# Supply Side Optimisation in Online Display Advertising

*Shuai Yuan*

A dissertation submitted in partial fulfillment

of the requirements for the degree of

**Doctor of Philosophy**

of

**University College London**.

Department of Computer Science

University College London

March 6, 2015

I, Shuai Yuan, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

# Abstract

On the Internet there are publishers (the supply side) who provide free contents (e.g., news) and services (e.g., email) to attract users. Publishers get paid by selling ad displaying opportunities (i.e., impressions) to advertisers. Advertisers then sell products to users who are converted by ads. Better supply side revenue allows more free content and services to be created, thus, benefiting the entire online advertising ecosystem. This thesis addresses several optimisation problems for the supply side.

When a publisher creates an ad-supported website, he needs to decide the percentage of ads first. The thesis reports a large-scale empirical study of Internet ad density over past seven years, then presents a model that includes many factors, especially the competition among similar publishers, and gives an optimal dynamic ad density that generates the maximum revenue over time. This study also unveils *the tragedy of the commons* in online advertising where users' attention has been overgrazed which results in a global sub-optimum.

After deciding the ad density, the publisher retrieves ads from various sources, including contracts, ad networks, and ad exchanges. This forms an exploration-exploitation problem when ad sources are typically unknown before trail. This problem is modelled using Partially Observable Markov Decision Process (POMDP), and the exploration efficiency is increased by utilising the correlation of ads. The proposed method reports 23.4% better than the best performing baseline in the real-world data based experiments. Since some ad networks allow (or expect) an input of keywords, the thesis also presents an adaptive keyword extraction system using BM25F algorithm and the multi-armed bandits model. This system has been tested by a domain service provider in crowdsourcing based experiments.

If the publisher selects a Real-Time Bidding (RTB) ad source, he can use reserve price to manipulate auctions for better payoff. This thesis proposes a simplified game model that considers the competition between seller and buyer to be *one-shot* instead

of *repeated* and gives heuristics that can be easily implemented. The model has been evaluated in a production environment and reported 12.3% average increase of revenue. The documentation of a prototype system for reserve price optimisation is also presented in the appendix of the thesis.

# Acknowledgements

I thank my Ph.D. supervisor Dr. Jun Wang from the bottom of my heart for supporting me during these past four years. Jun and I met in January 2010 for the first time when he introduced me to the computational advertising world, which later on became the focus of my study. Besides being an expert in Informational Retrieval and Computational Advertising, he is also the best academic advisor that I can expect. Apparently, without his help I would not be able to finish my Ph.D. and complete this thesis. He has provided so much beyond my research, too, from guidance for planning study, work, and life to suggestions of attending interesting modules and seminars, from encouragement for engaging with social activities to support of getting some industrial and business taste. Especially, He helps me to understand the entrepreneurship and opens my eyes to wider career options.

I am most grateful to Jun for his support of my family commitment. After the birth of my son in June 2012 I struggled to get back to work because my wife and I had little support, and a new-born is too time and energy consuming. Jun allowed me to slow down as well as pushed me to regain the strength. He allowed me to work from home and develop a new routine. He has been so patient and kind that I can never thank him enough for helping me finding a tricky balance; otherwise I could have easily failed both in my family and in career.

I especially want to thank Dr. Kai Xu and Dr. Natasa Milic-Frayling for reviewing my thesis and sitting as examiners of my Ph.D. Viva. During the viva they asked many questions, some of which I had never thought of. They also quickly helped me to find a state of being vigilant and calm that made the viva very productive. After the viva they provided me thorough comments and suggestions. They have been so careful and patient that the comments go from changing a single word to adjusting the structure for the better coherence. These have been very helpful to improve my thesis.

I also want to thank Dr. Shi Zhou, Professor Ingemar Cox, and Dr. Emine Yil-

maz for listening to and discussing with me. I appreciate their helpful advices and comments. Besides, they organised a lot of interesting seminars, talks, and reading groups which are very positive to keep us engaged and think actively. Especially, I want to thank Dr. Shi Zhou and Dr. Kristin Fridgeirsdottir for evaluating my first-year viva, and Professor Ingemar Cox and Dr. Dell Zhang for evaluating my transfer viva. Their comments on research ideas and methodology, experiment design, writing styles, and presentation skills helped me a lot. Their serious attitude and goodwill but sharp questions showed me what true academicians are like.

Computational advertising is a heavily industrial oriented topic; therefore, having support from the industry is important. I have been very lucky to have Joost Zuurbier and Maurice van der Meer who provided me the first opportunity to work on an advertising related project; Peter Mason who generously shared lots of crucial resources and threw many challenges at me; Rong Gu, Sonia Pham, Dr. Christoforos Anagnostopoulos, and Amit Manandhar - it has been so much fun working together with them; Anton Dam who is absolute smart and led me to Data Science in practice; and most importantly, Dr. Sam Seljan and Dr. Catherine Williams who provided me a long term internship and trained me to tackle a complex problem step-by-step in the real-world. Without their help, I could not be able to finish my Ph.D.

I believe a good research group is a key to surviving the Ph.D., and I have been fortunate to have one. Jagadeesh Gorla gave me suggestions and showed care even before I got to UK. He is still doing that now. Tamas Jambor helped me to learn a lot about our group and the university. I can never forget working together with Ahmad Zainal-Abidin, Marc Sloan, Simon Chan, Bowei Chen, Weinan Zhang, and Xiaoxue Zhao. The problems and challenges which we worked on come and go but the friendship and memory of hacking together will never fade.

