend is used to calculate the gradient of the optimization function. The gradient is given by

$$e = \frac{reference - daySpend}{reference} - 0.1\frac{v}{nomCPC}. \tag{8.2}$$

Since the weights can be set arbitrarily, the gradient can be set to remove constants and prioritize reference tracking to CPC with a factor 10 to 1. The gradient

is multiplied by the stepSize and added onto the integrator. The new bid level is calculated by $bidLevel = nomCPC \cdot exp(integrator)$.

There are not any major real time problems in the controller part of the algorithm. The controller is called once every 15 minutes and there is no time limit on the process. The process does not require any contact with remote servers or heavy calculations. In practical implementations it is important that the bidding agent is not affected by any outer controllers. Many systems temporarily block agents, or decrease traffic to them, when they spend money too fast. This is problematic since the controller cannot identify the reason for the low traffic and the feedback is removed. This might cause the controller to stay at a too high bid level, and might introduce slow dynamics and limit cycles.

8.3 Bid section

When a bid request is received the bidding agent calls the bidding section to calculate an optimal bid for the ad slot and a chosen creative. The click through rate is estimated. This is done by placing the impression into sets, collecting data from the servers, and weighting these. Since there are many possible sets, the data is too large to be collected locally and has to be collected from the servers for each bid. The estimated CTR is multiplied with the current bid level to produce a bid. This bid is randomly perturbated by a normal distribution with fixed relative standard deviation for the final bid. This is done to increase exploration and controller stability. The final bid is put into a bid response and returned. The theory behind the CTR estimation is described in Chapter 3

This part of the algorithm is very real time reliant. The SSP expects a bid response, no more than 100 ms after their bid request is sent. The bidding logic is not recommended to use more than 30 ms in order to make this possible. There are no heavy calculations in the bidding logic so that part should not take up too much time. The server call, which is executed for every bid, might take some time though. Depending on the location, quality and infrastructure of a server, a server call might take between fractions of ms to several seconds. Using the servers for every bid puts very high demands on the server infrastructure. For a global company this can be quite challenging.

8.4 Discussion

This algorithm was implemented into the Emerse DSP and tested live. It is not overly complicated and very stable. It does not require lot of data to run, but might need a lot of data to produce good results.

9

Max profit algorithm

9.1 Introduction

This algorithm maximizes the profit for every impression. This is done by maximizing

$$V(b) = w(b)(pv - b). \tag{9.1}$$

The algorithm needs to estimate a CTR p for each section. The bid for this section is adjusted to maximize the optimization function in Equation 9.1. In order to do this, it needs to model the auction for each section while it is running. Figure 9.1 shows a diagram of the profit maximizing algorithm. The algorithm has 4 static variables creative, v, stepSize and RSTD. Creatives are a list of creatives that are available. v is the value of one click. stepSize is the step size used for gradient ascent, and RSTD is the relative standard deviation of the bid randomizer. There are 3 dynamic variables. Bids is a dictionary of the current bid for each section. Wins and Losses are a list of wins and losses for each section. This algorithm also consists of two parts. The bidding section returns the optimal bid, and the model section adjust these bids. The calculations for this algorithm would be relatively fast, the biggest time would be spent on server communication.

9.2 Model section

This section is called whenever one section has sufficient data. This might happen once every 15 minutes. The algorithm uses this data to update the bid for this section. It starts with estimating the CTR. This is performed by a server call and the thoery from Chapter 3. The cdf and pdf of the auction is modeled from the wins and losses for this section and uses theory from Chapter 7. The cdf, pdf, p and v are used to estimate the gradient. This is done by

