Technische Universität München

Department of Mathematics

Master's Thesis

Bidding Strategies and Market Price Prediction in Real-time Computational Display Advertising

Stefan Heidekrüger

Supervisor: Prof. Dr. Felix Brandt

Advisor: Prof. Dr. Martin Bichler

Submission Date: September 30, 2016

I hereby declare that this thesis is my own work and that no other sources have been used except those clearly indicated and referenced.

München, September 30, 2016

Zusammenfassung

Real-time Bidding (RTB) ist ein Segment des Onlinewerbungsmarkts, welches in den letzten Jahren signifikant an Wichtigkeit zugenommen hat. Durch die große Anzahl verschiedener Akteure im Markt wäre es in der Praxis unwirtschaftlich, dass Werbetreibende und Publisher direkt miteinander verhandeln. Im RTB-Markt werden stattdessen das Angebot an Werbefläche von Publishern durch Zwischenhändler über komplexe Technologieplattformen in Echtzeit versteigert, sobald ein Endkunde eine Website besucht und sich die Möglichkeit bietet, ihm Werbung anzuzeigen. In dieser Arbeit nehmen wir die Rolle eines Zwischenhändlers auf der Nachfrageseite des Marktes (DSP) ein, welcher für seine Kunden – Werbetreibende – in Echtzeitautkionen auf Inventar bietet. Um dies erfolgreich zu tun, muss sich ein DSP mit einer Vielzahl von anspruchsvollen mathematischen Herausforderungen beschäftigen, welche keine analytische Lösung haben. Wir untersuchen diese Probleme aus einer ganzheitlichen Perspektive und fokussieren uns insbesondere auf die Aufgaben zukünftige Marktpreise vorherzusagen, sowie optimale Bietstrategien zu finden. Ersteres Problem kann als klassisches Machine Learning bzw. Data Mining Problem verstanden werden, letzteres hingegen ist ein funktionales Entscheidungsproblem, für welches unter anderem Lösungsansätze aus der Spieltheorie, Optimierung, stochastischen Analysis sowie aus der Finanzmathematik vorgeschlagen wurden. Wir vergleichen jeweils mehrere verschiedene Lösungsansätze und untersuchen die Ergebnisse empirisch anhand eines großen öffentlichen Datensatzes.

Abstract

Real-Time Bidding (RTB) is a segment of the Digital Advertising Market that in recent years has grown in importance significantly. Due to the massive fragmentation of the market, it's intractable for most advertisers and publishers to negotiate with each other directly. In RTB, advertising technology (AdTech) intermediaries act on behalf of advertisers and publishers and match supply with demand in real time using auctions, whenever a consumer visits a publisher's website. These technology intermediaries are faced with complex mathematical and computational problems. In the present thesis, we consider the position of a demand side intermediary (DSP) that buys advertising space in auctions on behalf of advertisers. We take a holistic view of the challenges that DSPs face and consider various solution heuristics with a focus on the problems of predicting market prices of future inventory and choosing optimal bidding strategies in auctions. While the former is a classical Machine Learning/Data Mining problem, the latter is a functional decision problem that has attracted attention from various fields such as optimization, game theory, stochastic analysis and mathematical finance. We compare and contrast multiple solution concepts and test the results empirically using a large-scale public RTB dataset and give recommendations depending on the DSP's customer's campaign goals.

Contents

Chapter 1

Introduction

Real-Time Bidding (RTB) is a market mechanism in Computational Advertising where opportunities to display advertisements are sold via auction in real-time as soon as they are created by a consumer visiting a website or service. The consumer will in most cases not notice that this process is happening while he opens a web page and in order to give the user a frictionless experience, the entire process of finding a buyer for the ad inventory on the web page generally takes less than 100 milliseconds from the time the user starts loading the page. Due to the technological sophistication that is required for this process, publishers are generally not able to hold these auctions themselves, and neither are advertisers able to bid by themselves as these activities are not part of their core business expertise. Instead, intermediaries will actually hold the auction: Demand Side Intermediaries will hold auctions on behalf of publishers and aim to maximize their customers revenue, while Demand Side Intermediaries identify relevant auctions and bid in these on behalf of their clients, advertisers. This constellation gives rise to an economically complex and technologically demanding marketplace which will be the subject of this thesis. In particular, we will consider the position of a demand side platform (DSP) and deal with the mathematical and computational problems that it faces when bidding on behalf of its customers.

In Chapter 2 we will discuss the Programmatic Advertising market in more detail, introduce the major players and define Real Time Biddings place within the wider internet advertising ecosystem. Furthermore, we will describe the prevalent auction mechanism in detail.

Then, in Chapter 3 we will consider the position that is taken in this marketplace by a DSP and illuminate the range of mathematical and computational problems faced by such an intermediary. In essence, a DSP aims to find relevant supply of advertising slots (inventory) and acquires it on behalf of its customers. This ambition naturally raises questions such as *how* to determine the relevant inventory in a sea of tens of thousands of available auctions *every single second*, how to determine the value of available inventory to the advertiser and ultimately how much to bid for a certain impression that is offered at auction. The complexity of this market resembles that of financial markets and mathematical methods and models can play an equally important role in this setting as they are known to play in traditional finance. We will especially focus on two problems: Predicting market prices (Chapter 4)—both for individual impressions and for the expected behavior of the whole market in general—, and finding optimal Bidding Strategies (Chapter 5: How much should the DSP bid on a certain impression on behalf of a customer's campaign? This question is complicated by the fact that customers usually expect their campaign to fulfill various properties at the same time, such as maximizing utility while at the same time spending the budget at a constant rate specified by the customer. For both of these questions, we will test multiple solution concepts using the iPinYou Contest Dataset (see Section 4.3.1), a dataset of bidding locks released by Chinese DSP iPinYou in 2013, containing logs of tens of millions of auctions. In order to to so, we have implemented a simulation framework that brings together solutions to all the problems introduced in Chapter 3 and tests them on hypothetical campaigns.

Finally, in Chapter 6, we will summarize our results, give recommendations to DSPs on how to use our findings and motivate further research opportunities in this space.

Chapter 2

The RTB Market

With more and more people using the internet every day and the rise of mobile devices, the market for online advertisement has been growing rapidly in the past decade. In 2015, online advertising had a global market size of around €150 billion and is expected double by 2021 (Statista, 2016). Around 45% of this market is *display advertising*, while the rest is made up by *sponsored search / Search Engine Advertising (SEA)* and online classified ads. Display advertising covers all visual ads that are shown to consumers while using a website or service. This includes banner ads on websites or mobile ads, banner and embedded ads in social media services, as well as video ads that are often used on mobile devices. Especially due to mobile video advertising, the display advertising market it expected to continue growing at a 14% Compound Annual Growth Rate through 2021. More than half of display ads (by value) are sold *programmatically*, meaning the sale of inventory the entire distribution of ads happens automatically through software.

2.1 The Programmatic Display Advertising Market

Due to the technological complexity of programmatic buying, a rich ecosystem of players has evolved. A high level representation can be seen in Figure 1. We will introduce the major players and concepts one by one:

- *Consumers* visit websites or use internet services. The service might contain an *ad slot*, i.e. a possibility to display an ad to the user. When an ad is actually served to the consumer, this is referred to as an *impression*.
- *Publishers* are individuals or organizations that run web services that display ads to users. They would like to sell this inventory for maximum profit.
- *Advertisers* are companies or organizations that run an advertising campaign. They might be represented in this context by an advertising or marketing agency that implements the campaign for them.
- *Ad Exchanges* are places where inventory is traded, similar to a stock exchange in financial markets. For the purpose of this work, we will assume that all inventory traded is solt using a Real Time Bidding (Section 2.2).
- *Private Marketplaces (PMP)* are smaller market places where inventory is sold programmatically, either through RTB in private auctions, where publishers and bidders have to be preapproved, or as "programmatic guaranteed" packages that are bought for bundles of many impressions in advance. These private marketplaces are usually used to trade higher-value inventory than one finds at open ad exchanges.
- *Ad Networks* work with both the supply side (publishers) and demand side (advertisers). They match demand and supply of their customers and earn a profit through arbitrage. Alternatively, an ad network might also decide to sell off or bid on some inventory at an ad exchange.

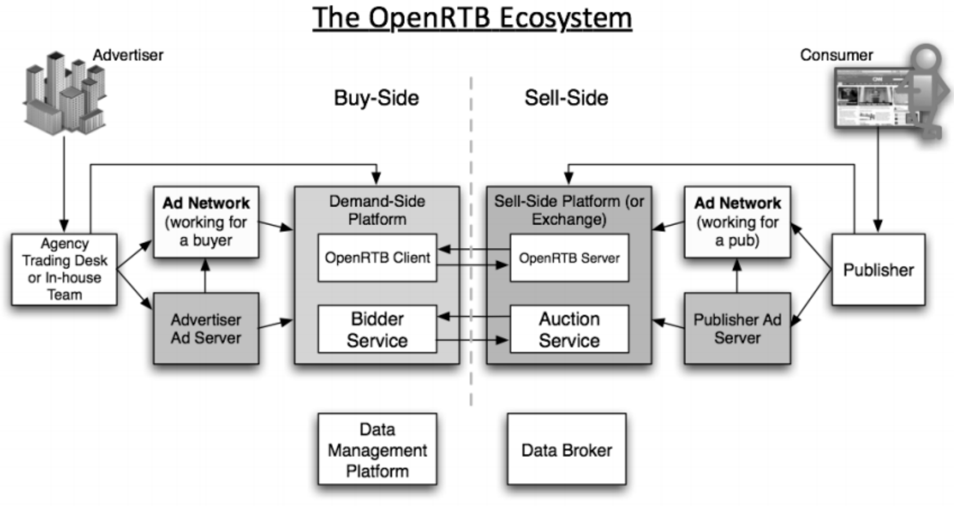

Figure 1: High-level communications in the Open RTB Ecosystem
(Source: (IAB, 2015, Chapter 1))

- *Supply Side Platforms (SSP)* are service providers that work for publishers in order to connect them to the remaining ecosystem and sell their inventory at maximum profit.
- *Demand Side Platforms (DSP)* are the demand side equivalent of SSPs. They bid in ad exchanges and run ad distribution on behalf of advertisers. Furthermore, DSPs provide the expertise to run sophisticated campaigns on behalf of their customers, including concepts such as retargeting (Section 3.6)
- Finally, *Data Management Platforms (DMP)* collect or acquire tracking data about consumers, aggregate this data and sell it to SSPs, DSPs, Exchanges or Ad Networks for a profit. Buyers of such data will try to use it to maximize the efficiency of their own operations.

The total number of programmatic sales can be divided into the groups of non-RTB sales that happen at Private Market Places and within Ad Networks, and Real-time Bidding sales. The latter will be the focus of this thesis and we will introduce a more detailed notion of RTB in the next section.
Generally speaking, one can observe a certain hierarchy in the display advertising market: The most valuable inventory, such as prominently featured banners at major newspaper outlets, are sold directly, using human agents. As inventory becomes more generic and less expensive, it will be sold using "programmatic guaranteed" contracts between publishers and advertisers (or their respective SSP/DSP), then private auctions at PMPs, and finally in Open RTB auctions on ad exchanges.

2.2 The RTB Auction Mechanism

For those impressions sold using real time bidding, a Supply Side Platform (SSP) or Ad Exchange will hold an auction on behalf of the publisher. Demand Side Platforms and ad networks will bid in these auctions on behalf of advertisers. A diagram of the players and their relationships is displayed in Figure 1. The Demand Side Platform that submitted the winning bid is then informed by the SSP that it won the bid. The DSP will then pay for the impression, allocate it to one of the campaigns that it's currently running and serve an advertisement to the publisher's ad slot. To deliver a smooth experience to the consumer, this entire process must take place while the website is loading. Commonly, an auction will only be open for about $100 - 200$ milliseconds, depending on the specific exchange. An illustration of the process is given in Figure 2.
In a *partially observable exchange*, none of the other bidders are informed about who won the auction or what the winning bid and payment price were. In a *fully observable exchange* these values (or at least the final payment price) are communicated to all participating bidders. The exact technical implementation

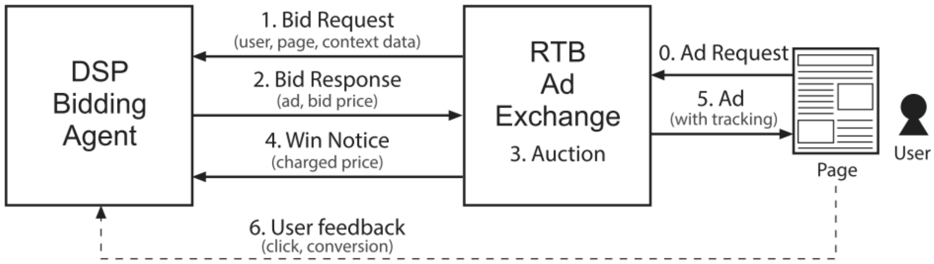

Figure 2: The process of auctioning a single impression.
(Source: Zhang et al. (2014b))

follows some publicly available protocol such as the OpenRTB protocol (IAB, 2015), but these conceptual notions of the auction mechanism are sufficient in the present context.

The auctions itself are usually modified *Second Price Sealed Bid (SPSB)* or *Vickrey auctions*. In such an auction, all bidders d submit a bid b_d to the auctioneer. The bids are only visible to the auctioneer and not to the other bidders. The auctioneer will then determine the winner as the bidder that submitted the highest bid, who will have to pay the amount of the *second highest bid* $b^{(2)}$. Vickrey auctions are a standard notion in the field of auction theory and have been analyzed in detail (Compare Krishna, 2009). In such auctions, it is a weakly dominant strategy for each bidder to bid her *private value* v_d, which is defined as the maximum price that the bidder would be willing to pay for the good.
A modification of Vickrey auctions that is usually employed in RTB is the presence of *reserve* or *floor prices*. These reserve prices are set by the ad exchange or directly by the publisher and aim to avoid impressions being sold for a value so low that the publisher deems the sale unfair or unprofitable. In this ecosystem, there are two types of reserve prices:

- Hard floor price ρ: A minimum sale price set by the ad-exchange. Any bid b that is submitted with $b < \rho$ is ignored by the auctioneer. If only one valid bid $v_d > \rho$ is submitted to the auction, then the good will be sold to bidder d at the price of ρ. If no valid bids are submitted, the good is simply not sold. In these cases, the supply side might choose to try to sell the impression in another subsequent auction (possibly with a lower reserve price), or simply to leave the ad slot blank or display advertisements for non-profit organizations that are sometimes alloted such unsold inventory.
  Conceptually, the hard floor price ρ can be interpreted as an additional bid submitted to the Vickrey auction by the auctioneer herself. In practice, hard floor prices are generally announced to potential bidders in advance.

- Soft floor price σ: A cutoff point at which the auction turns from a second price auction to a first price auction. If the highest bid $b^{(1)}$ is below the soft floor ($\rho \leq b^{(1)} \leq \sigma$), then the winner will be awarded the impression, but at the cost of $b^{(1)}$ rather than at the cost of $\max\left(b^{(2)}, \rho\right)$ In the case of $b^{(1)} > \sigma > b^{(2)} > \rho$, the payment price will be the soft floor.

In practice, hard floor prices are ubiquitous while soft floor prices are not employed by all ad exchanges. For SSPs, setting optimal floor prices is a mechanism design problem that is important in maximizing revenue to the SSP and the publisher. Here, however, we will take the point of view of a DSP, that faces the challenges outlined in Chapter 3. From the DSP's perspective, it is often useful to assume price-taking behavior, meaning that its own bidding behavior will have no effect on other advertiser's bids or the soft floor prices in the short run. Then any individual auction can simply considered a process that has some *winning price* or *price-to-beat* ω. When the bid submitted is higher than this winning price, the impression is won at the cost of the paying price π. Summing up the auction mechanism in this manner, an overview of possible winning and payment prices can be seen in Table 1.

	winning price ω	**payment price** π **(if** $b_d \geq \omega$**)**
$b_{-d} \geq \sigma \geq \rho$	b_{-d}	b_{-d}
$\sigma > b_{-d} \geq \rho$	b_{-d}	$\min(\sigma, b_d)$
$\sigma \geq \rho > b_{-d}$	ρ	$\min(\sigma, b_d)$
general	$max(b_{-d}, \rho)$	$\max\big(b_{-d}, \min(\sigma, b_d)\big)$

Table 1: Winning and Payment Price.
b_d denotes the bid submitted by DSP d, b_{-d} denotes the highest bid submitted by any other DSPs.