Ph.D. is a long journey. To me this journey began from the Master study in Beihang University when I had a taste of research for the first time. Although I changed my topic later on, I will forever be thankful to Professor Wei Li, Professor Hui Zhang, Professor Weifeng Lv, Dr. Gang Zhou, Dr. Yi Jin, Dr. ZiLong Chen, Dr. Ning Li, Zhuo Cheng, and so many others who worked together with me. Especially Professor Wei Li, my supervisor of Masters study, encouraged and supported me to pursue the Ph.D. in UK. He will always be a model to me.

I also want to thank Chinese Scholarship Council and Department for Business, Innovation & Skills of UK for funding my Ph.D.. Especially the latter, who generously provided the funding for the fourth year when I had to extend the programme. There are many professional, kind, and considerate people working hard to make sure the funding and scholarship work properly, including Dawn Bailey, Nan Wu, Lin Chen, Huanxian Wang, Jianjiang Kuang, Shanaz Begum, Jing Chang, Yanfeng Shao, Baoling Yin, and Yang Yan. I appreciate their work that virtually made my Ph.D. study possible.

I especially thank my mother and late father who have supported me in all ways that I can or cannot imagine. As a traditional and typical Chinese family, they have given everything to me: health and wealth, hope and wishes, time and life. There must be so much that they have done for me that I do not know. Whenever I think about them, I feel so fortunate to have them as my parents. It is too sad that my father passed away before I can present him my Ph.D. award, to make him happy and proud. It was hard to say goodbye. I wish I had done my study faster and he had given me more time.

As I grow up and have my son, I understand more about love and being loved. My family has given me so much fun and joy, and pacified me when days were grey. Yi Chen, my dear wife, has been with me for nearly 11 years. Apart from my parents, she is possibly the most important person that helps me truly grow up in a complicated world. We have been through so much that leave us lots of wonderful memories and sometimes regrets. During my Ph.D. journey she looks after and encourages me, challenges my research ideas, and listens to my complains. I am so grateful that she is always there whenever I need her. Xingqi Yuan, my bright and restless son, is now more than two years old. In fact, I truly believe the best outcome of the past four years is my son. He has taught me a lot about responsibility and pushing the boundary, trade-off and sacrifice, believing and sticking to the goals. He has been a great distraction, but he is also an important part of my success. My wife and I have learnt a lot from him, too. As a new chapter now begins, together we will live our life to the fullest.

*I dedicate this thesis to*

*my late father Mr. Yunhao Yuan, my mother, my wife, and my son*

*for their constant support and unconditional love.*

*I love you all dearly.*

# Table of Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Online advertising is one of the fastest growing areas in the IT industry. Over the past 11 years, its revenue has increased from $6.0 billion (2002) to $42.8 billion (2013), with a compound annual growth rate of 18.0% [1]. The display-related advertising, which has been set as the background of this thesis, totalled $12.8 billion in 2013, with a 7.0% increase from 2012 [1].

In academia research, online advertising is often referred to as *computational advertising* to emphasise its various core elements such as information retrieval, data mining, machine learning, and microeconomics. It has received notable attention from both industry and academia in recent years [2, 3, 4]. However, it is still a relatively new sub-discipline and requires good field knowledge, such as terminology and business model, to understand its unique challenges. To make the thesis easier to understand, a list of terminology and explanation has been included in Appendix A. To my best knowledge, such a list has not been found in the research literature.

From a research perspective, the online advertising could be generally categorised as *sponsored search* where a user query explicitly describes the intention, and *display advertising* where queries are absent and users' interests have to be inferred from browsing histories. User queries are usually given a very high weight in sponsored search because they directly show what users are looking for. Based on queries and other factors, ads are displayed along with search results [5] on the pages of search engines like Google.

Apart from user queries, there are other factors making sponsored search and display advertising different: 1) In sponsored search, ads are usually limited to some given formats. In general search engines like Google, it is one line text for the title, one line

for the URL, two lines for the description, and possibly some extensions like a phone number [6]. In product search engines like Amazon and Ebay, images and more details of the product could be included in the ads but the formats are still limited;

2) The Cost Per Click (CPC) [7] pricing model and Generalised Second Price auction [8] (where the winner is charged based on quality scores and the second highest bid) are usually the standard in sponsored search;

3) Sponsored search usually has a centralised structure where publishers (i.e., search engines) take almost full control of auctions and ranking, because they have full knowledge of advertisers and their campaigns [8].

On the contrary, display ads have much more flexible formats [9], including various sizes, animation, video clip, sound, and even interactive features. Impressions (an opportunity to display an ad to a user once) in display advertising are mostly sold using the Cost Per Mille (CPM) pricing model. *Mille*, which represents 1000, has been used to make figures easier to read and write because the cost for a single impression is usually low. Display advertising has been widely adopted by online publishers who provide contents (e.g., news, blogs, product comparison) and services (e.g., email, domain names, storage and hosting) to users for free and reimburse the running cost by advertising revenue.

In this thesis, online display advertising has been set as the background and various optimisation problems have been discussed for online publishers (the supply side). These problems include determining the number of ad placements of a website (ad density), pulling ads from different sources (ad selection), and optimising reserve price in Real-Time Bidding (RTB) auctions. These problems have been proposed to follow the typical life cycle of an ad-supported website while recognising the uprising programmatic buying trend (i.e., impressions are sold more automatically and intelligence and optimisation are more needed than ever). Therefore, they are meaningful and helpful to most of the publishers' (the supply side) businesses.