$$g = cdf(pv - b) - pdf. \tag{9.2}$$

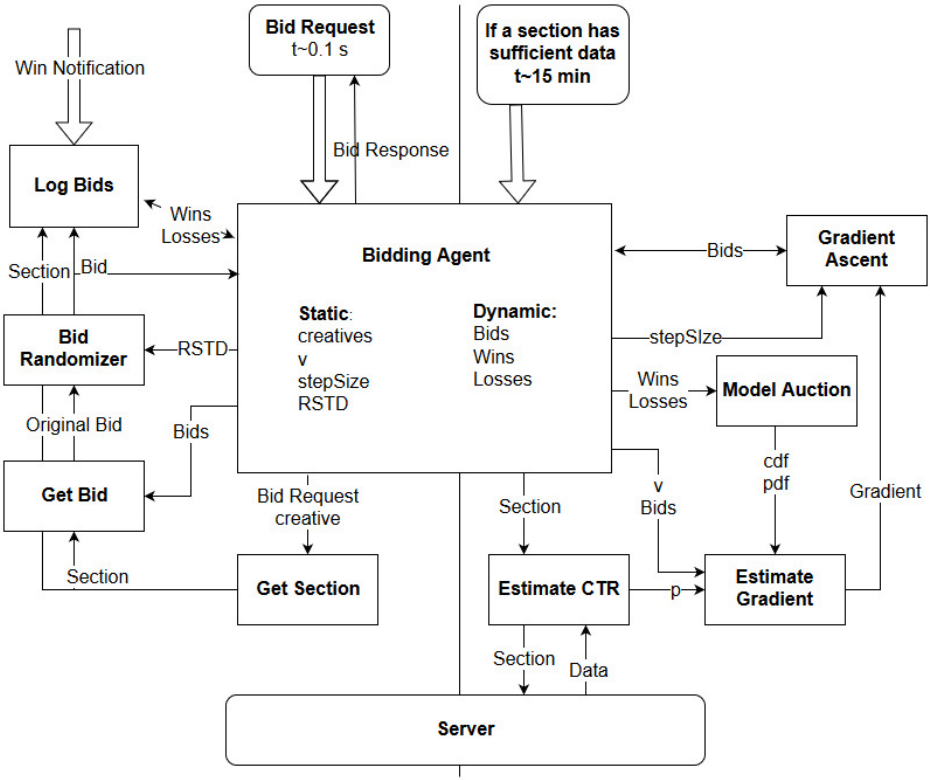


Figure 9.1 Diagram of the profit maximizing algorithm

Here b is the bid for the current section which is found in Bids. The gradient Ascent updates b by $b_{new} = b \cdot exp(g * stepSize)$, and the new b is stored in Bids. Losses and Wins are cleared for this section after the bid is updated. Depending on how large the data sets Wins, and Losses are this section might be slow. The auction modelling is a rather simple method, it only sums up values from a normal distribution, but it might not scale well for very large sets. It is also not clear how often this part of the algorithm would be called in a live environment.

9.3 Bidding section

This section returns the bid request with an optimal bid. The algorithm classifies the impression as part of one section. The current bid for this section is extracted from Bids. This bid is randomly perturbed to produce a final bid. This is sent back in a

bid response.

If a win notification is received from the SSP, the price is stored in the Wins dictionary. Else the bid is stored in the Losses dictionary. When there is enough data in the dictionaries, the bid gets updated by the model section. This section can be really fast, as there is no server communication needed. However, it might be hard to track the bids that have been sent and log these.

9.4 Discussion

This algorithm has performed well in simulations. However it becomes quite clear that it would be very difficult to implement in a live environment. Generally there is no limit on how many sections that an agent can bid on. Storing data for all these sections would be inefficient and probably impossible. Even connected to a server the amount of data would increase too fast.

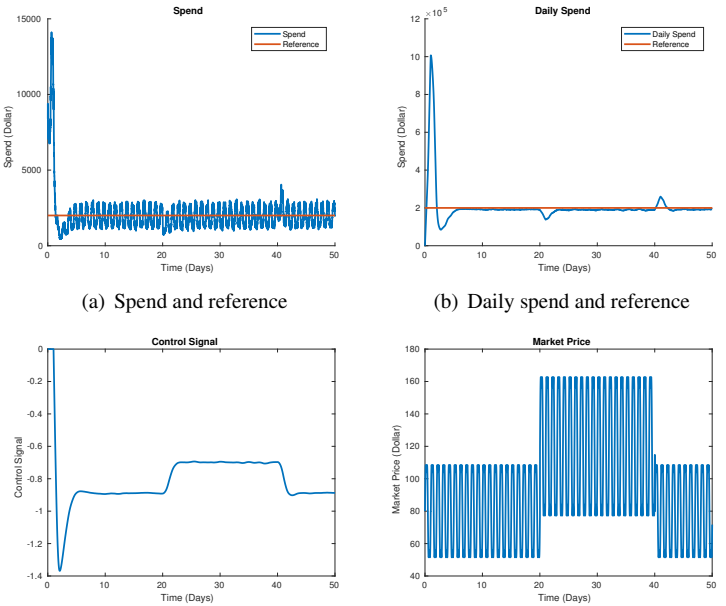
The modeling of auctions might also be inefficient and hard to evaluate if there are too many sections. Many sections would probably only have a few wins, and so a good model could never be made or needed. The algorithm is not suited for the general campaign. It could, however, work for a campaign with very few specific sections. Then the data might be reasonable and the models could be evaluated.

10

Results

10.1 Value optimization simulation

Figure 10.1 shows a simulation for the value optimization algorithm performed in Simulink. The model in Simulink uses the same logic and parameters that were implemented in the DSP and should be a good representation of the dynamics that would be expected in a live environment. The algorithm is connected to a plant that resembles an auction.