Chapter 3

Challenges Facing Demand Side Platforms

In this section, we will focus on Demand Site platforms and illuminate some of the problems and challenges that they are facing that are topics in academic research.
Ultimately, the goal of a demand side platform d is to deliver the best possible experience to their customers - advertisers - at the lowest possible price, which is achieved by matching advertisers demand for advertising space to the demand on the market.
In order to do this through Real Time Bidding, there are multiple subproblems that a DSP is faced with. In this section we will briefly describe these problems and ways that advertisers deal with them in practice.
For all subproblems in this section, we will assume that the DSP is facing a stream[1] I of bid requests i represented by their features $(x_i)_{i\in I}$ while running multiple campaigns for multiple customers. We denote the set of campaigns by K. For simplicity we will assume that each client of the DSP is running exactly one campaign $k \in K$. Further, each campaign k has predefined starting and ending times as well as a budget B_k and a set of goals that are specified by the clients. At any point in time we will refer to the *state* of a campaign by s_k, encompassing bid requests that have already been seen, their outcomes, the campaigns current remaining budget, as well as expectations about future requests and the market. We denote by $I_k \subset I$ the set of requests that are both potentially relevant to campaign k and also come in while campaign k is active.
Throughout this chapter and the rest of this thesis we will use the following notation:

- We denote by $W_k \subseteq I_k$ the set of impressions that have been won in the market and are awarded to campaign k and by $W := W_d := \cup_{k\in K} W_k$ the set of all impressions won by d.
  Analogously, we define the sets $Clicks_k$ and $Conv_k$ for the achieved clicks and conversions.

- The bid submitted to the exchange when bidding for impression i is denoted by $b_{i,d}$ or simply b_i.

- We denote by $W_k \subseteq I_k$ the set of impressions that have been won at the market and are awarded to campaign k , by π_i the price paid to the SSP or ad exchange for impression i, assuming all auctions are Vickrey auctions, π is thus given by

$$\pi_i = \begin{cases} \max(b_{i,-d}, \min(\sigma, b_i)), & i \in W \\ 0, & \text{else} \end{cases} \tag{3.1}$$

  as established in Section 2.2.

Furthermore, let us define the following notions for clarity. When considering an incoming ad request we will refer to the following *stages* that the DSP may be in:

[1] This “stream” can be formalized as a stochastic process, e.g. a Poisson process, but for simplicity, we will simply consider it to be a set of sequentially occurring events without any further structure

- In the *ex-ante* stage, no specific bid request has come in, but we might have some prior information about and expectations of future bid requests, such as the rate of bid requests and the distributions of their features, prices, and possibly values.
- In the *interim* stage, the DSP is presented with a specific bid request and can observe its features x_i as well as any second or third party data that the DSP has acquired from DSPs that pertain to the bid request. This corresponds to the time between events 1 and 2 in Figure 2.
- In the *ex-post* stage, all specifics of the bid request are known: We will know whether or not the impression was won, and if so, the paying price and any user feedback generated by the impression such as clicks or conversions.[2]

3.1 Bidding Strategies

When bidding on behalf of campaign k, the DSP's aim is to maximize the total utility to the customer while respecting the campaigns total budget constraint. Given some private valuation $v_{i,k}$ of an impression, the DSP must therefore find a *bidding strategy* $b_k(\cdot)$ that maps bid request x_i, the private valuation $v_{i,k}$ and the current state of the campaign s_k into the bid amount to be submitted to the exchange. It's of utmost importance that at any point in time, the DSP will have incomplete information about any future bid requests that will be contained in the stream, but can only observe past events, such as won or lost auctions, their prices and outcomes.
It is obvious that any reasonable strategy must rely heavily on the value of an incoming impression, the expected price that will have to be paid for it and the expectations about the further development of the campaign. Assuming that sufficiently accurate information about these metrics are available, the DSP is faced with the natural problem of deciding how much to bid for a certain impression.
We will call a function that maps an incoming bid request to the amount of money the DSP will bid for it a *bidding strategy.* Standard auction theory suggests that for a single second price auction, always bidding ones true value is an optimal strategy in the sense that it maximizes expected revenue to the bidder and constitutes a weakly dominant strategy (Krishna, 2009, Chapter 2.3). However, this notion does not necessarily generalize to the setting of many sequential auctions with a global budget constraint. Furthermore, in real life settings the private value of an impression is often not given intrinsically but needs to be inferred from the data available from the bid request. Commonly one would especially like to know the probability of an impression being clicked (Compare Section 3.2) and its expected market price (Sections 3.3, 4). For more sophisticated campaigns one would additionally like to predominantly reach browsers that have already been in contact with the advertisers website or service or with the campaign itself (*retargeting*), which will need to be reflected in the private valuation as well.
Finally, advertisers expect a certain spending behavior when running campaigns. For example, it is expected that the budget is spent relatively evenly over the predefined time-horizon of the campaign and that the entire allocated budget is actually spent on inventory. The problem of fulfilling these expectation is called *Budget Pacing.* Figure 3 gives an overview of factors that might be considered when choosing an optimal bidding strategy.
Assuming reasonably accurate solutions to all these subproblems are available - namely a way to determine the private valuation and expected price of an impression as well as information about the expected bid landscape over the time of the campaign - the problem of finding concrete bidding strategies will be analyzed in detail in Chapter 5.

3.2 Estimating Value: CTR and CVR

While the valuation $v_{i,k}$ of an impression might be fix and directly observable, more commonly it depends on unobserved variables such as the probability that the impression will be clicked, the *Click-through Rate (CTR):*

[2] In reality, the feedback cannot be observed instantaneously, as clicks or conversions might be attributed to a certain impression up to 10 days after the impression. Furthermore advertisers attribution models might give no or only partial attribution to a conversion that happened after the impression. For more information about attribution models, see Dalessandro et al. (2012) or Zhang et al. (2014c). For simplicity, we will assume that outcomes can be observed instantaneously.

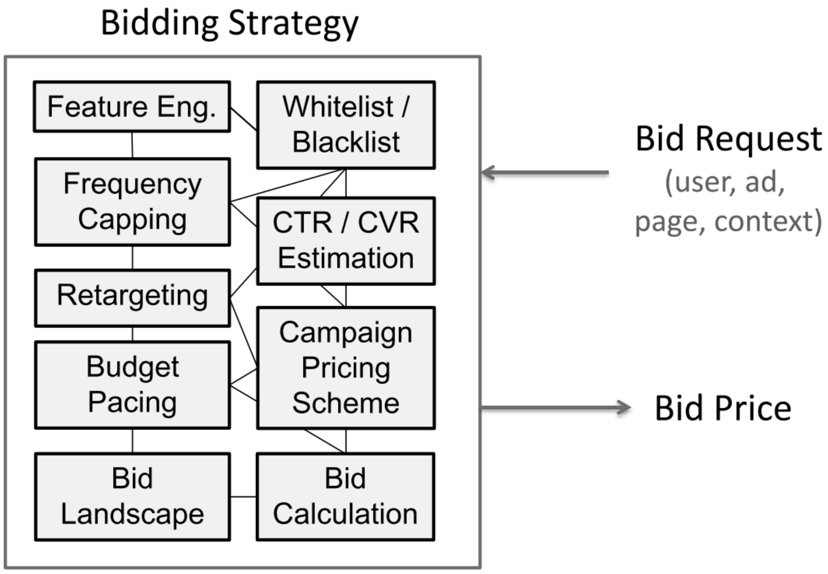

Figure 3: Overview of considerations when determining a bidding strategy.
(Source: (Wang et al., 2016, p. 64))

Let's consider an advertiser who is not primarily interested in many peopling seeing their ads, but instead in the number of people who click on them and are thus redirected to their website or service. In this setting, the true value of an impression to the campaign is given by

$$v_{i,k}^{true} = \begin{cases} v_{i,k}^{click}, & i \in Clicks_k \\ 0, & otherwise. \end{cases} \tag{3.2}$$

Since a click event can only be observed once an impression has been won, the true (ex-post) value of the impression using the previous definition is not known to the DSP when determining the bid price. Instead, the private valuation will have to be set by the expected true value using an interim estimator:

$$v_{i,k} := \mathbb{E}\left[v_{i,k}^{true}\right] = v_{i,k}^{click} \cdot P(i \in Clicks_k | i \in W_k) =: v_{i,k}^{click} \cdot eCTR_{i,k} \tag{3.3}$$

Finding the *expected click-through rate* eCTR can be seen as a supervised machine learning task, where the model can be trained using historical data from the current or past campaigns.
CTR estimation is the problem in RTB that has achieved the most academic attention and a wide range of solution concepts have been proposed, including factorization machines (Shan et al., 2016), deep neural networks (Du et al., 2016) as well as many others. A recient review of the CTR prediction literature can be found in Stratakos (2016).
Finally, it should be noted that most results in click through rate prediction can be generalized to CVR (conversion rate) prediction, i.e. estimating the probability that a user converts (E.g. buys a product or signs up for a newsletter) given the fact that she clicked an ad.

3.3 Estimating Cost: Winning Prices and Bid Landscape

Estimating the cost structure of a campaign can be separated into two main subproblems: Winning price prediction on the impression level on the one hand, and predicting a campaign wide price distribution or *Bid Landscape* on the other hand.

3.3.1 Winning Price Prediction

At the impression level, the expected winning price, i.e. the price to beat in order to win the impression, is an important input factor when optimizing a specific bid. On the one hand, it directly informs about

the bid level which is required in order to be competitive and have a realistic chance of winning the impression, on the other hand, in second price auctions the winning price is usually identical to the paying price (unless it is below the soft floor). As such, having an estimate for the winning price allows the calculation of KPIs such as expected Return on Investment of a specific impression.
When training data from previous (or the current) campaigns is available, we can use the features x_i of a bid-request to predict the winning price using machine learning. This leads to a regression-type supervised learning problem. We will discuss problem-specific details and solution concepts in Section 4.1

3.3.2 Bid Landscape Forecasting

While for setting a campaign-wide bid strategy the expected price of impressions is extremely important, predictions in the sense of the previous section are only possible once a bid request has come in and its features are known. Therefore, a different approach is necessary for modeling the future price distribution. In general, one is interested in some basic properties of the stream of incoming bid requests, such as the rate of the process, i.e. the frequency of bid requests, the distribution of winning prices, and—ideally—the distribution of private values and the relationship of these distributions. When these information are available it allows us to calculate the expected future spending when using a certain bid strategy, a tool that is invaluable in campaign planning.
A detailed discussion of the bid-landscaping problem can be found in Section 4.2.

3.4 Allocation of Impressions to Multiple Campaigns

When faced with a bid request x_i it might often be the case that the impression is relevant to multiple of the DSP's customers, i.e. $\exists K_i^* \subseteq K$ with $\#K_i^* \geq 2$ s.t. $x_i \in \cap_{k \in K_i^*} I_k$. It is common practice for DSPs to hold a *local auction* between the interested campaigns and award the impression to the winner of that local auction. While for a single impression this is efficient in the sense that it maximizes total utility, it can lead to severe problems when bidding repeatedly.

Example. Let k_1, k_2 be two campaigns with $I_1 = I_2 = I$ with constant private impressions values $v_{i,1} = 1.00, v_{i,2} = 0.99$ for all impressions $i \in I$. Then k_1 will win every local auction and be awarded every won impression, as long as it has any remaining budget. k_2, on the other hand, will not receive any impressions at all until all of k_1's budget has been spent.

Intuitively, while optimal in a utilitarian sense on single-impression level, this outcome seems unfair on a campaign level. More importantly, the advertiser k_2 might decide to run future campaigns on a competing DSP due to the disappointing results. As such, more involved rules for impression allocation might be desirable.

Stavrogiannis (2014, Chapter 6) takes a game theoretic approach to analyze competition between a duopoly of DSPs depending on their mechanisms for awarding impressions to customers in a setting with non-captive customers and finds that the DSP's "optimal" mechanism (with regard to maintaining customers) is dependent on the strategy employed by its competitor. However, the methods considered are variations of private auctions that all award the impression to the highest bidder.
Charles et al. (2013) describe a mechanism where certain campaigns are selectively "throttled out", meaning hindered from participating in some auctions depending on their current campaign status. The aim of this method is both to ensure evenly paced spending in each campaign (compare Budget Pacing in Section 3.1) and to maximize *Social Welfare* over the campaign set K rather than considering each campaign separately (as is the case in most of the bidding strategy literature. Although they use a Benthamite social welfare function in their framework (i.e. the one given by $sw_K = \sum_{k \in K} u_k$), they also note that the welfare (i.e. utility) of each individual customer must be kept sufficiently high as to avoid disappointment. This suggests a possible extension of their framework where the DSP might model their bidding strategies such as to maximize advertiser social welfare where the social welfare function is chosen to be nonutilitarian and thus incorporates some notion of fairness. This, however, is beyond the scope of this thesis.

3.5 Fraud Detection

Fraud is a common occurrence in RTB based advertising due to the sheer number of transactions and parties involved as well as the anonymity in the market and on the internet in general. A common scheme is that of *fraudulent impressions* where a publisher maliciously inflates the number of impressions on their website or service.
In this scheme, the publisher embeds ad slots on their website which are being sold through an ad exchange using the RTB protocol. However, the publisher or an accomplice will then generate illegitimate traffic to the website that serves the simple purpose of generating impressions (and possibly clicks) and thus ad revenue to the publisher. While the majority of this fraudulent traffic is generated through automated software posing as real users—often using networks of malware-infected computers or "botnets", there is also fraudulent traffic caused by actual humans. The latter includes a phenomenon called "Click Farms" where workers in low-wage labor markets such as Bangladesh spend their entire working day visiting websites and clicking on ads or giving the appearance of popularity to a mobile app, online service or social media profile (compare Arthur, 2013). In fact, this scheme is so prevalent that there are dozens of companies offering "Traffic Generation" as a professional service, causing advertisers hundreds of millions of US dollars in lost revenue each year (Springborn and Barford, 2013).
Naturally, as a DSP, one would like to detect this fraudulent traffic and blacklist the corresponding publishers or at the very least avoid bidding on such inventory. Stone-Gross et al. (2011) give a more detailed overview of the methods used by fraudsters as well as methods employed by ad networks and DSPs in order to detect them. These include, amongst others,

- *Anomaly Detection*: Publishers with unrealistically high or low CTRs or a sudden increase in requests from a certain domain are flagged.
- *Bluff Ads*: If a publisher is deemed suspicious, the DSP might replace the creative on a random subsets of impressions by a dummy, unappealing ad, such as a completely white or black picture. One would expect that no legitimate human user would have any reason to click on this ad, and thus one would expect the click through rate to fall of steeply for these bluff ads. In this way the DSP might employ A/B testing: If the click probability on the bluff ad remains the same, it is a clear indication for fraud.
- *Popularity and Page Rankings*: There are publicly available and trusted services that rank websites and services based on their popularity. These rankings can be used to infer the expected amount of ad traffic that a certain publisher is expected to generate. If the actual amount of impressions vastly exceeds these expectations, the publisher might be considered suspicious.

Stitelman et al. (2013) note that different fraudulent websites are commonly visited by the same users. This is the expected behavior if, for example, multiple websites employ the services of the same *Traffic Generation* provider. Conversely, the same set of users is unlikely to visit legitimate websites. Thus, by building a Co-Visitation graph of publishers by the browsers visiting them, dubious websites can be identified if they form an isolated subgraph.

3.6 Retargeting

Dealing with consumers that have previously been encountered by the campaign is relevant for two reasons: On the one hand, it is often desirable to reach the same consumer multiple times. A user who clicked on an ad and visited an advertiser's website or service but ultimately didn't drive a conversion could be shown a specific ad containing, for example, a discount code for the previously abrupted sale. Modern, sophisticated campaigns aim to reach consumers multiple times with differing creatives and have the ads "tell a story" to the user. For these reasons, identifying individual customers has become very important.
On the other hand, one wants to avoid showing a user the same ad too often, as this will lead to the ad going unnoticed or in the worst case even annoying the consumer. Solution concepts to the latter problem are *Frequency Capping* and *Recency Capping* (Compare, for example, Yuan et al., 2013).
Retargeting is beyond the scope of this work and we will not evaluate its effects on the other problems in the remaining work.

Chapter 4

Winning Price Prediction and Bid-Landscape Modeling

In this chapter, we will consider the problems outlined in Section 3.3, namely finding a winning price estimator for a given impression, as well as describing the bid landscape as a whole.

4.1 Prediction of Winning Prices

4.1.1 Problem Statement

Assume we are given a training set T ($\#T =: N$)of labeled points (ω_i, x_i), where for an impression $i \in T$, $\omega_i \in \mathbb{R}_{>0}$ is its *winning price* and $x_i \in \mathbb{R}^P$ is its feature vector. Using this data, we would like to be able to predict ω_j for any unlabeled point x_j that we encounter, whether it is contained in the training set or not. Finding such an estimator $\hat{\omega}(x)$ can then be seen as a typical instance of a supervised learning regression problem. We will start by stating the problem specifics.