## 1.1 Real-Time Bidding

First of all, the structure and history of online display advertising is illustrated in Figure 1.1. Before the emergence of Real-Time Bidding (RTB) in 2009 (i.e., announcement of support by major ad exchanges [10]), the display advertising market was

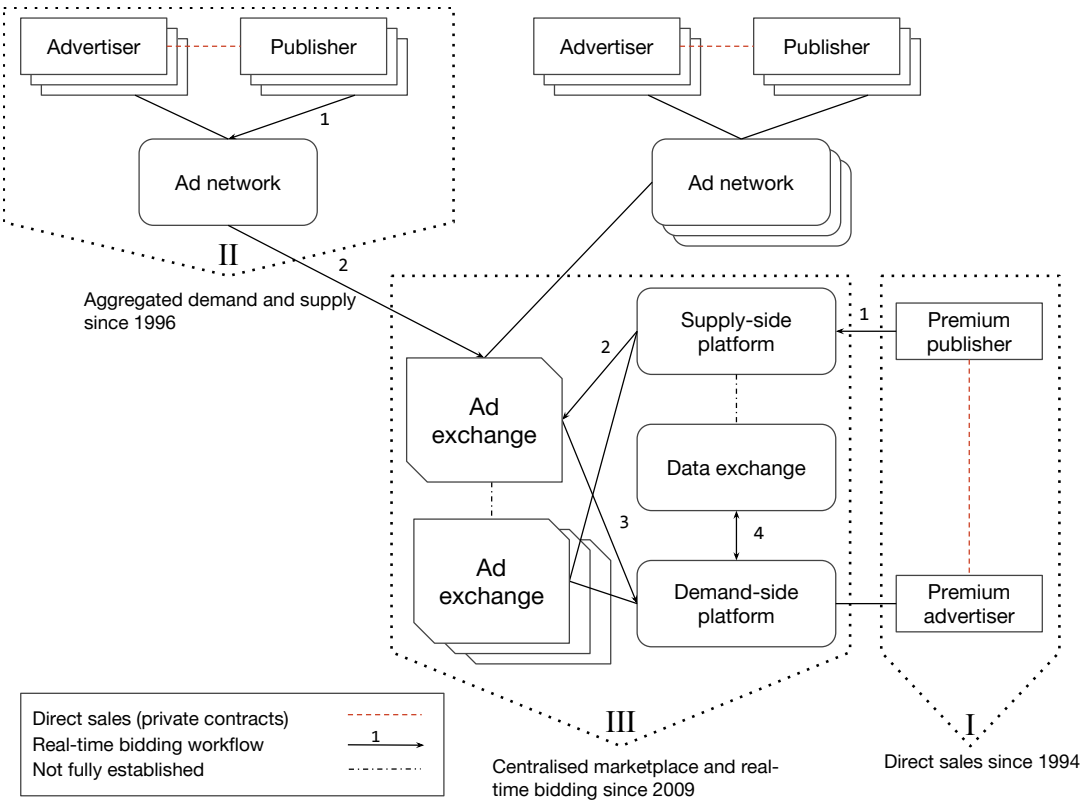


**Figure 1.1:** A brief illustration of the history and structure of online display advertising. Ad networks were created to aggregate advertisers and publishers. Ad exchanges were designed to resolve the unbalance of demand and supply in ad networks. Premium advertisers and publishers now prefer to work with ad exchanges through Demand-Side Platform (DSP) and Supply-Side Platform (SSP) to take the advantage of Real-Time Bidding (RTB). The arrows with a number describe the process of RTB: an impression is created and then passed to ad exchanges; advertisers are contacted through DSPs; advertisers choose to buy third party data optionally. Then following reversed path, bids are returned to the ad exchange, then the SSP or ad network, and winner's ad will be displayed to the user on the publisher's website. The link between SSP and data exchange, as well as those among ad exchanges, are potentially useful however not widely adopted in the current marketplace.

mainly divided by contracts between advertisers and publishers (since 1994) and ad networks which provide demand and supply aggregation (since 1996). The contracts pre-sell impressions as many as possible at a high price and publishers have to negotiate and make deals with advertisers directly, c.f. Block *I* in Figure 1.1. Advertisers usually propose to buy a certain amount of impressions from some given placements, regardless of the identities of users, when and how many times they have seen the ad, and so on. The purchase is totally about impressions and relies on the reputation of the publisher and reported audience profiles [11, 12, 13]. At the other end, the publisher needs to guarantee the delivery of impressions that have been agreed upon, otherwise a

penalty fee would be incurred [14, 15]. The pricing model used in contracts are mostly CPM. With these contracts, advertisers usually have little control over the audience, thus, it is more difficult to deploy goal-driven campaigns (e.g., booking a ticket) than branding ones (e.g., announcing a new product). A *campaign* refers to advertising effort within a period. The goal-driven campaigns are easier to execute when audience data is available (e.g., music-lovers are more likely to buy concert tickets). These contracts are sometimes called guaranteed deliveries [16].