(a) Spend and reference

(b) Daily spend and reference

(c) Control signal

(d) Market price

Figure 10.1 Simulation of value maximization in oscillating market

The auction is decided by the market price (d) and a random stochastic process. The market price oscillates with a frequency of 24 hours and makes two step responses. The control signal (c) is not largely affected by the oscillations and handles the step responses with minimal overshoot. It takes the control signal 2-3 days to settle, which is to be expected when combining an integrator with an MA filter.

The spend (a) and daily spend (b) follow the reference with a very small static error. This is to be expected from the second part of the optimization function. The spends are only affected during 1-2 days by the step responses, and only (a) reacts to the oscillations.

10.2 Profit maximization Matlab simulation

This section will present Simulink simulations for the profit gradient ascent algorithm. The bidder sends 10 randomized bids at a bid level and receives data only showing which of the bids are won. From this data the bidder tries to approximate the auction, and the corresponding optimization function. The bid level is moved along the gradient of the estimated optimization function. For every simulation the value v of the impression is set to different values. Every simulation bids on a total of 10^6 impressions.

Figure 10.2 shows results from Simulink simulations using three different models. All models are normalized so that there is a 50 % win rate at bid 1.0 and that the optimal bid has a win rate of 2.4 %. The same parameters are used in all simulations. The auctions are different for the three simulations, with increasingly narrower optimization functions.

The optimal bid for model 1 lies in 0.025. The mean bid level is 0.022 while the mean of the won bids is 0.024. From this simulation the total profit is 90 % of the profit that would be expected from the optimal bid. The optimization function is shown in (a) and the bid levels are shown in (b).

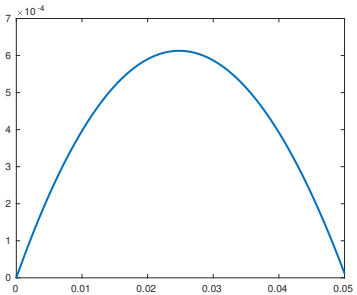
The optimal bid for model 2 is 0.078, with mean of bid levels at 0.074 and mean of wins at 0.083. The simulation for this model generates 83 % of optimal profit.

The optimal bid is given by 0.65 and the mean bid level is 0.62. It is worth noting that the mean win is 0.87, which is quite higher than the value of the impression. The randomized bidding makes it possible to have a higher mean win than the value and can be categorized as the winners curse. This causes the profit to be -222 % i.e the bidder loses money.

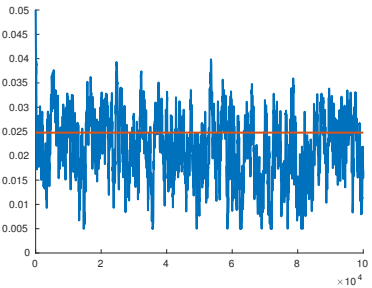
In general we can see that the estimations have a consistent bias of around 5 %. Since this bias is consistent it is not a problem. In general the winners curse cancel out the effect of low estimation since the winning bids tend to be higher than the average.

It is interesting to see that model 1 and 2 have really good relative profit and that model 3 has negative profit. Looking at subfigures (b), (d) and (f) we see that model 3 has a very low variance estimation while 2 and 1 are not as good. These results

suggest that the current algorithm has problem with getting profit from narrow auctions, but since the estimations are good, there is a lot of potential. The algorithm needs some automatic adjustments to balance up exploration (vs exploitation) cost.

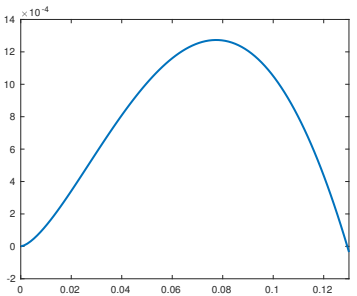


(a) Optimization function for bid levels model 1

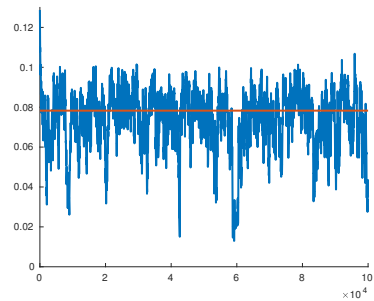


(b) Bid levels model 1

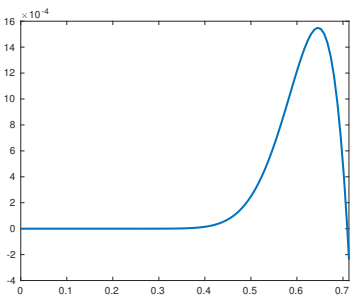
,



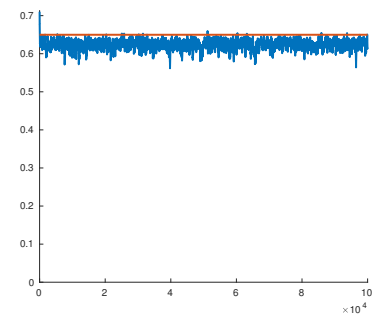
(c) Optimization function for bid levels model 2