We will consider the feature vectors $x_i \in T$ to be i.i.d. realizations of the random variable $X \sim p_X$ that is defined over a probability space $\mathcal{X} \subset \mathbb{R}^P$. We further assume that the actual winning prices ω_i are realizations of the random variable Ω that is given by

$$\Omega = f(X) + \varepsilon \tag{4.1}$$

where f is a function $f : \mathcal{X} \to \mathbb{R}_{>0}$ and ε is a zero-mean random variable. As such, we assume that an impressions winning price ω_i consists of some deterministic relationship (f) to its features and the noise ε which models the nondeterministic effects in the underlying process. A predictor $\hat{\omega}$ should then aim to approximate the unobserved function f as well as possible:

$$\hat{\omega}_i := \hat{\omega}(x_i) \approx f(x_i). \tag{4.2}$$

A predictor model $\hat{\omega}$ can then be evaluated by how well it fits the actual observed labels in the training set. In this regression setting, an appropriate evaluation metric is given by the *root mean squared (training) error*:

$$RMSE_T := \sqrt{\frac{1}{N}\sum_{i \in T}(\omega_i - \hat{\omega}_i)^2} \tag{4.3}$$

An alternative estimator is the *mean absolute error*

$$MAE_T := \frac{1}{N}\sum_{i \in T}|\omega_i - \hat{\omega}_i| \tag{4.4}$$

While the MAE has the nice interpretation of "how far" the average prediction will be from the truth, the $RMSE$ has the desirable property that predictions that are significantly off target will be penalized harder. In our analysis, we will use the $RMSE$ as the main performance metric.

4.1.2 Regression Algorithms

For solving this supervised learning problem, we consider two of the most prominent classes of regression algorithms: Generalized Linear Models (GLMs) and Tree Ensemble models such as Random Forests or Gradient Boosted Regression Trees (GBRT) (For an introduction to either of these models, see Bichler (2016), or compare Hastie et al. (2009) for a rigorous treatment.) In the RTB literature, linear models have been proposed for Winning Price Prediction by Li and Guan (2014) and Wu et al. (2015), while a variation of GBRT has been applied to this problem by Cui et al. (2011).

General Linear Models

GLMs start with the assumption that ε has a specific distribution and that f has a linear structure[3]

$$f(x_i) = \beta_0 + \sum_{j=1}^{P} \beta_j(x_{i,j}) = \beta^T x \tag{4.5}$$

where in the second equality we use some slight abuse of notation by identifying $x_i = (1, x_i) \in \mathbb{R}^(P+1)$ with $x_{i,0} := 1$ for all i. "Training" then refers to fitting optimal parameters $(\beta_0, \ldots, \beta_P) \in \mathbb{R}^{P+1}$ in the sense that it minimizes the (squared) residual errors or *Residual Sum of Squares (RSS)*:

$$\min_{\beta} RSS_T = \sum_{i \in T} (\hat{\omega}_i - \omega_i)^2 . \tag{4.6}$$

With this objective and the assumption that ε is normal distributed, the model is called Ordinary Least Squares (OLS) Regression. However, in order to avoid overfitting (See Figure 4), one usually uses *regularization*, and adds penalty terms to the objective function:[4]

$$\min_{\beta} RSS_T + \lambda_1 ||\beta||_1 + \lambda_2 ||\beta||_2. \tag{4.7}$$

with parameters $\lambda_1, \lambda_2 \geq 0$. A higher value of λ_2 will lead to smaller coefficients β_i; a higher value of λ_1 will induce sparsity into the solution, i.e. reduce the number of nonzero coefficients. Thus the presence of either one of these parameters reduces the overall model complexity. With $\lambda_1 = 0, \lambda_2 > 0$ the model is called *Thikonov regression* or *ridge regression*, with $\lambda_1 > 0, \lambda_2 = 0$ it's called the *LASSO (least absolute shrinkage and selection operator)*, when both types of regularization are present simultaneously, it's referred to as the *elastic net*.
The regularization parameters can be chosen by using a *holdout (or validation) set*: Here, the original training set is randomly split into two sets, a new training set and a holdout set that is usually significantly smaller. A model that overfits the training data will have a small error metric on the training dataset but this high accuracy will not generalize to unseen data. This can be mitigated by using the predictor on the validation set and choosing the regularization parameters optimally such that when trained on the training set, the error on the validation set is minimized. This relationship is visualized in Figure 4. A more sophisticated validation method is *$k-$fold Cross Validation*, where the training data is split into k subsets, and the model trained on any combination of $k-1$ of these sets and evaluated on the remaining one. When $k = N$, this is called *Leave-One-Out Cross Validation*. This method is especially useful when N is small and using some of the data exclusively for validation come at a cost as the training set size is reduced which might impact model fitting. In our case, however, we have the luxury of a very large N and can therefore afford to set aside a validation set.

Tree Based Methods

Tree methods aim to find a partition R of the feature space $\mathcal{X} := \dot{\bigcup}_{m=1}^{M} R_m$. In a *regression tree*, this partition is found by sequentially splitting the feature space into two, depending on the values of a single feature. The order of splits is usually found in a way that the next split will minimize the variance of the

[3] This specification refers to standard linear regression. In its more general form, f actually takes the form of $\Phi(x) := \big(\Phi_1(x), \ldots \Phi_Q(x)\big)$, a Q-dimensional vector of *feature transformations* achieved from the original features in some way. (Compare Section **??**). However, for simplicity of notation we will assume that x is already represented in its final form.

[4] Abuse of notation: β_0 is usually exempt from the penalty term.

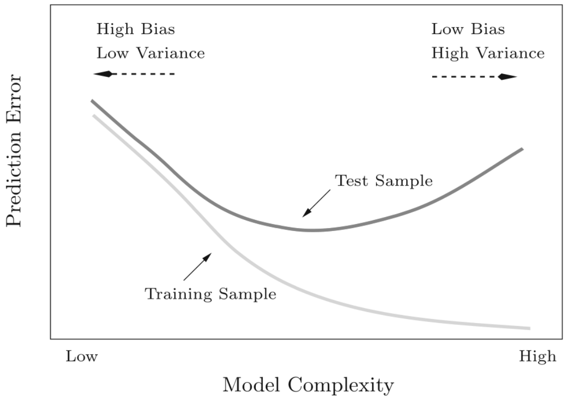

Figure 4: **Bias Variance Tradeoff**. A model with high complexity will fit the training set better, but will not generalize to unseen data. ("Overfitting") Lower complexity models, however, will be biased on the training data ("Underfitting"). An optimal model should make a tradeoff between bias and variance such that the results on the training set generalize well to unseen data.

(Source: (Hastie et al., 2009, p. 38))

response variable within the resulting subsets. A simple regression tree can be seen in Figure 5. Each leaf node m of the tree is then assigned the mean of the responses of labels that fall into this node:

$$c_m := \frac{1}{\# R_m} \sum_{i | x_i \in R_m} \omega_i \tag{4.8}$$

and f is estimated by checking which node an impression belongs to and using the corresponding c_m:

$$\hat{\omega}_i = \sum_m \mathbb{1}(x_i \in R_m) c_m \tag{4.9}$$

As in GLMs, there is a variance-bias tradeoff in tree-based models: The complexity is dependent on the depth of the tree, i.e. the number M of partitions. Overfitting can thus be avoided by limiting M and *pruning* the tree dynamically (Hastie et al., 2009, Chapter 9.2).

Tree Ensemble Methods

Finding the optimal tree-complexity parameters can be difficult in practice. Furthermore, finding the optimal splitting-structure is computationally expensive and thusly, trees are usually built in a Greedy fashion. It turns out that these Greedy solutions are often highly variant in their structure, i.e. a small change in the training set will have significant effects on the structure of the tree. To avoid these problems, tree ensemble methods base the final predictor on training multiple trees at once.
Random Forests uses *Bagging* to train many trees: One selects k different subsets $T_k \subsetneq T$ of the training data, then trains a tree on each of these k subsets and averages the results. When pruning each tree, the tree's performance is evaluated on those instances *not* contained in T_k and thus random forests have a variant of cross validation built-in. The final predictor is then given by

$$\omega_i := \frac{1}{k} \sum_k \sum_m \mathbb{1}(x_i \in R_{k,m}) c_{k,m}. \tag{4.10}$$

While random forests can be thought off as building many trees in parallel, *Gradient Boosted Regression Tree* models build trees sequentially. The first tree produces an estimator $\hat{\omega}_1$. The second tree then produces another estimator ω_2 that aims to estimate the squared *residual errors* $(\omega - \hat{\omega}_1)^2$ of the first tree. The third tree then aims to estimate the residuals of the second tree $((\omega - \hat{\omega}_1) - \hat{\omega}_2)^2$ and so on, until finally, after K trees, the total estimator is given by

$$\hat{\omega}_i(x_i) = \sum_{k=1}^{K} \hat{\omega}_{i,k} \tag{4.11}$$

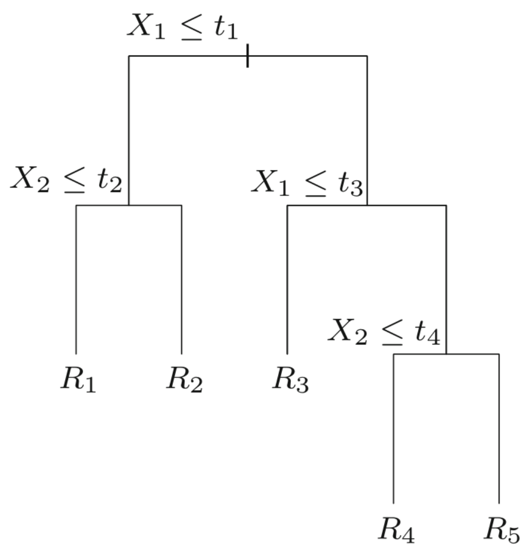

Figure 5: A simple regression tree with its induced partition of $\mathcal{X}$.
(Source: (Hastie et al., 2009, p. 306))

An advantage of gradient boosted tree models is that they can learn nonlinear relationships between features and the response variable as well as take into account interactions of features without explicitly including them in the feature specification x_i. This is desirable because the problem of winning price prediction is high dimensional due to the large number of categorical variables, and the dimensionality grows polynomially when additionally considering interaction terms (compare Section **??**).

Censored Regression

While using the problem formulation in Section 4.1.1 seems straightforward, there is one conceptual problem with it: When bidding on most ad exchanges the data that we observe and that is usable in the training set is *censored:*

- For impressions that were lost, we usually cannot observe the true winning price. Instead, we only know that our bid b_i was insufficient and thus observe $b_i < \omega_i$. For such impressions, the data is *right censored.*
- Similarly, when we won the impression our bid was below the soft floor price σ_i, then we will pay b_i as in a first price auction. As we can only directly observe the paying price $\pi_i = b_i$, the only thing that we can infer about the winning price is $\omega_i \leq b_i$, i.e. our data is *left censored* in this case.

In the classical machine learning sense of the problem, the training set would contain only those impressions that were fully observable.

Wu et al. (2015) consider a modified linear model that takes into account the information contained in the censored data as well. Here, we will explain their approach as it pertains to the right censored data, but it is equally applicable for left censored data as well. For standard OLS regression, the objective function was given by (4.6). It is a well known fact, that a coefficient vector β found by using this least squares method will at the same time be the *maximum likelihood estimate (MLE)* of the linear model

$$\omega = \beta^T x + \varepsilon. \tag{4.12}$$

Recall that a model assumption was that $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ with some standard deviation σ (that can be estimated from the residuals). This means that β could also be found by minimizing the negative log-likelihood, which for this model is given by

$$- \ell\ell_{OLS}(\beta) = \sum_i - \log\left(\phi\left(\frac{\omega_i - \beta^T x_i}{\sigma}\right)\right) \tag{4.13}$$

where ϕ is the probability density function (pdf) of the standard normal distribution.
Wu et al. now observe, that under the same model assumptions (4.12), we can also determine the likelihood of the censored data: We have

$$P(\omega_i < b_i) = P(\epsilon_i < b_i - \beta^T x_i) \tag{4.14}$$

$$= \Phi\left(\frac{b_i - \beta^T x_i}{\sigma}\right) \tag{4.15}$$

with Φ being the cumulative distribution function of $\mathcal{N}(0,1)$. The probability in (4.14) is called *winning function* and will be our subject of interest in Section 4.2. Under the linear model, the distribution of this winning function should then adhere to (4.15). Goodness of fit for a given β can then be observed on the data of lost won and lost bid requests, and a β that is a Maximum Likelihood Estimate for *censored linear model*, i.e. the combination of fitting both the uncensored and censored data can be found according to the Tobit-regression model (Tobin, 1958) by

$$\beta_{clm} = \arg\min_\beta \sum_{i \in W} - \log\left(\phi\left(\frac{\omega_i - \beta^T x_i}{\sigma}\right)\right) + \sum_{i \in L} - \log\left(\Phi\left(\frac{b_i - \beta^T x_i}{\sigma}\right)\right) \tag{4.16}$$

Wu et al. further note that the form of the winning function that is implied by (4.15) is inconsistent with the one observed in reality (compare Section 4.2) and show empirically that the CLM model itself does not constitute an improvement of $\hat{\omega}$ over simply using the OLS model. To mitigate this fact, they propose a mixture model of OLS and CLM based on empirical estimates of the winning function. We will omit the exact definition of that model here, but the authors show empirically that using this mixture model actually improves on simple OLS regression.

4.1.3 Feature Preparation and Computational Aspects

While the algorithms presented in the previous section require interval scaled inputs $x \in \mathbb{R}^P$, real world datasets are often not of this form. Instead, bid requests will contain categorical fields such as the AdExchange of an incoming impression which are represented as a string. The typical way of dealing with such data is *One Hot Encoding (OHE)* (Compare Bichler, 2016). However, this method has some properties that can lead to undesired behavior:
If the possible values that a feature can take are unknown before training, OHE requires that the feature be indexed, i.e. the possible categories are inferred dynamically from the data. For problem settings with very large sample sizes—as we have in RTB—this can become computationally expensive very quickly. Moreover, this representation is problematic when the test set might contain previously unseen values: As OHE typically represents a feature with k possible variables by $k-1$ dummy columns (to avoid linear dependency of columns), an unseen value that wasn't in the training set cannot be represented in this fashion. A third caveat is that expanding categorical columnns in this fashion quickly adds to the dimensionality of the problem, especially when considering interaction terms.

A method that deals with all of these problems is *Feature Hashing*, also called *the Hashing Trick*(Weinberger et al., 2009). In this method, one sets a number m of desired columns manually. For each categorical feature in the training set, each of its values is then hashed into one of m buckets using a hashing function ζ and is assigned a sign $\xi \in \{-1, 1\}$ using a different hashing function. For $1 \leq j \leq m$, and an impression i, the feature $x_{i,j}$ is then defined by

$$x_{i,j} = \sum_{c \in \text{Original features}} \xi(x_{i,c}) \cdot \mathbb{1}(\zeta(x_{i,c}) = j). \tag{4.17}$$

Weinberger et al. show analytically that this structure is able to retain most of the information of a sparse high-dimensional space within a lower dimensional representation. While some information might be lost in this way, it has great computational advantages in that it reduces dimensionality, avoids indexing and can deal with unseen data. Furthermore, feature hashing has been successfully applied to many real world datasets and as such has become a standard method for RTB-related machine learning tasks (Wu et al., 2015; Zhang et al., 2014a; McMahan et al., 2013).

4.2 Bid Landscape Forecasting

4.2.1 Problem Statement

The goal of Bid Landscape Forecasting is to generate an ex-ante estimate of the supply side of the market for the time frame of the campaign. Here, we assume the DSP to exhibit price taking behavior, i.e. the DSPs own behavior will not influence future market prices in the short run. One would like to know the *rate* of incoming bid requests per time, the distribution of market prices p_ω, often represented by its cumulative distribution function which we will call the *winning function* of the market:

$$w(b) := P(\omega < b) \tag{4.18}$$

This winning function will allow us to estimate how many impressions we will win when bidding according to a certain strategy. This is of utmost importance when trying to achieve budget pacing as desired by the advertiser.