Usually, contracts cannot sell all available impressions because it is almost impossible to predict future traffic volume and publishers tend to be conservative to avoid penalty of under-delivery. To sell those remnant impressions, ad networks are created. With ad networks, publishers register placements (a block on a webpage that is used to show ads) and offer impressions from these placements for sale. Impressions are largely sold using the second price auction mechanism [17] in ad networks. Advertisers (or their delegates) also register with ad networks to participate in auctions. C.f. Block $II$ in Figure 1.1. However, the impressions in ad networks are non-guaranteed, as opposed to premium contracts.

It is then the ad networks' responsibility to understand the webpage and the user, and to select advertisers based on their pre-defined targeting rules. The knowledge of webpage is usually referred to as *contextual information* [18], where ad networks crawl, parse, and extract keywords that summarise the target. Advertisers bid on these keywords which is very similar to sponsored search [19, 20, 21, 22]. A more advanced approach is to learn a model including various features of webpages, which could then be used to compute a relevance score of advertisers' targeting criteria [23, 24, 25]. The knowledge of users is usually referred to as *behaviour information*, where ad networks utilise the browsing history of users to infer interests, as well as geographical locations, local times, etc., for target matching [26, 27, 28].

In ad networks advertisers mostly adopt the Cost Per Click (CPC) or Cost Per Acquisition (CPA) pricing models where they only pay when certain goal is achieved. These choices reduce their risks, so are good for goal-driven campaigns. But since many publishers' inventories are sold using CPM, it is ad networks' responsibility to optimise for getting maximum clicks or conversions. To take the measurement of performance into account ad networks usually use the Generalised Second Price auction

(GSP) [8] which allow applying bid biases (e.g., the quality score [29]) that usually weight the historical Click-Through Rate (CTR) or Conversion Rate (CVR) heavily.

When there were more and more ad networks, a problem led to the birth of ad exchanges: the excessive impressions in some ad networks. It is preferable to have more demand than supply because intense competition leads to higher revenue of both ad networks and publishers. However, when there are plenty of impressions unsold, ad networks try hard to find buyers. Besides, a common practice for advertisers is to register with multiple ad networks to find cheap inventories, or at least to find enough impressions within their budget constraints. They have only found that managing numerous channels is difficult and inefficient (e.g., how to split the budget). Ad exchanges, like Google AdX, Yahoo! Right Media, Microsoft Ad Exchange, AppNexus, etc. have been created to address this problem by connecting hundreds of ad networks together, c.f. Block $III$ in Figure 1.1. Advertisers now have a higher chance to locate enough impressions with preferred targeting rules; publishers may receive higher profit, too, because of more potential bidders.

The most-significant feature introduced by ad exchanges is Real-Time Bidding (RTB), which queries bidding systems (advertisers) for responses for every impression. Along with the query, ad exchanges forward metadata of the webpage and the user. The exposure of this metadata enables advertisers to switch from the inventory-centric to user-centric optimisation. There are also noteworthy attempts to introduce concepts from finance market [30]. These attempts will make the ad exchange more mature and attractive. It is also worth noting that ad exchanges are getting popular with branding campaigns, too, partly because of the possibility of locating cheap impressions on good quality websites.

When advertisers want to take advantage of RTB, they work with third party platform that are usually referred to as Demand Side Platform (DSP). DSPs are delegates of advertisers that answer bidding requests and optimise campaigns at the impression-level. Comparing with ad networks, the advantages of using DSPs are: 1) advertisers do not need to manage their registration with many ad networks; 2) they can optimise at a finer granularity and a higher frequency because of local impression logs instead of aggregated reports from ad networks; 3) DSPs are also more customisable to better suit advertisers' goals.