(d) Bid levels model 2



(e) Optimization function for bid levels model 3



(f) Bid levels model 3

Figure 10.2 Simulation results from 3 different models

10.3 Live test 1

10.3.1 Performance

The first live test was started the 16th of May 2018. It lasted almost seven days. An implementation of the value optimization algorithm was run alongside a fixed price agent. Both were managed and controlled by and through the Emerse platform. Creatives were taken from a large campaign with a lot of data, and relative good performance. The agents were set to have identical configuration and creatives. Both agents were set to spend 35 dollars each during the duration of the test campaign which was 6.5 days.

Nominal CPC was set to 0.2 for the optimization agent, and the fixed price agent was set to bid at 0.4 CPM, which translates into 400 μDollar per impression. The average CTR is roughly 0.002 so the two agents will start bidding roughly at the same level. The reason for running the fixed agent in parallel, is to have a benchmark to compare results with. Every creative and campaign and day are different, so there is no good way to quantify performance. The only scientific evaluation is to compare the results of two methods under identical circumstances. A low CPM fixed price agent only buys the cheapest impressions. It can achieve low CPC without any advance technology and is a good reference.

Table 10.1 shows the performance from this live test. The averages for the creatives used were 0.97 CPM, 0.51 CPC 0.19 CTR. It is clear that both agents manage to pay less per impression and per click, but have lower click probability. When optimizing towards clicks both agents are better than average for the creatives. It is important to note that the creatives have generally not been used to maximize clicks. The agents spent roughly the same amount of money, but the fixed price agent bought much more impressions. This is caused by the optimizing agent bidding too high which causes CPM and CPC to be higher for the optimization agent. The optimization agent also has a lower CTR than the fixed price agent which is unexpected. This suggests that the CTR estimations might have been ineffective. The fixed price agent spent the money more efficient during this test, but it had the advantage of higher traffic.

Table 10.1 Performance for optimization agent and fixed bidder

	Optimization Agent	Fixed Bidding Agent
Spent (Dollar)	32.16	35.12
Impressions	64015	104839
Clicks	77	165
CPM (Dollar per mil)	0.50	0.33
CPC (Dollar per click)	0.42	0.21
CTR (percent)	0.12	0.16

It is also worth noting that the agent received more than 50 % clicks from non human sources, or bots. The DSP automatically removes them, and they have not been included in any results. But the optimization agent will include these clicks when estimating CTR. This could affect the performance of the agent and might cause the agent to target impressions that are likely to be bots, since it gets awarded for all clicks, bots or human.

10.3.2 Spend Control

Figure 10.3 shows the spend per hour of the optimization agent. Blue is the pure data, while red is 24h average. The yellow line shows a reference of 0.21 which is needed to spend the budget on time. The first thing to note is that the spend starts very low. Before T=1.5 the average spend is 0.1 Dollars per hour. This indicates that the flow of bid requests to the agent is low. The agent handles this by raising the bid level.

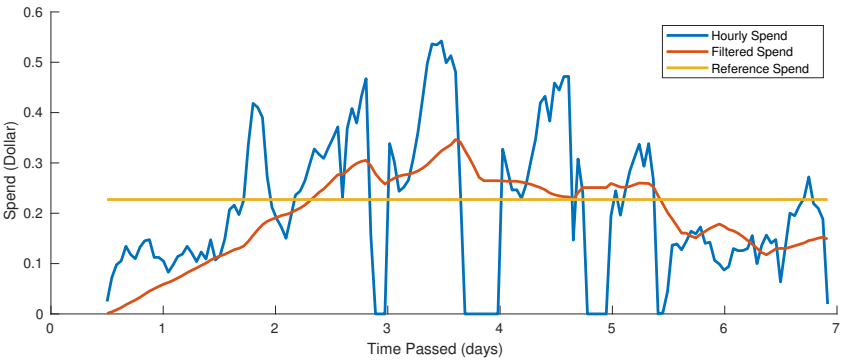


Figure 10.3 Spend flow for live test

The second thing to note is that the agent starts overspending after T=2. This happens because of two reasons. The controller in the algorithm has a slight overshoot, so temporary overspend is to be expected, and desired. During this experiment the reference signal was also changed automatically by the DSP. The DSP does this to encourage the agent to spend its entire budget, but cascading controllers like this can be dangerous so this was removed for the second live test. It can also be seen that the spend drops rapidly at T=5.5, this was also caused by reference changes from the DSP.

One problem with the live implementation was that the DSP automatically cut off traffic to the agent every time that it had spent its daily budget. This causes the spend to drop to zero at the end of days with large spend. At the end of the day the budget resets and the agent can spend money again. This block interferes with the agents inner controller and causes it to bid too high.