It should not go unnoticed that this representation implicitly assumes a very strong assumption: For the ex-ante stage winning function to be sufficiently informative, one must assume that the winning probability depends only on the bid amount and is independent (or only weakly dependent) on the actual realization of the impression. This seems counterintuitive at first: We would assume that impressions have an inherent value determined by the consumer creating it, and that all DSPs alike would be able to identify this inherent value represented in x_i and bid much higher accordingly and the winning function should thus be given by $w(b, x)$. However, in real world RTB settings, it is often the case that the private values of an impression will differ significantly between the DSPs bidding on it, due to different campaign goals and highly specific targeting rules. Often, auctions will have very low demand density, with only a handful of DSPs bidding in it. (Zhang et al., 2014a, Section 4.1) investigate this phenomenon empirically using the iPinYou data set (See Section 4.3.1) and observe that only a small minority of variance in the winning price can be explained by *between-feature variance* and conclude that making the assumption $w = w(b)$ is, in fact, warranted. They further note that making a corresponding assumption is a well-established practice in the related field of search engine advertising (Kitts and Leblanc, 2004; Zhang et al., 2012). One caveat that was not considered in these deliberations, however, is the possible presence of very similar campaigns: In a highly commoditized industry it might be a very reasonable phenomenon that multiple advertisers are running very similar campaigns and are targeting the same types of impressions. In such a setting, the bids of these campaigns would be highly correlated which would lead to a direct dependence of w on x if the campaigns are bidding at a competitive level. Such campaigns do not seem be present in the empirical data considered here and elsewhere. Therefore we consider the assumption to be justified in the present context.

4.2.2 Finding the Winning Function

Usually, an estimator for the winning function is found in one of two ways: Either by using the empirical distribution of ω in the training data (or current campaign); or by using a prior on the structure of w and then fitting optimal parameters according to the training data.

Empirical Methods

Using the empirical distribution of winning prices seems attractive at first glance, since this method is model-free and can thus replicate the observed distribution completely. Furthermore, in RTB settings, the training set can be assumed to be sufficiently large that the empirical distribution will approximate the real underlying distribution well.

However, the method can become computationally expensive: As function-spaces are infinite dimensional, we can only store the distribution up to some arbitrary level of precision. To accurately represent the empirical distribution, we would have to store all winning prices from the bid history, which is prohibitive both for storage and computation time reasons, as evaluation of the winning function has to be performed under low latency time constraints when choosing the bid amount.
Alternatively, the winning function can be approximated using a sequence of line segments, i.e. by storing function values for some discrete set of sample bid amounts, or by storing the quantiles of the distribution up to a desired accuracy.

In the absence of training data, one might want to learn the bid landscape dynamically from the current campaign. In this case, however, there might be a conflict of interest between bidding for those impressions that will contribute most to the campaign goal or those that will be most informative for getting a better estimate of the bid-landscape. Ghosh et al. (2009) propose two algorithms for the cases of fully-observable exchanges ("Learn-then-Bid", which essentially collects training data before bidding on the most valuable impressions only) and partially-observable exchanges ("Guess-Double-Panic"). The latter algorithm will change strategies adaptively based on the current state of information that the campaign possesses. The authors further show convergence of the landscapes learned by both these methods to the true underlying distribution.

Model Based Methods

The other approach for finding the bidding function is by enforcing some structure that is known from prior information. Once this structure is known, bid landscaping can easily be completed by fitting some parameters of the distribution. We will therefore content ourselves with reviewing some forms of suitable distributions that have been suggested for this task.

Log-normal Distribution

A form that is often used in the literature (e.g. Cui et al., 2011; Ghosh et al., 2009) is assuming the winning prices to be log-normally distributed:

$$\omega \sim p_\omega \approx \text{Log-Normal}(m, s) \tag{4.19}$$

with location parameter m and shape parameter s. This function can then easily be fitted to the training data by calculating the sample mean μ and sample standard deviation σ of ω and setting

$$m = \log\left(\frac{\mu}{\sqrt{1 + \sigma^2/\mu^2}}\right) \tag{4.20}$$

$$s = \sqrt{\log\left(1 + \frac{\sigma^2}{\mu^2}\right)}. \tag{4.21}$$

The winning-function is then given by

$$w(b) = F_\omega(b) = \Phi\left(\frac{\log(b) - m}{s}\right) \tag{4.22}$$

Both Ghosh et al. and Cui et al. note that a log-normal distribution fits winning price data well in real-life datasets.

Other distributions

Zhang et al. (2014a) use two forms of strictly concave winning functions of the form

$$w(b) := \frac{b}{b+c} \qquad \text{or} \qquad w(b) := \frac{b^2}{c^2 + b^2}, \tag{4.23}$$

Other authors use a Gamma distribution (Chapelle, 2015) or Gaussian distribution of $\sqrt{\omega}$ (Mirrokni et al., 2010).

A Hybrid Bid-Prediction and Bid Landscape Model

Cui et al. (2011) propose a model that combines aspects of Winning Price Prediction and Bid Landscape modeling that avoids the strong assumption that w be independent of x. In training, they split the feature space into m *templates* using a Decision tree method, such that the templates capture the variance of ω that is due to the variance in x. Within each template, the bid landscape is then assumed to be log-normally distributed with parameters learned from the training data.
In the campaign itself, the distribution of bid requests into the templates is found empirically, and the campaign-wide bidding function is finally found using an appropriately weighted mixture model of the template-specific winning functions.

4.3 Empirical Results

4.3.1 Experiment Setup

The iPinYou Contest Dataset

The iPinYou Contest Dataset was released in 2014 by iPinYou, one of of the largest DSPs in the People's Republic of China and is described in detail in (Liao et al., 2014). It was released as part of a competition in which participants had to design bidding strategies that would aim to maximize

$$\#\text{Clicks} + N\#\text{Conversions} \tag{4.24}$$

for a given campaign. The dataset contains of a training and a test set for three "Seasons" of the competition: In each season, participants could use the season's training and test data to design their strategies which were then evaluated in iPinYou's production system using real-world campaigns to determine the winner.
Due to very limited availability of public RTB data, the iPinYou dataset is —to our knowledge—the only publicly available large-scale data set that contains enough information to experiment on all of CTR-Prediction, Winning Price Prediction, Bid Landscaping and testing bidding strategies. As such it has become a standard benchmark dataset for academic research into RTB and has been used for empirical results in Zhang et al. (2014b); Wu et al. (2015); Zhang et al. (2014a); Du et al. (2016) and others. Zhang et al. (2014b) gives an overview of important summary statistics of the data set as they pertain to RTB related questions.

In this work, we use the data of Season 2 of the iPinYou data set, which contains bidding logs collected on 5 campaigns during 10 days in June of 2013 (Jun 6–12: Training, Jun 13–15: Test). We chose to focus on this season, because while both Seasons 2 and 3 contain richer data than Season 1 (some targeting information on users), the Season 3 data is much smaller than that of Season 2. In its original form, the Season 2 training set consists of 20.1 GB of data containing logs of 53'289'330 bids, 12'190'344 won impressions, 8'729 clicks and 391 conversions. (Test set: 1.27 GB, 2.5) These logs can be combined to get the resulting events for each bid request in the bidding logs.[5]

For each bid request, the dataset contains the features depicted in Table 2.

Results: Winning Price Prediction

We treated the winning price prediction as a supervised learning problem as described in Section 4.1.1. We observed that the total amount of data in the training set was more than sufficient and that using all of it did not bring significant improvements over smaller subsamples, regressors trained on 50'000, 100'000 up to 450'000 observations did not show any significant differences in test set accuracy. Therefore

[5]A version of the dataset that was preprocessed in such a way was provided to us by the author of Stratakos (2016).

Col #	Description	Example
*1	Bid ID	015300008...3f5a4f5121
2	Timestamp	20130218001203638
†3	Log type	1
*4	iPinYou ID	35605620124122340227135
5	User-Agent	Mozilla/5.0 (compatible; \ MSIE 9.0; Windows NT \ 6.1; WOW64; Trident/5.0)
*6	IP	118.81.189.*
7	Region	15
8	City	16
*9	Ad exchange	2
*10	Domain	e80f4ec7...c01cd1a049
*11	URL	hz55b00000...3d6f275121
12	Anonymous URL ID	Null
13	Ad slot ID	2147689_8764813
14	Ad slot width	300
15	Ad slot height	250
16	Ad slot visibility	SecondView
17	Ad slot format	Fixed
*18	Ad slot floor price	0
19	Creative ID	e39e178ffd...1ee56bcd
*20	Bidding price	753
*†21	Paying price	15
*†22	Key page URL	a8be178ffd...1ee56bcd
*23	Advertiser ID	2345
*24	User Tags	123,5678,3456

Table 2: Features of bid requests in the iPinYou dataset
(Source: Zhang et al. (2014b))

we used a random subsample of 500'000 impressions for training and validation.

We conducted only minor feature engineering and trained our models using Elastic Net Regularized Linear Regression and Gradient Boosted Regression Trees. Feature Preparation and model training have been implemented using the Apache Spark's ML machine learning library (Zaharia et al., 2010), which trains these machine learning algorithms using distributed stochastic optimization methods, such as Stochastic Gradient Descent or more sophisticated first-order (Kingma and Ba, 2014) or second-order quasi-Newton methods (Byrd et al., 2016). While the experiment was run on a single machine, the implementation supports scalable distributed execution. For feature preparation, the following components that are currently missing from the Spark library have been implemented manually:

- Feature Hashing: While Spark provides a framework to use feature hashing for text mining and natural language processing tasks, there are currently no libraries to apply feature hashing to column-format categorical data. We implemented this feature following the deliberations in Weinberger et al. (2009).
- String-Indexing: When predicting response variables on a test set, the standard Spark libraries currently support the following behaviors when detecting a feature that was previously unseen in the training set: Either the specific instance is skipped and not prediction is made for it, or execution fails with an error. As in our task we have multiple (theoretically) categorical columns that can take on a very wide range of possible values (such as publisher domain, consumer browser version, etc.) this behavior is unsatisfactory for our needs. To deal with this, we implemented a variant of Spark's String Indexer that deals with such cases by allowing such columns to be nullable and treating an unseen value equivalently to a NULL-entry.

The strucure of the training set as we considered it is shown in Table 4. We experimented with different subsets of features, feature hashing for features that will lead to high-dimensionality as well as different hyper-parameters for model tuning.
We observed that for similar problem dimensions (both in P and N), GBRT and GLM models performed very similarly. However, due to hardware constraints we were only able to train GBRT models low dimensions ($P \ll 10000$) at a sufficient depth for boosting to be effective. We were able to train linear models of higher dimensions by an order of magnitude. Our best model was trained on 450'000 impressions using elastic net linear regression with regularization parameter 1 and elastic net parameter 0.1, which corresponds to $\lambda_1 = 0.1$ and $\lambda_2 = 0.45$ in the notation of Section 4.1.2. As features, we used the full set of the training data, with the exception of the column Site_URL, which was hashed into

Table 3: My caption

Column	Scale	Known Set of Values	Description/Values
Weekday	categorical	yes	
Hour	categorical	yes	
Advertiser	categorical	yes	Name of the Campaign
AdEx	categorical	yes	AdExchange of the request
AdSlotFormat	categorical	yes	FirstView, SecondView, etc.
UA_os_type	categorical	yes	Desktop/Mobile
UA_os_fam	categorical	no	Operating System Family
UA_br_fam	categorical	no	Browser Family
UA_br_ver	categorical	no	Browser Version
UA_dev_fam	categorical	no	Device Family
UA_dev_brand	categorical	no	Device Brand
UA_dev_mod	categorical	no	Device Model
Region	categorical	(yes)	
City	categorical	(yes)	
Site_Dom	categorical	no	
Site_URL	categorical	no	
City_Population	ratio		
City_Long	interval		
City_Lat	interval		
AdSlotW	ratio		Width of the AdSlot
AdSlotH	ratio		Height of the AdSlot
AdSlotSize	ratio		Size in Pixels of the Slot
AdSlotFloorPrice	ratio		Hard Floor Price
UT_xxxxx	categorical	yes	User Tag Columns (values: yes/no)

Table 4: Structure of the iPinYou Dataset as used in Winning Price Prediction

5000 buckets using feature hashing. The resulting problem has 13923 dimensions. The solution has 5257 non-zero coefficients and achieves a root mean squared error of 42.96 on the training set and 44.30 on the test set, which corresponds to a Mean Absolute Test Error of 28.44 (For reference: The winning prices take values between 0.0 and 300.0.) The model has an R^2 statistic of 0.468, meaning it "explains" 46.8 per cent of the variation in winning prices.

Using OLS regression we tended to overfit the training data: While a fully unregularized linear model achieved a training set RMSE as low as 35.6, this did not generalize to unseen data (Test RMSE of 49.) Similarly, feature hashing was moderately successful on this dataset. When hashing all features with high numbers of categories (i.e. at least several hundred) down to 5000 total features, the best such model achieved RMSEs of 47.2 and 47.7. When bringing the number of features down to 1000, RMSEs of around 50 could be achieved.

Bid Landscape Forecasting

For ex-ante bid landscape forecasting, we followed the suggestions in (Ghosh et al., 2009; Cui et al., 2011) and assumed winning prices to be log-normally distributed for each considered sub-segment. As sub-segments, we considered each tuple (advertiser, weekday, hour), which results in $5 \cdot 24 \cdot 7 = 840$ separate bid landscape estimators in the training set. For each sub-segment, we calculated the empirical sample mean and variance as well as the rate of bid-requests of the full training set and defined the corresponding bid landscape as described in Section 4.2. While we did not perform any formal goodness-of fit tests, we also calculated and stored several point estimates of the empirical distribution of ω when finding the mean and standard deviance. As an example, we saved the value for $P(\omega_i < 100)$ and others for each segment. Comparing these to the estimates of the fitted log-normal distributions showed that the log-normal assumption was consistent with the data in each of the segments.

However, we found that our segment distributions from the test set did not describe perfectly the corresponding distributions in the training set. This might be due to the fact that the templates of campaign weekday and hour are inadequate, or that the test set inherently exhibits different behavior due to the way the data was generated (Compare Section 5.3.2).

Chapter 5

Bidding Strategies

In this chapter, we will analyze the problem of finding *Bidding Strategies* that determine the amount a campaign will bid for a certain impression. Throughout this chapter, we will consider the case of a single campaign k and will thus drop all indices pertaining to the campaign where the context is clear. We once again consider a stream of incoming impressions i represented by their features x_i.
As stated in Section 3.1, the goal of the DSP in a campaign is to maximize their clients utility.

Note that we will not consider any ulterior motives of the DSP in this setting. In reality, the long-term goal of any DSP—just as any other company—is naturally to maximize *its own* profits rather than those of its customers. However, we assume that the DSP is paid by the advertiser either independently of the campaign results, or that the incentives of the DSP and the advertiser are always aligned and as such the DSP acts completely in the interest of its clients. In Stavrogiannis (2014), a game-theoretic model is considered that takes into account revenues to the DSP separately to that of the advertiser. Alternatively, in modern campaigns advertisers often pay the DSP on a *Cost Per Action (CPA)* basis, i.e. the advertiser gets paid for each click or conversion that is attributed to its campaign. The payments from the DSP to the AdExchange or SSP, however are on a *Cost per Impression* (*CPM*, cost per mille = cost per 1000 impressions) basis. This opens the door for possible arbitrage opportunities to the DSP and results in a principal-agent dynamic: By maximizing the ratio of actions to impressions, it can actively affect its own profits, especially in the case of multiple concurrent campaigns. Zhang and Wang (2015) study this phenomenon from the point of view of the DSP and develop bidding strategies that maximize arbitrage. Here, however, we will restrict ourselves to the case where the DSP aims to maximize the campaign's utility as seen from the advertisers point of view. As Charles et al. (2013) note, this is intuitively useful as this will lead customers to spend more money with the DSP in the future and thus secure long-term revenues.

Unless otherwise stated, we will assume this utility to be additively determined by the individual utilities $u_{i,k}$ that we assume to be fully determined by the *(private) value* $v_{i,k}$ and the paying price of the impressions that have been won:

$$u_k = \sum_{i \in W} u_i = \sum_{i \in W} u(v_i, \pi_i) \tag{5.1}$$

Both the notions of utility and that of value used here are somewhat abstract: The actual individual utility function might or might not include the cost term π_i as we will see below and explain in Section 5.1. Furthermore, the notion of value is abstract and depends on the goal of the campaign and might also depend on the outcome (whether the impression is clicked and/or leads to a conversion.)
The simplest case is a brand awareness campaign might aim to be seen by as many people as possible, corresponding to any positive constant value of $v_k = v_{i,k}$. In this case, the utility function could then be expressed in the total payoff of the campaign when setting v_i to the *monetary* value of an impression

$$u_k^{payoff} = \sum_{i \in W} (v_i - \pi_i) \tag{5.2}$$

or by setting $v_k = 1$, in the total number of impressions achieved:

$$u_k^{Imps} = \sum_{i \in W} 1 = \#W \tag{5.3}$$

Similarly, one might measure the success of a campaign not by impressions but instead only by achieved clicks or even conversions. This can be modeled by setting $v_{i,k} = \mathbb{1}_{Clicks_k}(i) \cdot v_k^{click}$ or $v_{i,k} = \mathbb{1}_{Conv_k}(i) \cdot v_k^{conv}$, where v_k^{click} and v_k^{conv} denote the (constant) intrinsic values of a click or a conversion, respectively. Setting these parameters to the respective real monetary values will once again yield the payoff utility function. In fact, whenever we express values on the monetary scale, the resulting payoff utility function will be of the form

$$u_k^{payoff} = Profit_k = Revenue_k - Cost_k = \sum_{i \in W} v_i - \sum_{i \in W} \pi_i. \tag{5.4}$$

It should be noted, however, that assigning intrinsic monetary values to impressions, clicks and conversions might be problematic in reality: The valuations have to be supplied by the customer, as they are properties of the advertiser's business model rather than the DSPs bidding model. The customers in turn might not have the expertise, experience or resources to determine these values. Therefore, in practice, it is sometimes easier to define utility in terms of absolute numbers of impressions (clicks, conversions) rather than some monetary value.