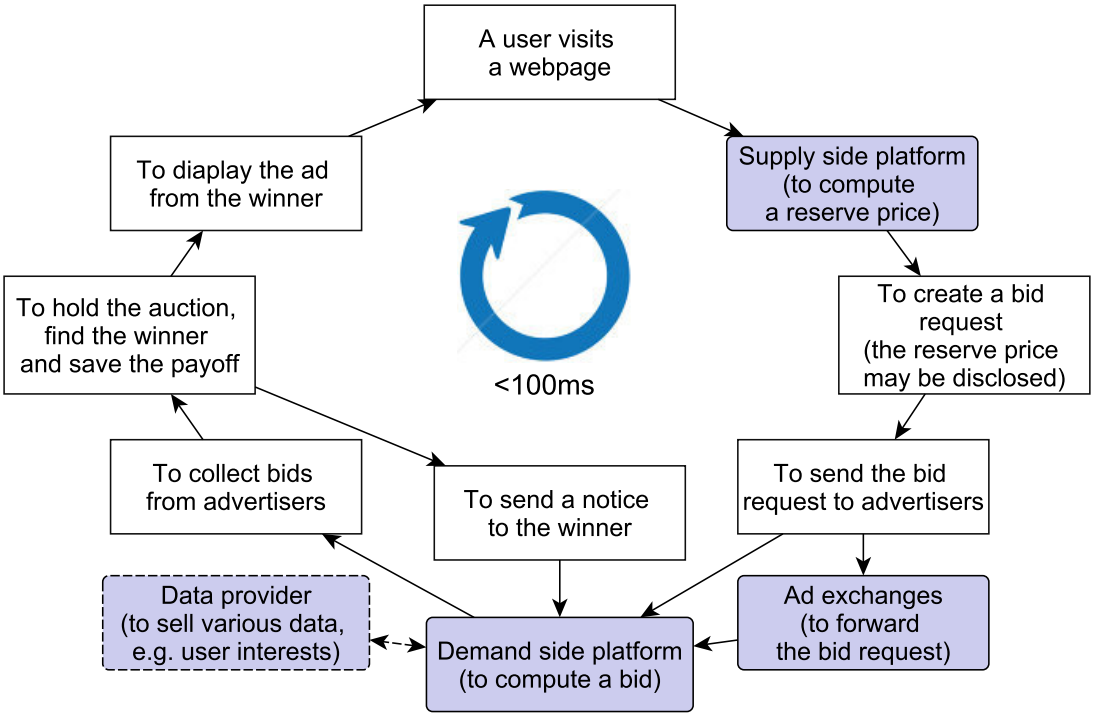


**Figure 1.2:** The summarised process of Real-Time Bidding advertising. Major entities are coloured in blue where processes are transparent. One iteration usually finishes in less than 100 ms to achieve users' satisfaction.

At the other end, Supply Side Platforms (SSPs) have been created to serve publishers. Similarly, SSPs provide a central management console with various tools for publishers' ultimate goal: the yield maximisation. For example, SSPs allow publishers to set reserve prices on a group of impressions. These impressions are usually constraint by geographical location, time of day, or even particular auction winners. Some SSPs also allow publishers to have a preference over buyers through bid bias [31].

In Figure 1.1 arrows with a number describe the simplified process of RTB:

1. An impression is created on publisher's website;
2. The bidding request is sent to ad exchanges through ad network or SSP;
3. The ad exchanges query DSPs for advertisers' bids;
4. The DSP could contact data exchanges for 3rd party user data;
5. The bid is generated and submitted; winner will be selected at ad exchanges, then at SSP. Following the reversed path the winner's ad will be displayed on publisher's website to the specific user.

This process is also summarised in Figure 1.2.

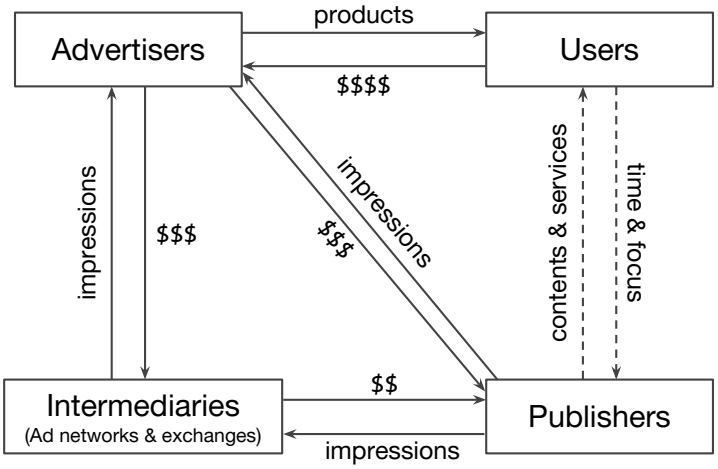


**Figure 1.3:** A simplified illustration of the cash flow in online display advertising. Users spend time and focus on websites to receive free contents and services. Their browse impressions are generated and sold to advertisers with or without intermediaries (ad networks & exchanges). Advertisers buy impressions to show ads of their products and services to users. When users convert (persuaded to buy), advertisers receive revenue.

## 1.2 Supply Side Optimisation

As shown in Figure 1.3, publishers usually run a two-sided business. On one hand, they provide free contents (e.g., news, reviews, and answers) and services (e.g., email, maps, and various online tools) to attract users whose browsing activity in turn generates impressions. Impressions are sold to advertisers directly or via intermediaries (ad networks & exchanges). Publishers use ad revenue to compensate the running cost. Therefore, optimisations for publishers are not only crucial to their business, but also highly related to the health and diversity of the whole Internet.

This thesis addresses several supply side optimisation problems within the context of RTB, where advertisers and publishers are both given the maximum freedom at the finest granularity: the impression-level. Specifically, these problems are discussed:

1. How many ads should be displayed on the website?
2. Which ad sources should be used?
3. What is the optimal reserve price in RTB auctions?

By answering these questions, several important parts of supply side optimisation are covered. An overview and connections are illustrated in Figure 1.4.