10.3.3 CTR Estimation

Figure 10.4 shows a histogram of CTR estimations of all impressions that the optimization agent bid on during the first live test. The distribution has a mean of 0.00140 and standard deviation of 0.00090. From this we can assume that the majority of CTR estimation are in the interval 0.0005-0.0023. There is no certain way to know if these estimations are correct. They have been weighted to slightly favour impressions with low data to ensure good exploration. This means that they might not represent the true CTR, but a measure of the value to the agent. The variance of the CTR estimations give a suggestion about the possible precision of the estimator and the potential of the optimization. The higher the variance, the better results the optimization agent can get, given that the estimations are correct.

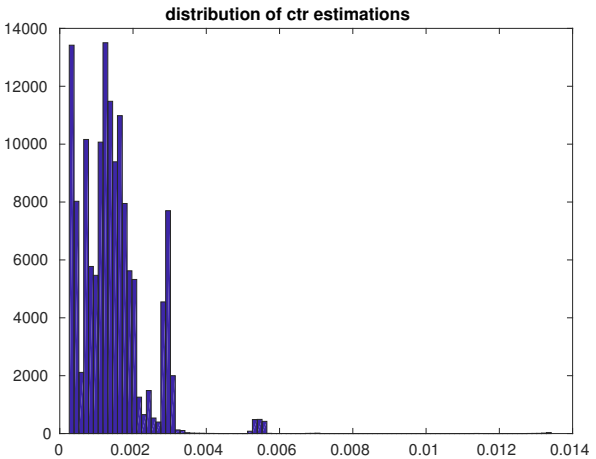


Figure 10.4 Histogram of CTR estimations for live test

10.3.4 Discussion

In this experiment the fixed price bidding agent outperformed the optimization agent. The biggest cause of this is that the flow of bid requests to the optimization agent was much too low. The flow of bid requests dictates how picky the agent can be, and is an important resource for successful optimization. The DSP changing reference, and blocking traffic cause the agent to increase the bid level higher than it had to, which caused the high CPC. The DSP blocks traffic to the optimization agent because it is spending money to fast. But the agent experiences the lack of traffic as a sign that it is spending money too slow. This mixed in with an integrator controller can cause the bid level to go up very high. The second part of Equation 8.1 keeps the bid level from becoming too high.

The low CTR indicates ineffective CTR estimation. This might have been caused by bot clicks in the data set, or too high exploration rate. In general you would expect the distribution to be smooth and resemble a gamma or log normal distribution. There are some big peaks, maybe periodic in the distribution, and an island at 0.0057. Some of this might be the effect of small data sets and weighting estimations, which give periodic appearances and bumpiness. It can also be a sign that the estimations are incorrect. It is also possible that some of the peak, especially the one at 0.0057 could caused by bot clicks, that cause unusually high CTR estimations for some impressions.

10.4 Live test 2

10.4.1 Performance

The second live test was run on a different set of creatives, and with an increase of traffic to the optimization agent. The averages of these creatives were 1.41 CPC, 1.84 CPM. There once again was an optimization agent at 0.2 nominal CPC run with a fixed price agent at 0.4 CPM with a goal spend of 5 dollars per day. The results for these agents can be seen in Table 10.2 Once again both agents are able to produce lower CPC levels than average for the creatives. The optimization once again had higher CPM, which was caused by DSP integration problems with the controller. Despite this both agents had close to identical CPC and CTR, which indicates that the optimization agent worked better in this test compared to the last one. Still this gives no conclusive evidence that the optimization agent is more effective.

Table 10.2 Results for optimization agent and fixed bidder

	Optimization Agent	**Fixed Bidding Agent**
Spent (Dollar)	18.15	20.71
Impressions	44915	55650
Clicks	24	28
CPM (Dollar per mil)	0.40	0.37
CPC (Dollar per click)	0.76	0.74
CTR (percent)	0.05	0.05

10.4.2 Spend Control

Figure 10.5 shows the spend flow for the optimization agent. During this test, the reference signal sent to the agent was kept static at 5 dollars per day, or 0.21 per hour. There was also an increase in traffic for the optimization agent. Still the agent has to increase bid level slightly to increase spend. Once again the agent is being blocked from traffic, which causes the bid level to increase when they should decrease, at T=2. From this experiment it could be concluded that the DSP blocks the agent before it can reach reference signal given by the DSP. This encourages the agents to spend their budget faster, but becomes a problem for an agent with an I controller. Despite of this the optimization agent was better at keeping an even spend than the fixed price agent.