5.1 Problem statement

We will now formally state the problem of finding optimal bidding strategies. Let I be the stream of incoming expressions during the time frame of the campaign and $N := \mathbb{E}[\#I]$ the expected total number of bid requests in the campaign. For a bid request $i \in I$ let x_i be the features of the bid request. (Additional information y_i about the bid request might be available to the DSP from tracking data or third party data providers, but w.l.o.g. we consider this information to be included in x_i.) The goal is then to find a bidding strategy, i.e. a function b that maps the request to a specific amount that should be bid. We will require b to have the following properties:

- b should only depend on the (expected) utility of the request[6] (which in turn can be determined from its features, see below) thus we might interchangeably write

$$b_i = b(x_i) = b(u(x_i)) = b(u_i) \tag{5.5}$$

- b must be monotonously increasing in u, i.e. a higher expected utility will never result in a lower bid:

$$\frac{\partial b}{\partial u} \geq 0. \tag{5.6}$$

In theory, the bid strategy problem can then be modeled as the following optimization problem

$$\begin{aligned} \max_{b(\cdot)} \quad & u \\ s.t. \quad & \sum_i \pi_i \leq B. \end{aligned} \tag{5.7}$$

That is, one aims to maximize the advertisers utility while respecting the campaigns budget constraint. However, the true values of impressions are only observed ex-post (as they might depend on outcomes such as clicks or conversions), as are the winning prices and the set of won impressions. The problem in (5.7) is thus ill-defined. Instead, we will replace it with the problem

$$\begin{aligned} \max_{b(\cdot)} \quad & \mathbb{E}[u] \\ s.t. \quad & \mathbb{E}\left[\sum_i \pi_i\right] \leq B. \end{aligned} \tag{5.8}$$

[6]Additionally, b may possibly depend on the current state of the campaign, which will however be omitted from any notation.

where the objective function and the payments have been replaced by their respective expectations. Any bidding strategy will have to solve this problem in an online-fashion using only the information that is available at the time a bid-request comes in.
In order to find these expectations we will introduce some additional assumptions that determine the availability of certain estimators. Not every bidding strategy will rely on all of these assumptions to be fulfilled but we nevertheless choose to introduce them all in one place:

1. All impressions x_i are i.i.d. drawn from some (unknown) distribution p_x over the feature space $\mathcal{X} \subset \mathbb{R}^P$.

2. The private valuation v_i is fully determined by the features of the request through the value function $v : \mathcal{X} \to \mathbb{R}_{\geq 0}$ that is identical for all requests:

$$v_i = v(x_i) \tag{5.9}$$

   Note that other factors than the original P features might play a rule in determining v_i: Sometimes the *ex-post value* $\tilde{v}$ is only available after winning an impression and observing its outcome. For example, when measuring campaign success in terms of clicks, the ex-post value of an impression would be

$$\tilde{v}_i = v^{\text{click}} \cdot \mathbb{1}(i \in Clicks|i \in W) \tag{5.10}$$

   In these cases we would like the private valuation to approximate the ex-post value using interim estimators for unobservable terms, such as the Click Through Rate $CTR_i = P(i \in Clicks|i \in Win)$ leading to a value function of the form $v_i = v(x_i, CTR_i)$. For a given impression we will then assume the availability of a sufficiently accurate estimator of this factor which only depends on x_i. In the case of Click Through Rate, this would mean the availability of an estimator

$$\widehat{CTR}_i := eCTR(x_i) \approx \mathbb{E}\left[CTR_i|x_i\right]. \tag{5.11}$$

   which is found using the methods described in Section 3.2. In this case, we can then replace the true value by its expected value which ultimately only depends on x_i:

$$v_i := v(x_i, \widehat{CTR}_i) = v(x_i) \approx \mathbb{E}\left[\tilde{v}_i\right] \tag{5.12}$$

   For some bidding strategies, we will also assume the availability of an ex-ante estimate of the distribution of v

$$\hat{p}_v(v(x)) \approx \frac{p_x(x)}{||\nabla v(x)||} \tag{5.13}$$

   which can be found using training data.

3. In the interim stage, i.e. once the features of an impression x_i become known but before submitting a bid and observing the outcome, we assume the existence of an estimator for the winning price ω

$$\hat{\omega}_i := \hat{\omega}(x_i) \approx \mathbb{E}\left[\omega_i|x_i\right] \tag{5.14}$$

4. In the ex-ante stage, i.e. for future impressions where x_i cannot yet be observed, we assume the winning prices of impressions to be distributed according to some distribution p_ω that is independent of x.[7] Let F_ω be the cumulative distribution function of p_ω. Then we further assume the availability of a *winning function* $w : \mathbb{R}_{\geq 0} \to [0, 1]$ that describes the possibility of winning the impression given a bid b:

$$w(b) \approx F_\omega(b) = P(\omega \leq b) = P(b \text{ wins the impression}) \tag{5.15}$$

   To be a consistent cdf, we require that $w(0) = 0$ and $\frac{\partial w}{\partial b} \geq 0$.

5. Furthermore we will assume to have an estimate $\hat{\alpha}$ that describes the rate of bid requests (dimension: impressions/time). Together, $w(\cdot)$ and $\hat{\alpha}$ then describe the expected bid landscape. Additionally, using $\hat{\alpha}$ we can estimate the total number of future bid requests:

$$\hat{N} := \hat{\alpha} \cdot \text{time in campaign} \approx N \tag{5.16}$$

[7] We will discuss this assumption in detail in Section 5.2.3

6. In the ex ante stage, we estimate the distribution of paying prices (conditional on winning an impression) to be identical to that of the winning prices

$$\hat{p}_\pi \approx p_\omega. \tag{5.17}$$

This would mean that all impressions are sold in second price auctions, which is not the case whenever $\omega_i \geq b_i < \sigma_i$, as in this case $pi_i = b_i > \omega_i$. Thus, for ex-ante cost estimation we do not consider the existence of soft floors.

Similarily, in the interim stage, we estimate π_i by the biased estimator

$$\pi_i := \hat{\omega}_i. \tag{5.18}$$

7. Finally, using items 2 and 6 we approximate the expected utility of an impression i by

$$\hat{u}_i := u(v_i, \hat{\pi}_i) \tag{5.19}$$

and its ex ante distribution $\hat{p}_u$ depending on $\hat{p}_\pi$ and $\hat{p}_v$, although the exact transformation depends on the form of u.

When all these assumptions are made, (5.8) can be written as

$$\begin{aligned} \max_{b(\cdot)} \quad & \hat{N} \cdot \int_x \hat{u}(x) w(b(\hat{u}(x))) dp_x(x) \\ s.t. \quad & \hat{N} \cdot \int_x \hat{\pi}(x) w(b(\hat{u}(x))) dp_x(x) \leq B \end{aligned} \tag{5.20}$$

where the integral terms represent the ex-ante expected utility and payment for an impression. Note that in this formulation p_x remains unknown. While the objective function can be rewritten in terms of p_u and thus be brought into a form that can be (at least numerically) computed, this is not possible for the integral in the constraint: Both π and u depend on x and their relationship is unclear. Nevertheless, we found it useful to state the problem in the form of (5.20) as we will later introduce several heuristics that deal with the conundrum of computing the constraint term in different ways (see Sections 5.2.2, 5.2.3).

Using these ex-ante interim and ex ante estimators, different heuristics use various definitions of value and utility as well as different ways to rewrite (5.8) or (5.20) and thus transform the problem in this general form into more specific versions. Not every heuristic will make use of all of the assumptions above: Some will require fewer assumptions, which will usually lead to simpler bidding functions but often is not able to react well to the budget constraint. Other models will be more involved and require additional parameters, but will have the advantage that they can dynamically adapt to the state of the campaign by re-tuning their parameters and refining the estimators based on the results so far and the remaining time and budget in the campaign. We will consider some of these strategies in the next section.

5.2 Bidding Strategies

In this section, we will motivate and describe concrete bidding strategies. We will start with the most naïve methods that were popular in the early days of RTB.

5.2.1 Naïve Strategies

Constant Bidding

The simplest form of bidding strategy is to always bid a constant amount c:

$$b^{\text{const}}(x_i) = c \tag{5.21}$$

This strategy was very wide-spread in the early days of RTB (Compare Perlich et al., 2012; Liao et al., 2014) and the bidding constant c for a campaign would be set manually by employees of the DSP or the advertiser based on market expertise. Comparing this to auction theoretic approaches such as "Truthful Bidding" (Section 5.2.2), this seems justifiable when the goal of the campaign is to maximize the number of impressions (rather than clicks or conversions), by identifying the private valuation of any impression as

$$v_i := c \tag{5.22}$$

in which case constant bidding is equivalent to truthful bidding.

Random Bidding

Constant bidding has the disadvantage, that the SSP can easily observe the strategy and adapt its reserve prices accordingly. An easy fix for this is choosing ones bid randomly between some bid constraints b_{min} and b_{max}:

$$b_i^{rand}\ U(b_{min}, b_{max}) \tag{5.23}$$

5.2.2 Auction Theoretic Approaches

Truthful Bidding

It's a well known result from action theory, that in a single-auction i it is a weakly dominant strategy to bid one's true value v_i (Compare, e.g. Krishna, 2009, Chapter 2): If we define the utility u_i as the *payoff* to the bidder, i.e.

$$u_i^{payoff} = \begin{cases} v_i - \pi_i, & \text{win} \\ 0 & \text{loose} \end{cases} \tag{5.24}$$

then bidding $b_i = v_i$ on the one hand guarantees that the payoff can never be negative. On the other hand, the payoff in a won auction does not depend on b_i and bidding v_i maximizes the probability of winning the auction and thus the expected utility.
This naturally leads to a simple bidding strategy, *Truthful Bidding*, described in algorithm 1.[8]

Algorithm 1: Truthful Bidding

Input: B: budget, I: Stream of bid requests
while $B > 0$ *and* $\exists$ *next bid-request* i **do**
  bid $\max(v_i, B)$
end

while optimal for a single auction, this strategy does not take into account several issues:

- It completely ignores the budget constraint and might thus spend inventory too quickly and miss out on more valuable impressions that occur later.
- While truthful bidding is a dominant strategy for a single auction, and thus follows from a game-theoretic approach, this strategy is problematic for future repeated auctions: When all DSPs bid in a truthful fashion, the ad exchange may be able to observe a pattern of valuations from the bids submitted and adjust its floor prices accordingly. A long term best response of the ad exchange to all DSPs bidding v_d truthfully would thus be setting the floor price to $\min_d v_d$ and with that the long-term expected payoff to the advertisers would just be 0.

A simple mechanism that deals with the second point is *Random Bid Shading*, where the DSP uses random bidding (Section 5.2.1) with the true valuation as upper bound.

[8] Note that for all further bidding algorithms in this Section, we omit the algorithm formulation and minor details such as never bidding above budget. Consider this algorithm a surrogate for the different strategies to come.

Bid Shading and the Fluid Mean Field Equilibrium

Since the market of Real Time Bidding with repeated Vickrey auctions and budget constraints is quite complex and thus difficult to describe mathematically as a dynamic game with incomplete information, classical game theoretic solution concepts such as the Bayes-Nash equilibrium are not tractable and to our knowledge no such concepts have been proposed for this problem. However, new equilibrium concepts have been designed specifically to describe an RTB market. Balseiro et al. (2015) investigate an RTB market with one exchange, multiple bidders with different types and second price auctions, where for each bidder d[9] and every impression i, the values $v_{d,i}$ and paying prices $\pi_{d,i}$ are i.i.d. distributed according to p_{v_d} and p_{π_d} and are independent of each other. Here, paying prices are interpreted as the maximum of bids by other players, where the (hard) reserve price is considered an additional bid submitted by the ad exchange itself and soft floor prices are not present. The bid requests of the ad exchange are modeled as a stochastic process and each bidder is presented with a subset of the requests according to time and relevance constraints. Each player is then faced with a an optimization problem of the form

$$\begin{aligned} \max_{b(\cdot)} \quad & N \cdot \mathbb{E}\left[\mathbb{1}\left(\pi \leq b(v)\right)(u-d)\right] \\ s.t. \quad & N \cdot \mathbb{E}\left[\mathbb{1}\left(\pi \leq b(v)\right)\pi\right] \leq B \end{aligned} \tag{5.25}$$

While utility is defined in terms of payoff, Balseiro et al. differ from our problem statement above in that the bidding function only depends on value v_i rather than utility u_i. Taking this into account, we could use our framework and rewrite (5.25) as a variation of (5.20):

$$\begin{aligned} \max_{b(\cdot)} \quad & \hat{N} \cdot \int_u \hat{u} w(b) dp_u(u) \\ s.t. \quad & \hat{N} \cdot \int_{(v,\pi)} \hat{\pi} w(b(v)) d(p_v \otimes p_\pi)(v,\pi) \leq B \end{aligned} \tag{5.26}$$

Balseiro et al. show that an optimal solution to (5.25) is given by bidding

$$b^{\text{FMFE}}(x_i) := \frac{v_i}{1+\mu,\hat{C}} \tag{5.27}$$

where $\mu, \hat{C}$ is the optimal solution of the dual problem, i.e. the Lagrange multiplier for the budget constraint in the primal problem. We can thus interpret $\mu, \hat{C}$ as the marginal utility of an additional unit of budget to the campaign. Thus, if the budget constraint is inactive in the optimization problem, i.e. the campaign is not expected to spend its entire budget, then $\mu, \hat{C} = 0$ and the campaign should bid its true value. Otherwise $\mu, \hat{C} > 0$ and the campaign should shade its bid by the constant factor $1 + \mu, \hat{C}$.

This strategy has the desirable properties that $\mu, \hat{C}$ can be found efficiently and that it constitutes a *Fluid Mean Field Equilibrium*: Loosely speaking, this means that if all DSPs bid according to b_d^{FMFE}, the price distributions p_{π_d} for each DSP will be of such a form that for each DSP d, b_d^{FMFE} is the optimal solution to (5.25).

5.2.3 "Optimal" Real Time Bidding

Zhang et al. (2014b) take an approach that most closely resembles our general framework: They consider the problem of finding an optimal bidding strategy to be a decision problem that they aim to solve using functional optimization. The DSP is considered to behave as a price taker: Market prices are fixed and the bids submitted by the DSP will not influence the price landscape in the medium run.

In our framework, starting from (5.20), they make the following two assumptions:

- Utility is simply defined in terms of value: $u(x_i) = v_i$.[10]

[9] More formally, for each type of player.

[10] In Zhang et al. (2014a), v itself is actually defined to be the click through rate, but the method can be easily generalized to any definition of value that can be computed directly from x.

- Payment prices are approximated using the *bidding price* as an upper bound: $\hat{\pi}_i := b_i$.

While it might seem unreasonable to not include the cost term in the utility function, this can be justified by the fact that the value of every impression is always nonnegative and usually strictly positive and therefore, in an optimal solution, a campaign should spend all of its budget. When the budget constraint is thus active, the total cost is fix at $\sum_i \pi_i = B$ and the total payoff of the campaign thus depends only on the value generated. Nevertheless, this concept might allow overbidding and thus negative payoff on individual impressions.
The second assumption is used to solve the conundrum of evaluating the integral in the budget constraint that depends on p_x: By replacing π with b which is fully determined by v due to the first assumption, the integration term only depends on the distribution of v, which can be estimated using bid landscaping. As such, (5.20) can be written as

$$\begin{aligned} \max_{b(\cdot)} \quad & N \cdot \int_v v \cdot w(b(v)) dp_v(v) \\ s.t. \quad & N \cdot \int_v b(v) w(b(v)) dp_v(v) \leq B \end{aligned} \tag{5.28}$$

This problem can be solved using calculus of variations and the optimal solution b^{ORTB} can be characterized by the Euler-Lagrange condition:

$$\lambda w(b(v)) = (v - \lambda b(v)) \frac{\partial w(b(v))}{\partial b(v)} \tag{5.29}$$

Once again, λ is the optimal Lagrange multiplier of the budget constraint in (5.28). It should be noted that this solution concept breaks down when $\lambda = 0$, which is consistent with the implied assumption in the model setup that all available budget is to be spent. Therefore it's reasonable to replace the inequality constraint in (5.28) with equality.