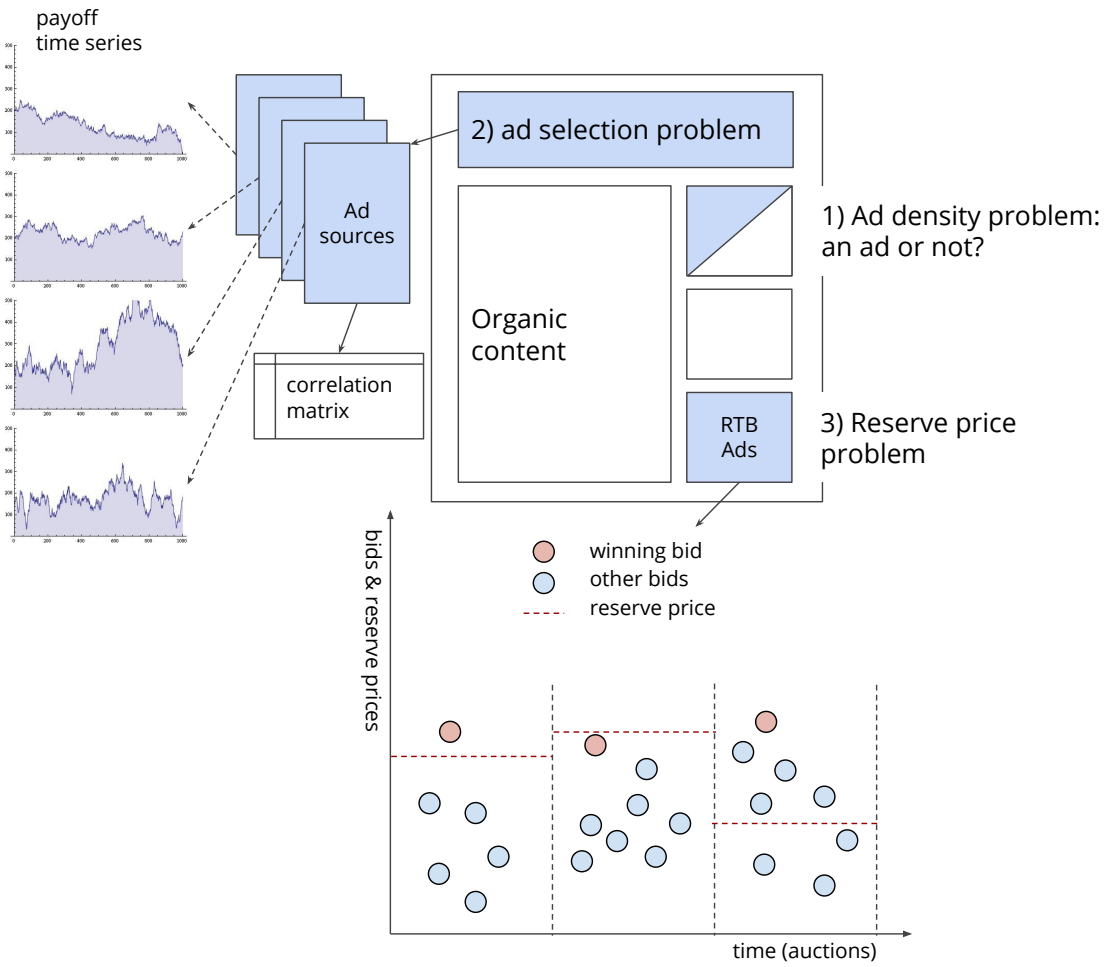


**Figure 1.4:** An overview of optimisation problems in this thesis and their connections.

### 1.2.1 Ad Density

The first question could be interpreted as the *ad density* problem. When a publisher creates a website, he needs to decide how many ads (i.e., the ad density $\rho$) to put on at each time step to generate the maximum payoff $R$, usually under the assumption of users' preference over ads. Many other factors also need to be considered, including: the unit payoff $c$, running cost $k$, website traffic $x$, and competition with other publishers providing similar contents or services.

Chapter 3 studies the ad density problem both empirically and theoretically and shows that ads on the Internet are overcrowded and users' attention has been overgrazed. Economically this is described as *the tragedy of the commons* [32] due to the publishers' competition. The chapter presents the first large-scale empirical study of display ad density evolution over the last seven years. By web crawling and linking together datasets from multiple sources (unit payoff, websites, users' interest segments,

daily page-views, archival and current webpages, and ad slots), for the first time, the general increasing trend of display ad density for most websites has been unfolded; various factors that affect the ad density decisions have been unveiled, too. The chapter then presents a mathematical model with carefully designed objective and state transition functions to optimise ad density. The model is constructed based on findings from the empirical study and solved by using game theory [33] and control theory [34]. The solution from the model nicely matches findings from the empirical study.

### 1.2.2 Ad Selection

The second question could be interpreted as the *ad selection* problem. After determining the ad density and setting up the website template, the publisher needs to decide where to get ads from. Often there are plenty of choices including ad networks, ad exchanges, and direct contracts. Chapter 4 proposes a Partially Observable Markov Decision Process (POMDP) based model [35] to find the best selection strategy $\pi$. Bayes' theorem is used to update the expected payoff $X$ of different ads and utilise their correlations $\Sigma$, so that the expected payoff of any candidates could be updated even if it is not selected. The objective is to maximise the cumulative payoff within $T$.

After deriving belief update equations both exact solution using value iteration and approximate solution using multi-armed bandits are presented. The experiments show that with correlated belief update the exploration speed increases, and the ad revenue is uplifted (23.4% better than the best performing baselines) using real-world data.

### 1.2.3 Reserve Price

The third question could be interpreted as the *reserve price optimisation* problem. The second price auction [17] is widely used in RTB auctions where the winner pays the second highest bid, or the minimum value (usually one cent) if he is the only bidder. When sending bid requests to selected ad exchanges, publishers are allowed to set a reserve price (also known as the hard floor or the floor price) in these auctions as the minimum value [36]. Therefore, it could potentially increase the revenue if it is between the first and the second highest bids, when the winner has to pay the reserve price instead of the second highest bid.

For each auction, suppose there are bids $b_1 \geq b_2 \geq \cdots \geq b_K$. With a reserve price $\alpha$, the preferable case is $b_1 \geq \alpha \geq b_2$ where the winner has to pay $\alpha - b_2$ more than a

second price auction without a reserve. However, the publisher has to take the risk of auction failure that gives zero revenue when the reserve price is above the highest bid, i.e., $\alpha > b_1$.