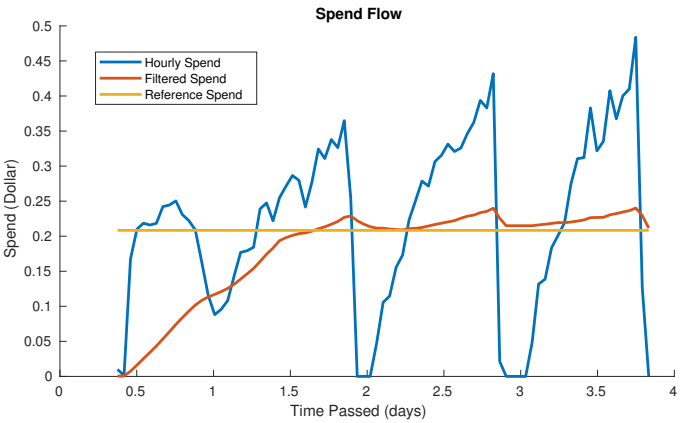


Figure 10.5 Spend flow for live test

10.4.3 CTR estimation

The estimations for the live test had a mean of 0.0013 and standard deviation of 0.00077. The estimations were more centralized in one island, but there were still periodic bumps. The performance of the optimization suggests that the CTR estimations for this experiment might have been more accurate. Perhaps the data was less polluted with bot clicks.

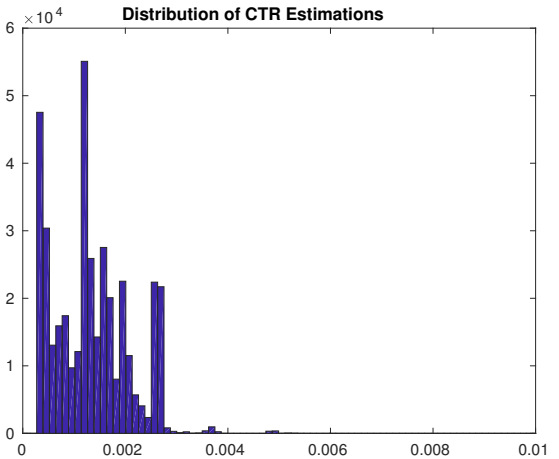


Figure 10.6 Histogram of CTR estimations for live test

10.4.4 Discussion

The optimization agent was more successful in the second experiment. There was still problems with the agent being blocked, which caused the bid level to be unnecessarily high. The CTR estimations might have been better, but the agent was not able to increase CTR noticeably. It is possible that a large scale experiment could have proven a difference between the two agents. It has to be remembered that all live tests are performed in a high variance stochastic environment, and no results can be deterministic. With around 25 clicks each both agents would have relative standard deviation of around 20 % $(1/\sqrt{25})$ which means that no results can be counted as conclusive.

10.5 Live test 3

10.5.1 Performance

The last live test was run on a third set of creatives. The averages of these creatives were 0.63 CPC, 1.23 CPM. There once again was an optimization agent at 0.2 nominal CPC run with a fixed price agent at 0.4 The results for these agents can be seen in Table 10.3. The optimization agent was set to spend 6.4 dollars per day, and the fixed price agent 3 dollars per day. The optimization agent was able to lower the CPM marginally while still spending the budget. Both agents had CPM below average. The optimization agent had very low CTR which caused it to have very high CPC. The low CTR is a product of the CTR estimator, and causes the optimization agent to be worse than the fixed price agent in this test.

Table 10.3 Results for optimization agent and fixed bidder

	Optimization Agent	**Fixed Bidding Agent**
Spent (Dollar)	15.05	9.49
Impressions	42323	25038
Clicks	18	20
CPM (Dollar per mil)	0.36	0.38
CPC (Dollar per click)	0.84	0.47
CTR (percent)	0.04	0.08

10.5.2 Spend Control

Figure 10.7 shows the spend flow for the optimization agent. During this test, the reference signal was 0.28 Dollars per hour. The bid level is increased slightly to match the reference signal and then seems to be steady. The test was too short to see the complete dynamics of the controller but it seems to stabilize after 2 days. For this experiment, the reference signal had been changed to avoid the agent getting blocked. It can be seen that the agent spends the budget continuously through the test. The bid level is kept steady through the day and all oscillations are from the market.

10.5.3 CTR estimation

The estimations for the live test had a mean of 0.0014 and standard deviation of 0.0077. The estimations seem to be very spread out, with many sharp peaks. The CTR estimation for this experiment was probably the worst of the experiments. The poor CTR estimations are probably caused by the agent prioritizing exploration. On these small tests the agent does not get any award from the exploration, so it might have been better to use a greedy estimation algorithm.