ORTB with a Log-Normal Winning Function

While the authors consider two classes of concave bidding functions that have closed-form solutions for b^{ORTB}, we are additionally interested in the setting with log-normal bidding functions, as we will consider these in our experiment in Section 5.3.2. Therefore we will deduce the Euler-Lagrange condition for the case of log-normally distributed winning prices $\omega \sim \text{Log-Normal}(m, s)$ with location parameter m and scale parameter s. We have

$$w(b) = F_\omega(b) = \Phi\left(\frac{\log(b) - m}{s}\right) \tag{5.30}$$

and thus

$$\frac{\partial w}{\partial b} = \frac{1}{sb} \phi\left(\frac{\log(b) - m}{s}\right) \tag{5.31}$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are the cdf and pdf of the standard normal distribution, respectively. The Euler-Lagrange condition (5.29) thus becomes

$$\lambda \Phi\left(\frac{\log(b) - m}{s}\right) = \frac{v - \lambda b}{sb} \phi\left(\frac{\log(b) - m}{s}\right) \tag{5.32}$$

Similarities to Bid Shading

Conceptually, the motivation ORTB is quite similar to that of the FMFE bid shading strategy in Section 5.2.2. We observe that for very specific conditions on the bid landscape and the budget, they do take in fact a very similar form:

Proposition 5.1. *Let* $(b(\cdot), \lambda)$ *be primal- and dual-optimal solutions to* (5.28) *with* $\lambda > 0$. *If in this optimal solution* λ *is equal to the winning-rate elasticity of the bid price, then the optimal bidding function is characterized by the closed-form solution*

$$b^{\text{ORTB}}(x_i) = \frac{v_i}{1 + \lambda} \tag{5.33}$$

While the condition in the proposition does not seem to have a trivial interpretation and it is not clear whether such solutions exist, it's an enticing observation that there might be a deeper relationship between FMFE and ORTB bidding.

5.2.4 Knapsack Model and Threshold-Bidding

The Knapsack Model

An alternative way of looking at (5.8) is by modeling it as a knapsack problem. For a moment, let's not consider the amount we want to bid on each impression, but instead on finding the optimal set of impressions that we would like to win. We will start in an offline setting: Let us assume we know the entire set of bid requests including each impressions utility and market price and that for any incoming bid request, we may choose to either buy it at the market price π_i (denoted by $w_i = 1$) or not ($w_i = 0$). Then the problem of finding the optimal set of impressions to buy can be modeled as an Integer Linear Program (ILP) which is an instance of the *Knapsack Problem*

$$\begin{aligned} \max_{w} \quad & \sum_{i \in I} w_i u_i \\ s.t. \quad & \sum_{i \in I} w_i \pi_i \leq B \\ & w \in \{0,1\}^N \end{aligned} \tag{5.34}$$

Solving the Knapsack problem is generally NP-hard. However, it is a well-known fact that the continuous variant of the problem, where integrality constraint is replaced by $0 \leq w_i \leq 1$, fulfills the Greedy property and can be solved efficiently in $\mathcal{O}(N \log N)$ time using Algorithm 2. (Compare Korte et al., 2012, Chapter 12)

Algorithm 2: Greedy Solution to the Continuous Knapsack Problem

Input: B: total budget, I: Set of bid requests with utility u_i and cost π_i
Order the bid requests by u_i/π_i, let $i^{(n)}$ be the request with the n^{th} highest ratio
for $j \in \{1, \dots, N\}$ **do**
  if $B \geq \pi_{i(j)}$ **then**
    Set $w_{i(j)} := 1$
    Set $B := B - pi_{i(j)}$
  else if $B > 0$ **then**
    Set $w_{i(j)} := B/\pi_{i(j)}$
    Set $B := 0$
  else
    Set $w_{i(j)} := 0$

In this continuous knapsack setting, the decision variable w_i can be interpreted as our winning function $w(b_i)$, and as we required w to be strictly monotonous, it is invertible and we can therefore determine $b_i := w^{-1}(w_i)$. However, as in our problem setting N is very large and we have $\pi_i \ll B$, Algorithm 2 will assign values of 0 or 1 to w_i for the vast majority of impressions and in fact there would be only a single impression that would be assigned a nonintegral winning probability. The method thus essentially tells us to bid only on the most "valuable" impressions as measured by the ratio u_i/π_i. When utility is defined as payoff, this ratio takes on the form of *Return on Investment (ROI)*:

$$R_i := ROI_i := \frac{u^{\text{payoff}}}{\pi_i} = \frac{v_i - \pi_i}{\pi_i}. \tag{5.35}$$

Going back to the online setting, we may then use our ex-ante estimators $\hat{p}_v$ and $\hat{p}_\pi$ to determine the estimated distribution $\hat{p}_R$, or alternatively use the empirical distribution of R from the training set. Using this distribution, we can then calculate a *cutoff* $\hat{C}$ such that we expect to spend exactly our entire budget when buying all impressions that have an expected ROI higher than $\hat{C}$

$$\begin{aligned} \hat{C} := \quad & \arg\min_{C>0} \quad C \\ & s.t. \quad \hat{N} \cdot \mathbb{E}_{p_R \otimes p_\pi} \left[\mathbb{1}(R > C)\pi \right] \leq B \end{aligned} \tag{5.36}$$

These deliberations then naturally give rise to a bidding strategy.

ROI Based Bidding

With the cutoff $\hat{C}$ found using ex-ante estimation, for any bid request i we can use the our private valuation v_i and the interim stage estimator $\hat{\pi}$ to determine the impression's expected ROI:

$$\hat{R}_i := \frac{v_i - \hat{\pi}_i}{\hat{\pi}_i}. \tag{5.37}$$

While the optimal solutions w_i above suggest that the winning function should be maximized for any impression (i.e. $b_i \to \infty$/maximum possible bid) it seems reasonable to instead only bid the private valuation v, as in this way, we can assert that there will never be negative payoff. The ROI-based bidding strategy is then given by

$$b^{\mathrm{ROI},1}(x_i) = \begin{cases} v_i, & \text{if } \hat{R}_i \geq \hat{C} \\ 0, & \text{otherwise.} \end{cases} \tag{5.38}$$

Bid Shading for ROI Based Bidding

It turns out that the strategy in (5.38) can be improved: Consider the following bidding strategy:

$$b^{\mathrm{ROI},2}(x_i) = \frac{v_i}{1 + \hat{C}} \tag{5.39}$$

We show that (in the absence of budget constraints), this strategy wins exactly those impressions with ROI above the cutoff.

Proposition 5.2. *For an impression i, let v_i be its private valuation, π_i its unobserved market price and $\hat{\pi}_i$ an interim stage estimator for π_i.*
Then $b^{\mathrm{ROI},2}$ wins an impression if and only if $ROI_i \geq C$.

Therefore $b^{\mathrm{ROI},2}$ dominates $b^{\mathrm{ROI},1}$ in the following sense:

Corollary 5.3. *Let the conditions of Proposition 5.2 hold. Then*

1. *Any impression won by $b^{\mathrm{ROI},1}$ with $ROI_i \geq \hat{C}$ will also be won by $b^{\mathrm{ROI},2}$.*
2. *Any impression won by $b^{\mathrm{ROI},1}$ but not by $b^{\mathrm{ROI},2}$ will have $ROI_i < \hat{C}$.*

Additionally, $b^{\mathrm{ROI},2}$ will also win those impressions that $b^{\mathrm{ROI},1}$ passes on (due to overestimating π_i and thus underestimating ROI_i) but that nevertheless achieve the required ROI-level.
Putting these aspects together, choosing the variation with bid-shading not only seems more appropriate for an online solution to (5.34) than $b^{\mathrm{ROI},1}$, but also makes an interim stage market price estimator obsolete. Furthermore, the form of (5.39) closely resembles that of (5.27), which intuitively suggests it might share some of the same desirable game theoretic properties. Determining a specific relationship might be part of a future work.

5.2.5 Other Models

In this final section of bid strategies, we will introduce some models that do not follow any of the more rigorous approaches from an optimization or game theoretic perspective, but instead rely on intuitive reasoning only.

Linear-form Bidding

One aspect in our framework that can be problematic in real life is determining any notions of value that are exogenous to our framework (such as the true private value of a click v^{click} or of a conversion v^{conv}) that might be required to use most of the strategies discussed above. Perlich et al. (2012) note that often, these are simply unknown. When considering campaigns that aim to maximize clicks or conversions, they therefore propose to bid proportionally to the *score ratio* or *Inventory Score* $\Phi(x_i)$ of a bid i. This score is defined as the probability of the desired event occurring for the specific bid (interim phase estimator) with respect to the average probability in the entire market (ex-ante estimator). For a campaign that maximizes clicks as its goal, the strategy is thus given by

$$b_i^{\text{Lin}} = b^{\text{base}} \cdot \Phi(x_i) = b^{\text{base}} \frac{\widehat{CTR}_i}{\widehat{CTR}_k} \approx b^{\text{base}} \frac{P(i \in Click | i \in Win,\ x_i)}{P(i \in Click | i \in Win)} \tag{5.40}$$

where b^{base} is set to be the *basis bid level.* While we hadn't mentioned an ex-ante CTR estimator before, this can be easily achieved by taking the average CTR from the training set or directly from the running camaign. This method has later been called *Linear-form* bidding in Zhang et al. (2014b), Zhang et al. (2014a) and others, and we will refer to it by that name as well.

However, this strategy naturally brings up the question of how to determine the basis bid level, which on first impression seems just as hard as determining a private value, which was the motivation for this method in the first place; Perlich et al. simply describe b^{base} as "determined by the account management team" based on the industry of the campaign. This seems unsatisfactory at a first glance. It should be noted however, that since the paper is from 2012 and thus one of the earlier papers in RTB research, the strategy was originally contrasted to naïve methods such as constant bidding, that also require some form of human input. Furthermore, determining b^{base} exogenously using domain knowledge might in fact be more feasible than doing so with v^{click} or v^{conv}: Conceptually, the basis bid level is much more similar to the value of a single impression or the CPM than to the values of clicks or conversions. Therefore, staff at a DSP might very well have the domain expertise to determine reasonable values based on experience with past campaigns (as opposed to private values of clicks or conversions which will be highly dependent to the specific customer's business model.) Finally, as with other parametric approaches, we may use b^{base} as a parameter that can be used for budget pacing purposes.

Lift Based Bidding

Xu et al. (2015) consider CPA-based campaigns, i.e. those that aim to drive conversions or other actions. In light of different attribution models and retargeting, they observe that there is a certain background conversion rate, i.e. the probability that a user might drive a conversion that is attributed to our campaign even when we *do not* show him a further ad. The authors thus propose that the valuation of the impression should not be based on the user's conversion rate itself, but rather on the expected difference in conversion rate when winning the ad to the background conversion rate, which they call "Action Lift". They show promising empirical results using online experiments. Thus, while beyond the scope of our framework, their approach should be included here for completeness.

5.3 Benchmarking

5.3.1 Experiment Setup

Ideally, we would like to test these bidding strategies empirically in a real world setting. Unfortunately, however, academic access to real-world RTB campaigns is restricted due to privacy and business concerns. While some authors have been able to test their models in real-world settings, this was not an option in the context of this work.
To mitigate these circumstances, we will resort to the Season Two Test Set of the iPinYou Contest Dataset (Liao et al., 2014). The test set contains RTB logs of five campaigns collected on the 13$^{\text{th}}$ to 15$^{\text{th}}$ June, 2013. An overview of the original campaigns can be seen in Table 5.

Advertiser	Impr.	Clicks	Conv	Budget	CTR	CPM	CPC
Oil	536'795	366	0	46'356	0.068%	86.36	126.66
Software	300'928	260	58	34'160	0.086%	113.51	131.38
Tire	523'848	287	11	43'628	0.055%	83.28	152.01
Chinese eCommerce	614'638	515	0	45'216	0.084%	73.57	87.80
Intl. eCommerce	545'421	445	0	45'715	0.082%	83.82	102.73

Table 5: Campaigns in the iPinYou Season 2 Test Set. (All monetary values in Chinese Fen = Chinese Yuan/100.)

We will run separate experiments for each of the original five campaigns in the dataset. For these experiments, we will consider all impressions that have originally been won by the corresponding campaigns to be the incoming bid requests that are potentially relevant to the advertiser. In the experiment, we assume each campaign to be endowed with a fraction of the budget of the original campaigns in the dataset. In this setting, no campaign will be able to "buy up" the entire market. A similar setup has been used in Zhang et al. (2014b,a) and other benchmarking simulations. We have considered the following fractions of the original budgets: ½,⅓,¼,⅛,⅒,1⁄20, 1⁄100.
Campaigns were set up with the goal of generating clicks, i.e. the private valuations were defined as

$$v_i := v^{\text{click}} \cdot \widehat{CTR}_i. \tag{5.41}$$

with v^{click} being set to the CPC of the original campaigns from the test set.

It should be noted that this data basis can by no means be considered perfect due to the fact that effects such as selection bias (only impressions won by the original campaign) might distort the dataset such that it no longer resembles an unadulterated stream of bid requests. Due to the censoring of lost bid requests in the dataset the existence of such effects can, however, neither be confirmed nor ruled out. In the absence of more trustworthy data, we will therefore content ourselves with the assumption that the stream of bid requests won in this way will in fact be an adequate representation of reality.

Estimators and Priors

As many of the bidding strategies require CTR and Winning price estimators, as well as priors on the density of bid requests, expected campaign wide ctr and the bid landscape, among others, the training set of iPinYou Season 2 has been used to generate this information.

CTR and Winning Price

The interim-stage winning rate predictor has been trained using L_1-regularized Linear Regression on a sample of 500.000 observations of the training data. We use a CTR estimator supplied by the author of Stratakos (2016) which is based on a Gradient Boosted Regression Tree method used on the Season 2 Training data.

Bid Landscape

For estimating bid-landscapes, we proceed as follows: First, we used the entire training set for each campaign k and pair of weekday and hour h, and determined number of bid requests $N_{k,h}$, the mean $\mu_{k,h}$ and standard deviation $\sigma_{k,h}$ of the (so far unspecified) distribution of winning prices $p_{\omega_{k,h}}$. $N_{k,h}$ was used as an estimate for the hourly rate of incoming bid requests. Since the test set runs across exactly 7 days, we thus could estimate these values for any possible pair (k, h).
Next, we assumed all winning functions to be log-normal distributed. To be consistent with the mean and variance of the sample winning distribution, for each hour and campaign, we thus estimated

$$\omega_{k,h} \sim \text{Log-Normal}(m_{k,h}, s_{k,h}) \tag{5.42}$$

Finally, we noticed that there were sometimes significant differences between the training landscape achieved in this way and the actual landscapes in the training data, which is likely due to selection bias as explained earlier. (The test set is set at a time where some of the campaigns are approaching their end and behavioral changes are visible in the data. Therefore it is likely that the Make-up of impressions won might have changed, most notably the rate of won impressions became lower when compared to the training set.) In order to smoothen the bid-landscape we thus used as the current bid landscape bl_c of the campaign a mixture of the training landscape bl_t and the actually observed landscape bl_p during the previous hour of the campaign. Given weights $\lambda_t, \lambda_p = (1 - \lambda_t)$ we defined as the mix of these landscapes as follows:

$$N_c := \lambda_t \cdot N_t + \lambda_p \cdot N_p \tag{5.43}$$

$$\mu_c := \frac{\lambda_t \mu_t + \lambda_p \mu_p}{N_c} \tag{5.44}$$

$$\sigma_c := \sqrt{\frac{\lambda_t N_t \sigma_t^2 + \lambda_p N_p \sigma_p^2 + \lambda_t N_t (\mu_t - \mu_c)^2 + \lambda_p N_p (\mu_p - \mu_c)^2}{N_c - 1}} \tag{5.45}$$

Adaptive Bidding

Using these hourly training bid landscapes, we had each campaign update its current expected bid landscape once every hour and used this to hourly re-tune the parameters for any adaptive bidding strategy.

This corresponds to solving the optimization problem (5.8) once every hour, achieving a bidding strategy, bidding according to this strategy for one hour and then repeating with updated campaign parameters.

In order to do solve the problem in models that take parameters, we numerically solved the dual problem in an EM-fashion by alternatingly updating the dual (i.e. $\lambda, \mu, \hat{C}$) and primal parameters until reaching convergence. Expected future spend was calculated as the sum of expected hourly spends using the bidding functions of the respective landscape forecast and the estimated average bid using current parameters.