Reserve price optimisation has been used for sponsored search and been well studied in that context [37, 38]. However, comparing with sponsored search and contextual advertising, this problem in the RTB context is less understood yet more critical for publishers because: 1) Bidders have to submit a bid for each impression, which mostly is associated with user data that is subject to change over time. This, coupled with practical constraints such as the budget, campaigns' life time, etc. makes the theoretical result from optimal auction theory not necessarily applicable and further empirical study is required to confirm its optimality from the real-world system; 2) In RTB, an advertiser is facing nearly unlimited supply and the auction is almost done in last second, which encourages frequent and dynamic budget reallocation to avoid high-cost ad placements. This could imply the loss of bid volume over time if an incorrect reserve price is used.

Chapter 5 reports the first empirical study and live test of the reserve price optimisation problem in an operational environment. The results suggest that the proposed game theory based ONESHOT algorithm performed the best and the superiority (12.3% on average) is significant in most cases.

The study of this thesis is highly relevant to the online advertising industry. With sponsorship from UCL Advances, the implementation of the reserve price optimisation system is included in Appendix B by providing infrastructure requirement, code samples, and documentation.

### 1.2.4 Contributions

The contributions of the thesis are three-fold. First, it solves three important problems for the supply side in online display advertising. For ad density, the thesis develops, solves, and analyses the model under different circumstances by applying the optimal control theory and game theory. The consideration of publishers' competition is novel and has not been addressed before. For ad selection, a POMDP model and correlated belief update are used to find the optimal choice with accelerated exploration. This is the first time that POMDP model and correlated belief update have been used in such

context. For reserve price, effective heuristics have been derived from a simplified game model between the publisher and the auction winner. Such innovative modelling and discussion of limitation of optimal auction theory in RTB have not been seen in the research literature. Most importantly, by solving these problems, the presented work increases the publishers' revenue which benefits the entire ad ecosystem.

Second, there are several empirical studies in the thesis, especially the ad density evolution for the Internet in recent years (c.f. Section 3.2), and characteristics of RTB auctions (c.f. Section 5.2. It is expected that players in online advertising ecosystem recognise these findings and adjust their activities accordingly, and maintain a healthier online advertising ecosystem.

Last but not least, the thesis describes the implementation of a reserve price optimisation prototype system in detail. This helps the interested publishers or technology providers to get hands on this optimisation quickly.

## 1.3 Thesis Structure

The thesis is structured as follows.

A survey of related works is provided in Chapter 2. It discusses the general research problems of computational advertising including the demand side bidding optimisation. The detailed discussion of related works of each problems is split into individual chapter.

The ad density optimisation is studied in Chapter 3. First a large-scale empirical study on Internet ad density evolution in recent years is presented, which unveils many interesting findings including the relationship of ad density with website categories, unit payoff price, web traffic, and competition. Based on these relationships a dynamic model is proposed and solved for the optimal ad density in the planning horizon using control theory and game theory. By comparing cases with and without competition, the proposed model shows *the tragedy of the commons* in online display advertising.

The ad selection problem is studied in Chapter 4. A POMDP model is built for dynamic selections and solved by using value iteration and multi-armed bandits models. Experiments based on real-world payoff time series data confirm the benefits of correlated belief update. In this chapter, a toy example is provided to help understanding the belief update process; an adaptive keyword extraction system is provided to help

manipulating the ad source by supplying different keywords when possible.

Followed by a large-scale empirical study of RTB auctions, the reserve price optimisation problem is studied in Chapter 5. Based on the recognition of bidders' lifetime and bidding pattern, heuristics are derived from a simplified game model and approve its effectiveness in online experiments.

At last, the thesis gives conclusions and discusses future work. Additionally, a list of terminology of computational advertising is given in Appendix A. The description of a reserve price optimisation prototype system is given in Appendix B.

## Chapter 2

# Related Works

In each chapter the related works have been discussed in details. This chapter summarised the research works of computational advertising and other aspects of the supply side optimisation problem. Around the topic of supply side revenue optimisation there is good research effort in recent years. For example, there is rich literature on ads scheduling especially when guaranteed and non-guaranteed deliveries are both available [39, 40, 14, 41, 42]; choosing the pricing models [43, 44, 45]; and the reserve price optimisation [38, 37, 46, 47]. The perishable feature of impressions has been addressed in [48]. The paper suggests showing publishers' own ads when impressions cannot be sold. The authors of [49, 30, 50] propose to sell impressions in advance for better planning.

Besides, user modelling has been an important topic in information retrieval and computational advertising research. In [51] the author study the sponsored items in recommendation systems and conclude that the decision will be largely based on users' preference. When there are competing publishers they will try to make the sponsorship more exclusive. In [52] a users' preference model over online advertising is studied and authors break the causes of avoidance of ads into three parts: perceived goal impediment, perceived ad clutter, and prior negative experience. The work of Chapter 3 recognises the avoidance or aversion of display ads; however, following the common industrial practice, the proposed model do not control the content of ads; thus, the change of users' preference over different ads is not modelled. Interestingly, in [53] the authors report that ads that match both website content and are obtrusive do worse at increasing purchase intent than ads that do only one or the other.