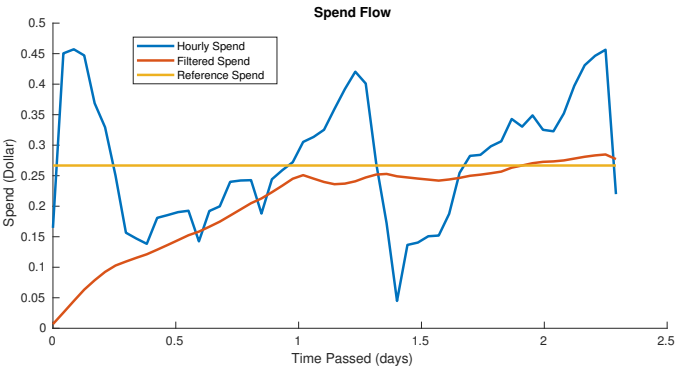


Figure 10.7 Spend flow for live test

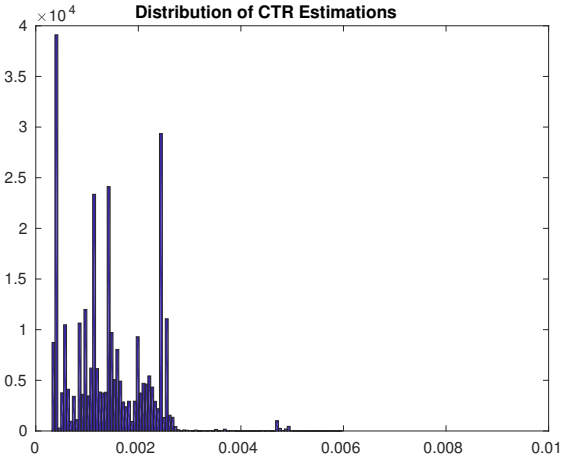


Figure 10.8 Histogram of CTR estimations for live test

10.5.4 Discussion

During this experiment the controller part of the algorithm worked good. It kept a low CPM and a even spend. The spend was, however, distributed on impressions with very low award. This was caused by poor CTR estimation. The CTR estimation might be poor because of too high exploration priority, or not enough data.

11

Conclusions

11.1 Theory

Real time bidding has proven a lot of potential for theoretical optimization and modeling. There are many possible optimization functions that could be used and large amounts of data that can be used to create models. There are lots of methods that could be used to create better CTR estimations. Finding a way to generalize features and data could improve CTR estimations and reduce the the data that need to be stored in servers.

Many of the theoretical optimization ideas prove to be too complicated to implement efficiently and practically. A lot of times there existed simple solutions that well approximated optimal solutions.

Like in reinforcement learning, the agents had to choose between exploration vs exploitation. It is hard to decide how these shall be valued, especially when there are multiple agents.

11.2 Simulations

The simulations all showed promising results. All simulations showed that the algorithms were stable and robust. The simulations were a good place to test that the algorithms were truly scale invariant. In a live environment it is important that an algorithm works exactly the same with a budget of 100 dollars as with a budget of 1 dollars. The simulations were limited since there is no way to know the true models behind an auction, or a person clicking on an ad. These have to be tested in a real environment.

11.3 Live experiments

The optimization agent was run in a live environment during several live experiments. The experiments were successful in that the agent worked the way it was supposed to. It estimated CTR and bid proportionally to it and adjusted the bid level

periodically. The agent was generally able to produce lower CPC than the average for the creatives used. This comparison is not very scientific since the campaigns of the creatives were generally not optimized towards minimizing CPC. For this reason a fixed price agent with a low CPM was run parallel to the optimization agent in the live tests. Since the experiments were small scale, most differences were not significant. Larger scale experiments could possibly reveal differences between the agents.

After some adjustments the control system worked as intended. It was able to adjust the bid level to spend the budget evenly. This control system would outperform a fixed price bidding agent with poor settings. The control system assures that the buyers do not pay more than necessary for impressions and would increase efficiency. This control system also makes sure that bid requests are handled evenly which could increase global efficiency for a DSP.

The evaluation of impressions through CTR estimation seemed to, sometimes, get lower value per impression than a fixed bid. This suggests that the CTR estimation were ineffective at evaluating an item. Larger experiments, with more chances to collect data might produce better results.

The current state of the algorithm cannot yet be proven to be more effective than a well optimized fixed price bidding agent.

11.4 Improvements

One of the biggest weaknesses and trials of the algorithms was the CTR estimation. This was never meant to be a large part of this thesis. The agents had access to data that had only been gathered for about a month. It is possible that collecting data for, maybe a year, could produce better results. The bot clicks would also need be removed from the data.

The exploration rate for the agents might have been too high to produce good results. A single agent trying to explore the all possible features of impressions might be ambitious. If many agents were doing it for a long time, it might have shown results. But for single experiments, it might have been better to have the agents be fully greedy, i.e, aim for short term results.