5.3.2 Simulation Results and Analysis

As expected, it turns out that the more sophisticated methods, such as ORTB, Bid Shading (derived from either FMFE or ROI-based bidding) and Linear Form Bidding vastly outperform the more naive methods. In fact, out of the methods considered here, these three seem to be competitive, although their exact performance depends on the sub-experiment. As an exemplary output, the results of different strategies on the campaign “Oil” with ⅛ of the original budget is shown in Table 6. As a benchmark, we included the original bidding in the test data set, which due to the experiment setup is equivalent to buying every bid request at its market price. While in this example the ShadedValueBidders seem to be absolutely dominating, it should be noted that for this strategy we did some manual tuning:

In order to solve the dual problem, and thus find the optimal shading factor, we need an estimator of the average bid level over the rest of the campaign. A simple approach would we to take the bid corresponding to the average campaign wide CTR. It turns out, however that since the distribution of the CTR is skewed, such an estimator vastly overestimates future spend for the BidShading strategy which will lead to lower bids and suboptimal results. We therefore performed some manual tuning in order to lower the shading factor and increase spending.

Other methods such as the LinearBidder do not suffer from this problem, as the form of the score ratio automatically ”unskews” the distribution of values.

Full results of all campaigns are presented in Appendix B for the more competitive Bid Strategies. Achieved clicks (ex-post and interim estimated clicks) are shown in Figure 6. We observe that ShadedBidding (tuned) seems to have the best performance, in particular in simulations with tight small budget constraints. This is further reinforced by reading Figures 8 (CPC, CPM), 9 (CTR, WR) and 10 (ROI). Figure 7 shows campaigns’ actual spending versus their target spending (i.e. budget constraint.) A value of 1 means a campaign spent the entire budget that was allotted to it. We can see from the charts, that our adaptive methods had trouble spending all of their budget for those campaigns with looser budget constraints. This is in spite of the fact that at most stages during the campaigns, all dual parameters

bidder_type	**Imps**	**clicks**	**Spent**	**ctr**	**cpm**	**cpc**	**Payoff**	**roi**
OriginalBidding	66349	41	**1.00**	0.00062	87.34	141.332	-601	-0.10
ConstantAvgValue	102038	45	**1.00**	0.00044	56.79	128.768	-95	-0.01
TrueValue	93554	99	**1.00**	0.00106	61.94	58.532	6744	1.16
RandomUpToTrue	**112928**	275	0.95	0.00244	48.81	20.044	29318	5.31
LinearBidder	83877	70	**1.00**	0.00083	69.08	82.78	3071	0.53
ORTB	83130	54	**1.00**	0.00065	69.7	107.307	1044	0.18
LinearWithPacing	88921	**291**	0.80	0.00327	52.22	15.957	**32213**	6.93
ORTB	110785	216	0.95	0.00195	49.79	25.539	21841	3.95
ShadedValueBidder1	33438	213	0.23	**0.00637**	40.3	**6.327**	25630	**19.01**
ShadedValueBidder2	40679	249	0.27	0.00612	**39.08**	6.385	29947	18.83

Table 6: Campaign results, Advertiser: Oil, ⅛ of original budget.

were found to be nonzero, i.e. the campaign considered itself budget constrained when it really wasn't. Without manual tuning, ShadedBidding performed similarly to the Linear Bid strategy in terms of clicks throughout the campaigns, but spent less than a third of its budget, except in the most constrained campaigns. It should also be noted that in Zhang et al. (2014a), ORTB performed better than in our benchmark, as the authors had chosen the dual parameter manually as well. We believe that the suboptimality in spending comes from the fact that (a) we have some selection bias in the data and the campaigns in the test set behave differently from the training set. This can be seen for example in Figure 11, which shows the hourly progress of 5 sample campaigns over the time frame of the training set. It's visible that some campaigns (Software, red line) did not receive any bid requests for hours at a time, which might be due to the fact that the original campaigns were about to end and were severely budget constrained during the last day of the training set. This leads us to believe that our trained Bid Landscape does not accurately reflect the landscape in the training set, especially with respect to the hourly rate of bid requests, which in turn might explain why all adaptive campaigns seem to be spending "too little money."

Nevertheless, our adaptive models outperform the baseline and simple models (such as constantly bidding ones average private value) by orders of magnitude in any metric considered.

Chapter 6

Conclusion and Outlook

In this thesis, we gave an introduction to the field of Real-time Bidding as seen through the eyes of a demand side platform. We explained the main challenges that a DSP faces and dealt in detail with the issues of predicting market prices and finding optimal bidding strategies. As our main focus, we compared various bidding strategy models both analytically and empirically. While the results are promising, structural problems in the experimental data make it a natural next step to test these bidding strategies in practice with real world data, as well as working on adapting these strategies to more sophisticated campaigns that make heavy use of retargeting and storytelling.
Furthermore, the structural similarities in the final bidding strategies b^{FMFE}, $b^{ROI,2}$ and possibly b^{ORTB} in spite of their very different model formulations warrants further work about how the models behind the method might be related to one another. Our analytical work on bidding strategies showed that an ex-ante estimation of winning price is much more important in practice than an interim estimator, as the value of price prediction lies mostly in solving dual problems in order to optimize adaptive bidding strategies and in these problems we cannot rely on interim-stage data. Nevertheless, our winning price prediction model showed that a sizable share of winning price variance can indeed be explained by the features, and thus that the assumption $\omega(b, x) = \omega(b)$ should not go unchallenged. In light of this, a mixed template model as described in Section 4.2.2 might be the way forward in Bid Landscaping.

Appendices

Appendix A - Proofs

Proof of Proposition 5.1.
First of all, w is strictly monotonously increasing in b, thus w is invertible and we can write $b(w) = w^{-1}(w)$. Thus the winning rate elasticity of the bid price is well-defined by

$$\varepsilon_{b,w} = \frac{b}{w}\frac{\partial w}{\partial b} \tag{1}$$

Using $\lambda > 0$, we can rewrite the Euler-Lagrange condition (5.29):

$$\lambda w = (v0\lambda b)\frac{\partial w}{\partial b} \tag{2}$$

$$\frac{\lambda w}{\frac{\partial w}{\partial b}} = v - \lambda b \tag{3}$$

$$\lambda b = v - \frac{\lambda w}{\frac{\partial w}{\partial b}} \tag{4}$$

$$1 = \frac{v}{\lambda b} - \frac{\frac{\lambda w}{\lambda b}}{\frac{\partial w}{\partial b}} = \frac{v}{\lambda b} - \varepsilon_{w,b} \tag{5}$$

$$1 + \varepsilon_{w,b} = \frac{v}{\lambda b} \tag{6}$$

$$b = \frac{v}{\lambda + \lambda\varepsilon_{w,b}} \tag{7}$$

The proposition follows with $\varepsilon_{w,b} = {}^{1}/_{\varepsilon_{b,w}}$. □
Proof of Proposition 5.2.

"⇒" Let i be an impression won that is won by $b^{\mathrm{ROI},2}$, then it follows that

$$b_i = \frac{v_i}{1+\hat{C}} \geq \pi_i \tag{8}$$

$$\Leftrightarrow \qquad ROI_i = \frac{v_i - \pi_i}{\pi_i} = \frac{v_i}{\pi_i} - 1 \geq \frac{v_i}{\frac{v_i}{1+\hat{C}}} - 1 = (1+\hat{C}) - 1 = \hat{C}. \tag{9}$$

"⇐" Follows from inverting the calculations in "⇒".
□
Proof of Corollary 5.3. Follows directly from Proposition 5.2 □

Appendix B - Additional Figures and Tables

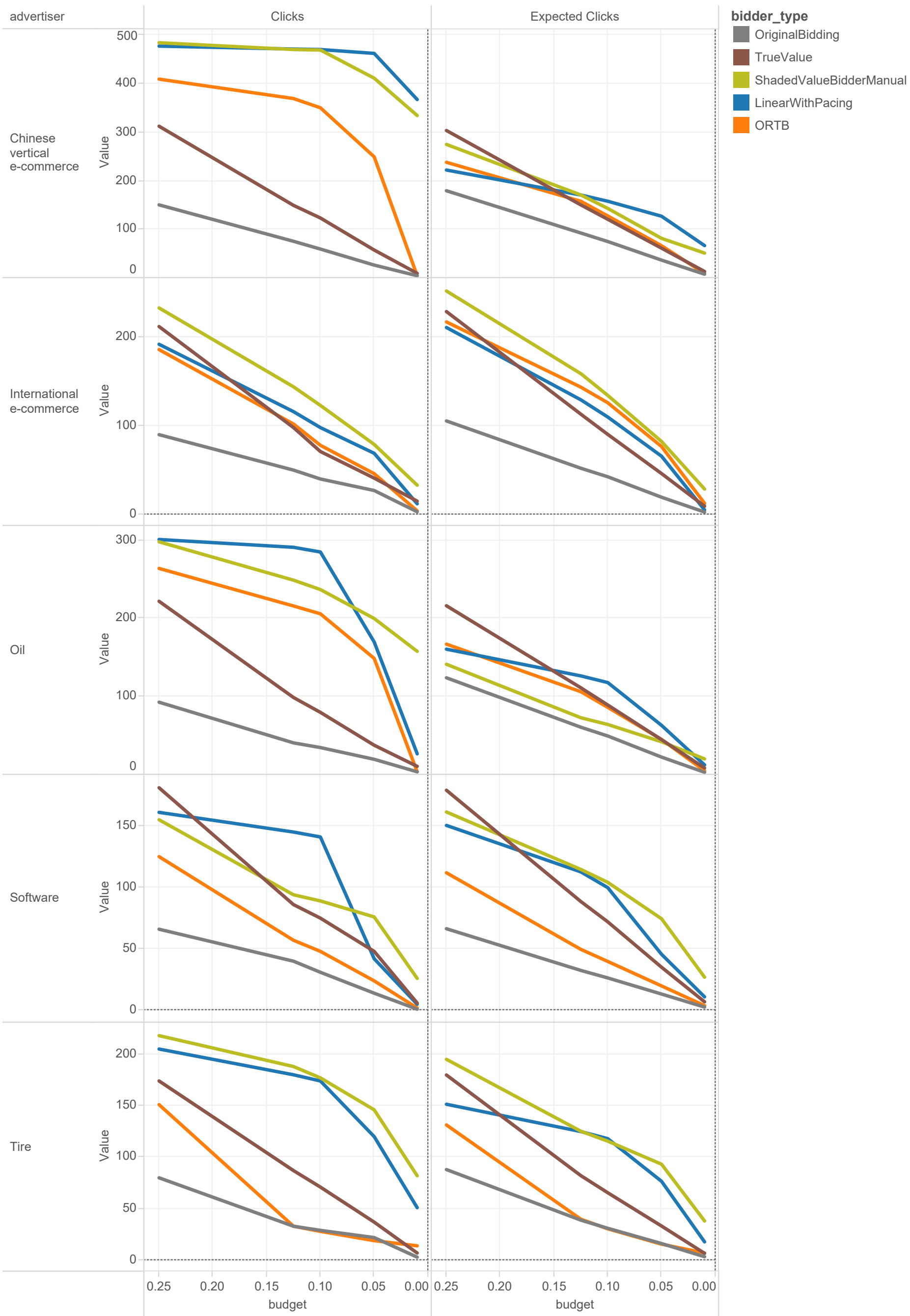

Figure 6: Simulation Results: Achieved clicks (ex-post) and expected clicks (interim) for different capaigns, budget constraints and strategies.

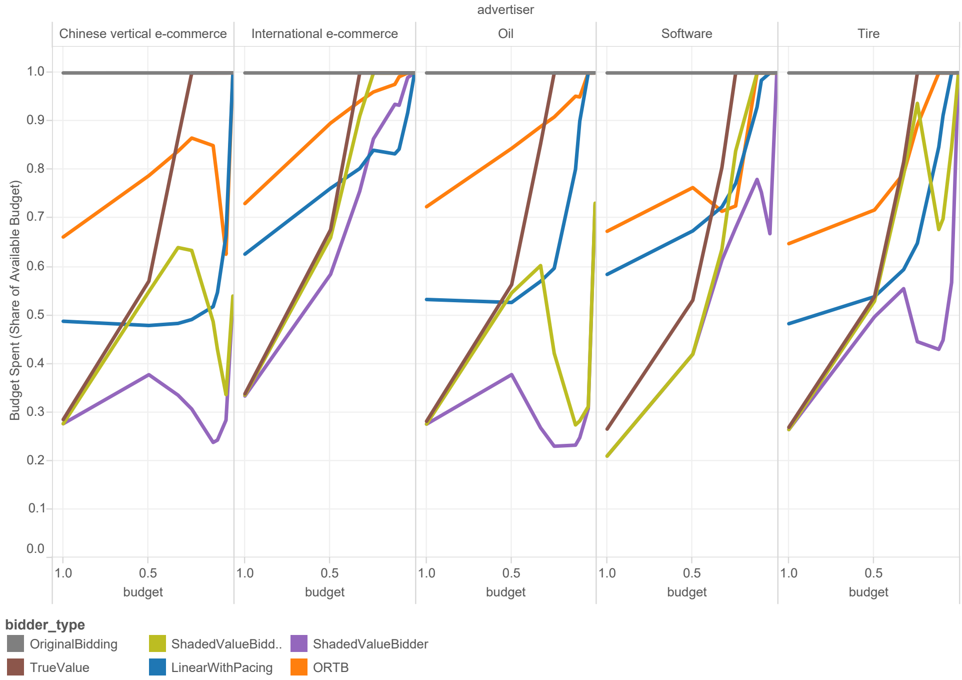

Figure 7: Simulation Results: Percent of available budget spent per bidder and available budget.

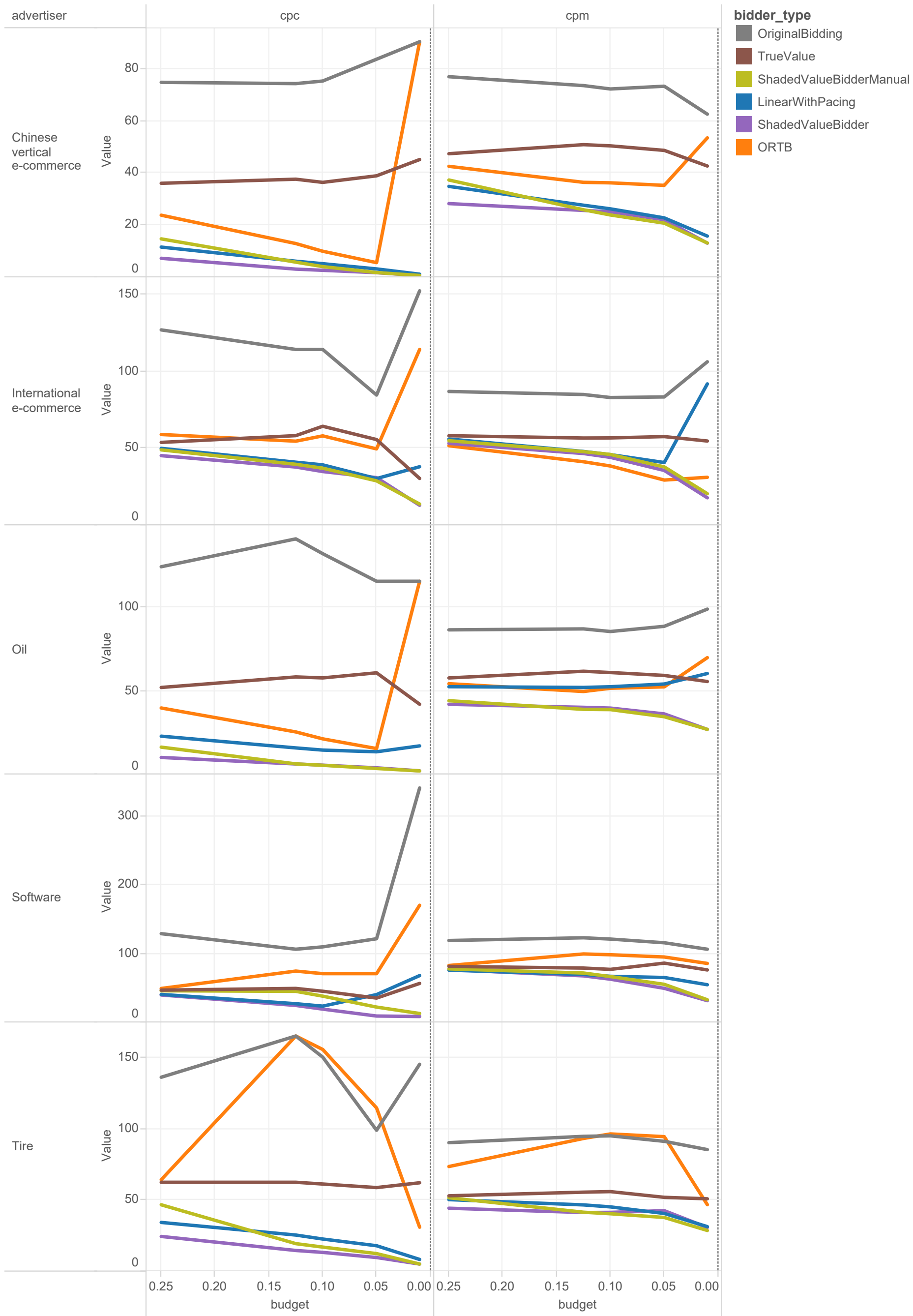

Figure 8: Simulation Results: Campaign Wide Cost per Click and Cost per 1000 impressions for each bidder and available budget.

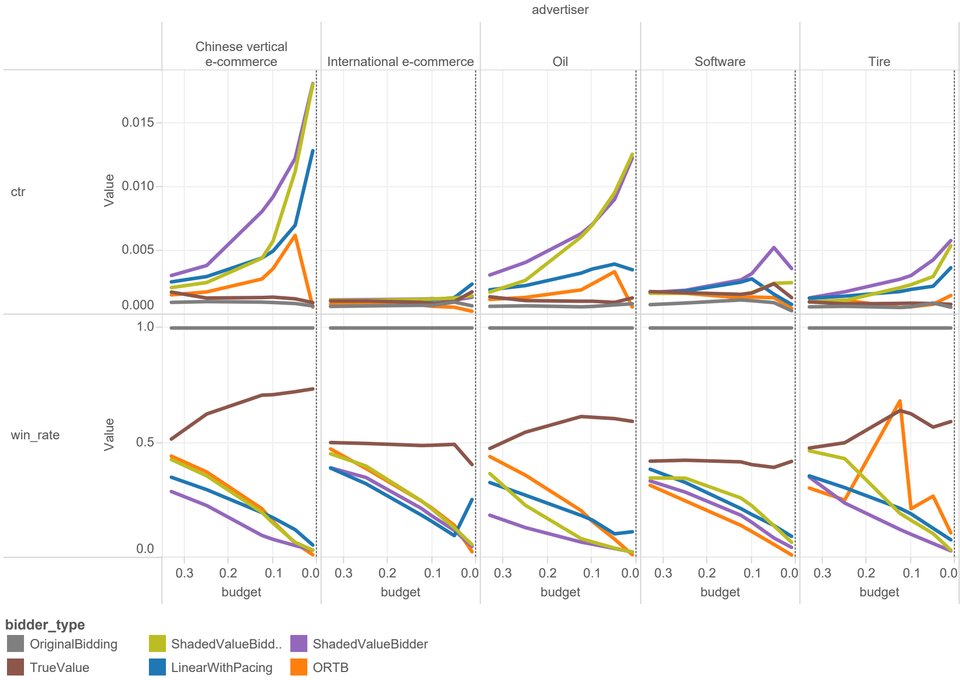

Figure 9: Simulation Results: Campaign wide Click Through Rate and Winning Rate.

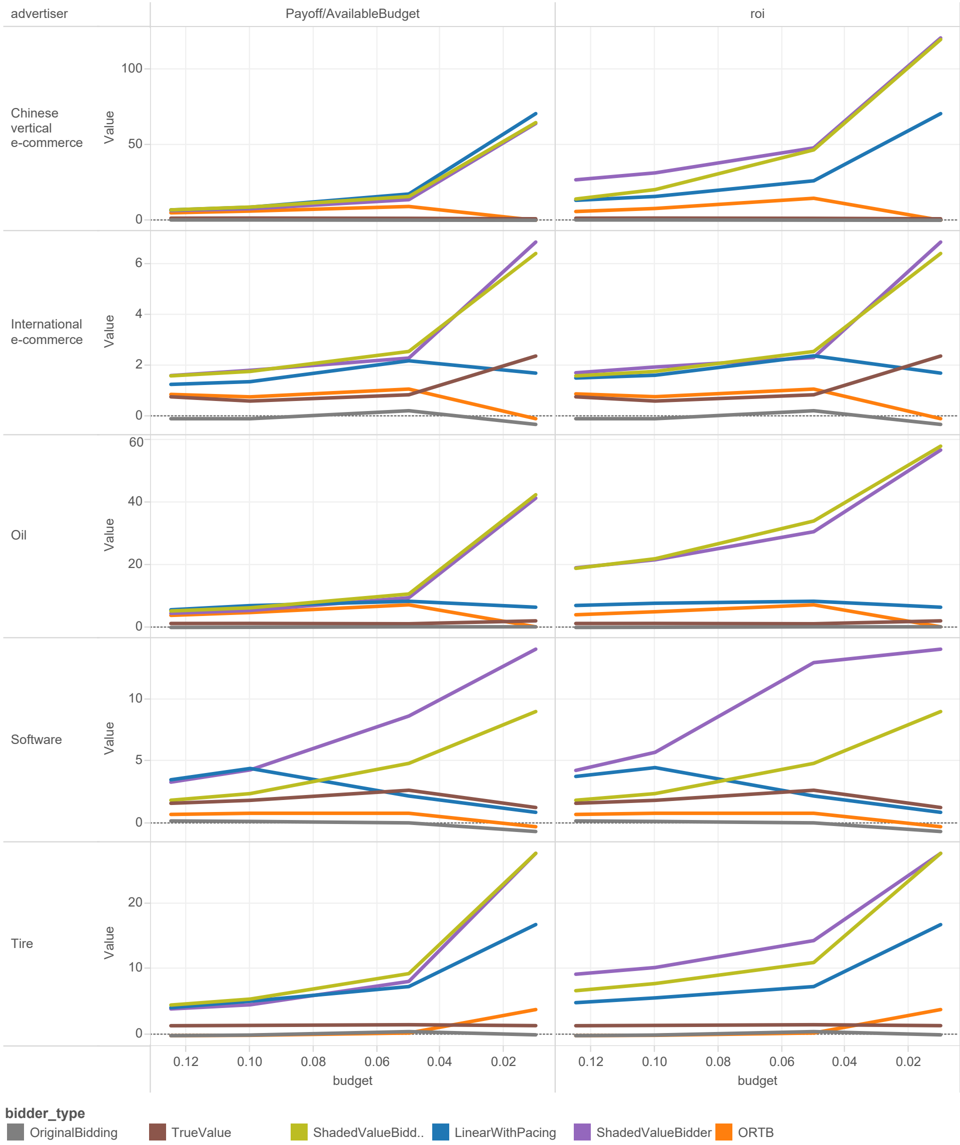

Figure 10: Simulation Results: ROI against the entire available budget (left) and against actual spending (right).

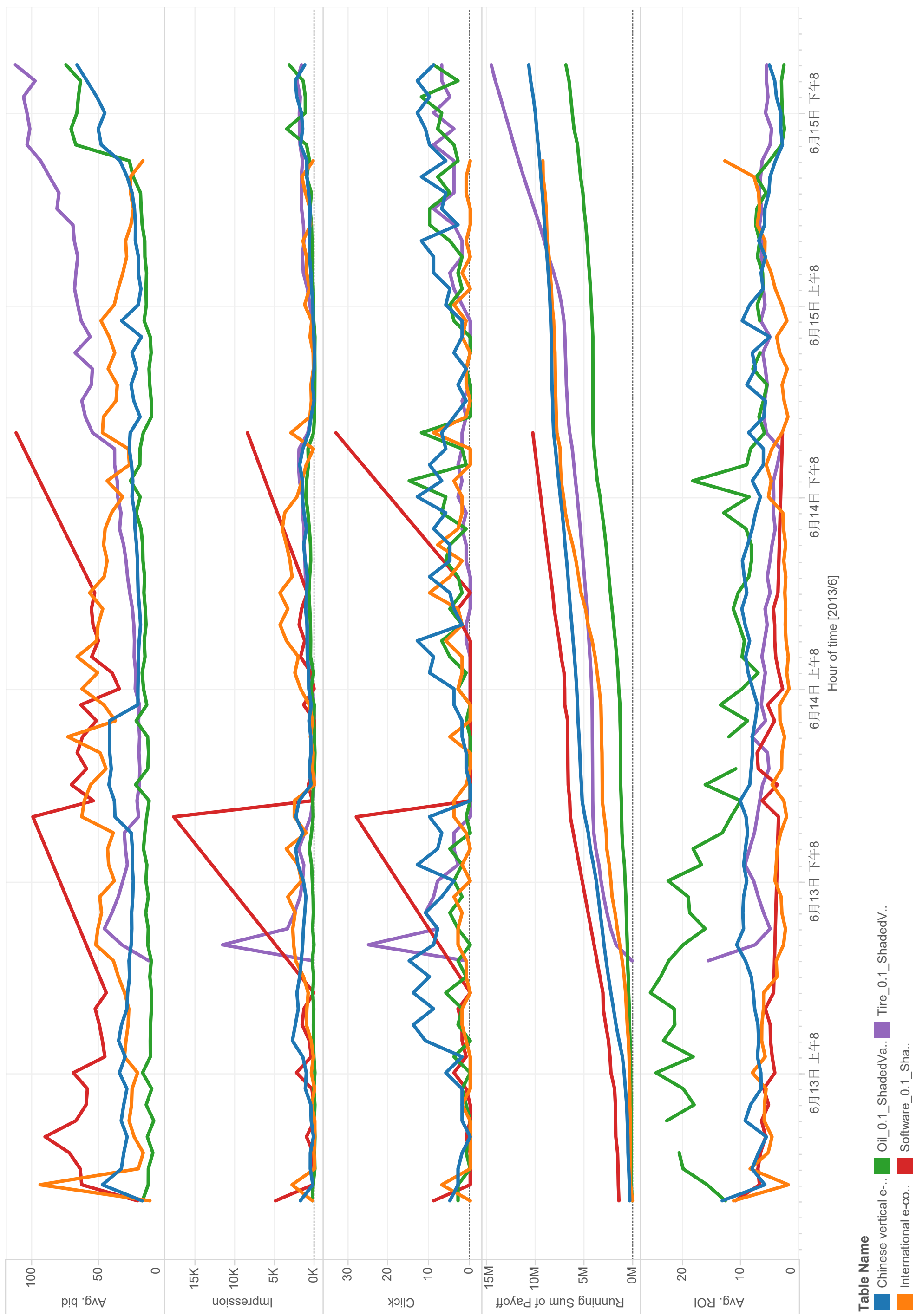

Figure 11: Simulation Results: Progression of 5 example campaigns. (Bid Shading Strategy, 10% of original budget.)

Bibliography

C. Arthur. How low-paid workers at 'click farms' create appearance of online popularity. Aug.2, 2013. URL `https://www.theguardian.com/technology/2013/aug/02/click-farms-appearance-online-popularity`.

S. Athey. Single crossing properties and the existence of pure strategy equilibria in games of incomplete information. *Econometrica*, 69(4):861–889, 2001.

S. R. Balseiro, O. Besbes, and G. Y. Weintraub. Repeated auctions with budgets in ad exchanges: Approximations and design. *Management Science*, 61(4):864–884, 2015.

M. Bichler. *Business Analytics (IN2028)*. Lecture notes, Technische Universität München, 2016.

F. Brandt. *Algorithmic Game Theory (IN2239)*. Lecture notes, Technische Universität München, 2015.

R. H. Byrd, S. Hansen, J. Nocedal, and Y. Singer. A stochastic quasi-newton method for large-scale optimization. *SIAM Journal on Optimization*, 26(2):1008–1031, 2016.

O. Chapelle. Offline evaluation of response prediction in online advertising auctions. In *Proceedings of the 24th International Conference on World Wide Web*, pages 919–922. ACM, 2015.

D. Charles, D. Chakrabarty, M. Chickering, N. R. Devanur, and L. Wang. Budget smoothing for internet ad auctions: a game theoretic approach. In *Proceedings of the fourteenth ACM conference on Electronic commerce*, pages 163–180. ACM, 2013.

Y. Cui, R. Zhang, W. Li, and J. Mao. Bid landscape forecasting in online ad exchange marketplace. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 265–273. ACM, 2011.

B. Dalessandro, C. Perlich, O. Stitelman, and F. Provost. Causally motivated attribution for online advertising. In *Proceedings of the Sixth International Workshop on Data Mining for Online Advertising and Internet Economy*, page 7. ACM, 2012.

T. Du, W. Zhang, and J. Wang. Deep learning over multi-field categorical data. In *European Conference on Information Retrieval*, pages 45–57. Springer, 2016.

A. Ghosh, B. I. Rubinstein, S. Vassilvitskii, and M. Zinkevich. Adaptive bidding for display advertising. In *Proceedings of the 18th international conference on World wide web*, pages 251–260. ACM, 2009.

T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning, 2nd edition*. Springer series in statistics Springer, Berlin, 2009.

OpenRTB API Specification. IAB (Iteractive Advertising Bureau), Advertising Technology Council, 2015. Version 2.3.1.

D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

B. Kitts and B. Leblanc. Optimal bidding on keyword auctions. *Electronic Markets*, 14(3):186–201, 2004.

B. Korte, J. Vygen, B. Korte, and J. Vygen. *Combinatorial optimization*, volume 2. Springer, 2012.

V. Krishna. *Auction theory.* Academic press, 2009.

K. J. Lang, B. Moseley, and S. Vassilvitskii. Handling forecast errors while bidding for display advertising. In *Proceedings of the 21st international conference on World Wide Web*, pages 371–380. ACM, 2012.

X. Li and D. Guan. Programmatic buying bidding strategies with win rate and winning price estimation in real time mobile advertising. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 447–460. Springer, 2014.

H. Liao, L. Peng, Z. Liu, and X. Shen. ipinyou global rtb bidding algorithm competition dataset. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*, pages 1–6. ACM, 2014.

H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, et al. Ad click prediction: a view from the trenches. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1222–1230. ACM, 2013.

V. Mirrokni, S. Muthukrishnan, and U. Nadav. Quasi-proportional mechanisms: Prior-free revenue maximization. In *Latin American Symposium on Theoretical Informatics*, pages 565–576. Springer, 2010.

M. J. Osborne and A. Rubinstein. *A course in game theory.* MIT Press, 1994.

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

L. Shan, L. Lin, C. Sun, and X. Wang. Predicting ad click-through rates via feature-based fully coupled interaction tensor factorization. *Electronic Commerce Research and Applications*, 16:30–42, 2016.

K. Springborn and P. Barford. Impression fraud in on-line advertising via pay-per-view networks. In *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*, pages 211–226, 2013.

Statista. Statista digital market outlook 6.2016. 2016.

L. Stavrogiannis. *Competition between demand-side intermediaries in ad exchanges.* Phd thesis, Diss. University of Southampton, 2014.

O. Stitelman, C. Perlich, B. Dalessandro, R. Hook, T. Raeder, and F. Provost. Using co-visitation networks for classifying non-intentional traffic. 2013.

B. Stone-Gross, R. Stevens, A. Zarras, R. Kemmerer, C. Kruegel, and G. Vigna. Understanding fraudulent activities in online ad exchanges. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 279–294. ACM, 2011.

I. Stratakos. *Click-Through Rate Estimation in Display Advertising.* MSc thesis, forthcoming, Technische Universität München, 2016.

J. Tobin. Estimation of relationships for limited dependent variables. *Econometrica: journal of the Econometric Society*, pages 24–36, 1958.

J. Wang, S. Yuan, and W. Zhang. Real-time bidding based display advertising: Mechanisms and algorithms. In *European Conference on Information Retrieval*, pages 897–901. Springer, 2016.

K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1113–1120. ACM, 2009.

W. C.-H. Wu, M.-Y. Yeh, and M.-S. Chen. Predicting winning price in real time bidding with censored data. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1305–1314. ACM, 2015.

J. Xu, X. Shao, J. Ma, K.-c. Lee, H. Qi, and Q. Lu. Lift-based bidding in ad selection. *arXiv preprint arXiv:1507.04811*, 2015.

S. Yuan, J. Wang, and X. Zhao. Real-time bidding for online advertising: measurement and analysis. In *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising*, page 3. ACM, 2013.

M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. Spark: cluster computing with working sets. *HotCloud*, 10:10–10, 2010.

W. Zhang and J. Wang. Statistical arbitrage mining for display advertising. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015.

W. Zhang, Y. Zhang, B. Gao, Y. Yu, X. Yuan, and T.-Y. Liu. Joint optimization of bid and budget allocation in sponsored search. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1177–1185. ACM, 2012.

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

W. Zhang, S. Yuan, J. Wang, and X. Shen. Real-time bidding benchmarking with ipinyou dataset. *arXiv preprint arXiv:1407.7073*, 2014b.

Y. Zhang, Y. Wei, and J. Ren. Multi-touch attribution in online advertising with survival theory. In *2014 IEEE International Conference on Data Mining*, pages 687–696. IEEE, 2014c.