There were some difficulties in integrating the optimization agents with the DSP. The integral controller proves to be a problem when connected to an outer controller. The DSP can block the agent anytime, for any duration. In this duration the integrator might wind up very high. The DSP connection is good, however, since it keeps the agent from using money that it does not have.

Bibliography

Başar, T. and G. Olsder (1998). *Dynamic Noncooperative Game Theory, 2nd Edition*. Society for Industrial and Applied Mathematics. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611971132.

Cui, Y., R. Zhang, W. Li, and J. Mao (2011). "Bid landscape forecasting in online ad exchange marketplace". In: *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '11. ACM, San Diego, California, USA, pp. 265–273. ISBN: 978-1-4503-0813-7. DOI: 10.1145/2020408.2020454. URL: http://doi.acm.org/10.1145/2020408.2020454.

Fernandez-Tapia, J. (2015). "Real-time bidding rules of thumb: analytically optimizing the programmatic buying of ad-inventory".

Fernandez-Tapia, J. (2017). "Statistical modeling of vickrey auctions and applications to automated bidding strategies". *Optimization Letters* **11**:4, pp. 771–780. ISSN: 1862-4480. DOI: 10.1007/s11590-016-1045-1.

Hardin, G. (2010). "The tragedy of the commons". URL: http://pages.mtu.edu/~asmayer/rural_sustain/governance/Hardin$\%$201968.pdf.

IAB (2017). "Openrtb 3.0 framework". URL: https://iabtechlab.com/wp-content/uploads/2017/09/OpenRTB-3.0-Draft-Framework-for-Public-Comment.pdf.

Karlsson, N. (2016). "Control problems in online advertising and benefits of randomized bidding strategies". *European Journal of Control* **30**. 15th European Control Conference, ECC16, pp. 31 –49. ISSN: 0947-3580. DOI: https://doi.org/10.1016/j.ejcon.2016.04.007. URL: http://www.sciencedirect.com/science/article/pii/S0947358016300127.

Sluis, S. (2017). "Big changes coming to auctions, as exchanges roll the dice on first-price". URL: https://adexchanger.com/platforms/big-changes-coming-auctions-exchanges-roll-dice-first-price/.

Steven, S. L. (2010). "A modern bayesian look at the multi-armed bandit". *Applied Stochastic Models in Business and Industry* **26**:6, pp. 639–658.

Wittenmark, B. (1995). "Adaptive dual control methods: an overview". *IFAC Proceedings Volumes* **28**:13. 5th IFAC Symposium on Adaptive Systems in Control and Signal Processing 1995, Budapest, Hungary, 14-16 June, 1995, pp. 67 –72. ISSN: 1474-6670. DOI: https://doi.org/10.1016/S1474-6670(17)45327-4. URL: http://www.sciencedirect.com/science/article/pii/S1474667017453274.

Zhang, W., S. Yuan, and J. Wang (2014). "Optimal real-time bidding for display advertising". In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '14. ACM, New York, New York, USA, pp. 1077–1086. ISBN: 978-1-4503-2956-9. URL: http://doi.acm.org/10.1145/2623330.2623633.

Zhang, W., T. Zhou, J. Wang, and J. Xu (2016). "Bid-aware gradient descent for unbiased learning with censored data in display advertising". In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. ACM, San Francisco, California, USA, pp. 665–674. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939713. URL: http://doi.acm.org/10.1145/2939672.2939713.

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER'S THESIS

Date of issue
June 2018

Document Number
TFRT-6060

Author(s)
David Rådberg

Supervisor
Rasmus Larsson, Emerse, Sweden
Carl-Johan Grund, Emerse, Sweden
Karl-Erik Årzén, Dept. of Automatic Control, Lund University, Sweden
Martin Maggio, Dept. of Automatic Control, Lund University, Sweden
Anders Rantzer, Dept. of Automatic Control, Lund University, Sweden (examiner)

Title and subtitle
Optimal Real Time Bidding in Online Advertising

Abstract

This thesis explores some of the possibilities of demand side optimization in online advertising, specifically how to evaluate and bid optimally in real time bidding. Theory for many types of optimizations is discussed. The thesis evaluates auctions from a game theory and control theory perspective. It also discusses how big data sets can be used in real time, and how agents can explore unknown stochastic environments.

All items are valued through an estimated action probability, and a control system is designed to minimize the cost for these actions. The control system aims to find the lowest possible price per item while spending the entire budget. Periodic market changes and censored data makes this task hard and imposes low pass characteristics on the closed system. Using data to evaluate items is a high dimensional problem with very small probabilities. When data is limited the algorithm is forced to choose between low variance and precision. The choice between exploring and exploiting the unknown environment is crucial for long and short term results.

An optimization algorithm was implemented and run in a live environment. The algorithm was able to control the spend optimally, but distributed it suboptimally.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-69

Recipient's notes

Security classification