## 2.1 Inventory Management

Inventory management is a well-studied topic for the supply side optimisation. An interesting aspect of ad inventory management is to find the optimal pricing models. The CPM and CPC models are compared in [54]. In [44], the study concludes that a combination of these two pricing models could be the most optimal under particular constraint. The same choice problem is studied in [43]. However, the author concludes that the optimal choice for a price taking publisher is either CPM or CPC but not a combination. Similar conclusion is reported in [45], in which the competition between two models is discussed but limited to a static setting: a single time step of planning horizon. In Chapter 4, the goal is similar to those papers since the target is to find optimal ad sources. However, the proposed model is focused on Real-Time Bidding setting which provides more flexibility (the publisher could choose a source for every impression); the correlation of ads is also used to accelerate the discovery process. For instance, different from [43] no assumption is made on market elasticity; neither the publisher is constraint to a contract-first setup [40].

Another aspect of inventory management is ad scheduling with the constraint of geometrical features of website, uncertainty of advertisers, and available ad inventory (impressions) [39, 14, 55]. If the publisher fails to deliver promised impressions a good-will penalty would be incurred. In [55] the ad scheduling problem is modelled in video games and a dynamic algorithm is developed. When considering pricing models publishers need to understand the webpage content and user preferences to achieve a better scheduling. Because displaying ads may not earn the publisher anything if ads are not relevant or users are not interested, therefore, no clicks. Traditionally, optimising matching between webpages and ads belongs to the field of contextual advertising research [24, 19]. In [24] a system utilising both semantic and syntactic features is proposed to address the problem. Similarly the correlation of user-ad (behaviour targeting) is studied in [56, 25], focusing on improving advertising effectiveness by modelling attitude and feedback from users. The work of Chapter 4 can be consider as an extension of the above works. The contextual matching can be used as a prior belief in our model and it can be updated and verified using our update formulas sequentially.

In recent years, programmatic guarantee (sometimes called programmatic direct/reserve/premium) has gained much attention [57]. It is essentially an allocation and

pricing engine for publishers that brings the automation into the selling of guaranteed contracts. A weighted assignment algorithm is studied in [41] where the publisher is assumed to be interested not only in fulfilling the guaranteed contracts but also in delivering well-targeted impressions to advertisers (due to the *free disposal effects*). This work actually investigates a matching problem of guaranteed impressions. The allocation of impressions between guaranteed and non-guaranteed channels is discussed in [58] where the publisher may act as a bidder who bids for guaranteed contracts. Therefore, RTB bids have to be high enough otherwise the impression would be allocated to guaranteed contracts. Stochastic control techniques are used by [59] to solve the similar problem. For a given price of impression, the publisher can quickly decide whether to send it to ad exchanges or to assign it to an advertiser with a reservation. The total revenue from ad exchanges and reservations are then maximised. In [14], the publisher can dynamically select which guaranteed buy requests to accept and then delivers the guaranteed impressions. The model considers the uncertainty in buy requests and website traffic in the revenue maximisation. To price guaranteed contracts containing the bulk of impressions, the work of [16] discusses two algorithms based on the users' visits to the webpages. The revenue is then maximised for the given demand.

Cancellation has been discussed in [60, 49] where the publisher can cancel the guaranteed contracts with advertisers by paying a penalty. Publisher thus has a lot of flexibility in the selling but there exist speculators in the game who pursue the penalty only. Such mechanisms are similar to the over-selling booking systems of flight tickets [61]. Ad option contracts discussed by [62, 63, 30], on the other hand, are advertiser-focused guaranteed contracts. They are a kind of special guaranteed contracts because advertisers are allowed to pay a certain premium fees in advance to exchange for the priority buying right (but not the obligation) of inventories in the future. In [63, 30], advertisers can choose a combined payment scheme in online advertising. For example, paying by CPC in display advertising or paying by CPM in sponsored search. The introduction of ad options encourages the exploration of statistical arbitrage. However, it is left to the future works.

## 2.2 Bidding Optimisation

It is equally important to follow the demand side research effort because the advertising market is naturally made of buyers and sellers and they compete. The bidding optimisation has been one of the most important demand side topics in the computational advertising research [64, 65, 66, 58, 67, 68], especially in the sponsored search context [8, 69, 70, 71]. Because of its complexity, most of the works in the research literature only focus on a part of the problem. For example, much attention has been paid on click-through rate (CTR) prediction in the sponsored search context [72, 73, 74, 75], where algorithms predict the the probability of click events given the ad and the context. This prediction is than used in ranking bids in the Generalised Second Price auctions (GSP). Due to the nature of sponsored search, keywords (queries) are usually the factor considered firstly and put heavy weight on. In [72] keywords are clustered by relevancy to make confident predictions based on historical data. The authors of [76] use logistic regression and various features of ads to predict CTR, and discuss its variations to predict term-CTR and related-term-CTR. In [77] decision rule based regression model is learned to solve the prediction problem. Some works also consider behavioural targeting to improve the performance. In [73] uses' intentions inferred from query features and search engine result pages are introduced in the prediction model. An important finding is that number of ads and rank of ads affect the CTR, too. Later in [78] users' demographic-based features (age, gender, etc.) and user-specific features (interaction with ads) are introduced in the prediction model. These works have unveiled the importance of audience data, however, these features are not comparable with the complexity of user segments in RTB.

Considering CTR (conversion rate in RTB) alone could result in less effective bidding algorithms. In [67] the authors propose a bidding function linear to the predicted conversion probability of the inventory. That is, if showing an ad on a particular piece of inventory doubles the probability of conversion over some random inventory, the bid price should be twice the average bid price as well. An advantage of such bidding function is not requiring knowledge of campaign's base conversion rate or the private value distribution. The limitation of such linear bidding strategy has been discussed and validated in [79].

Bid landscape forecasting is anther important aspect of bidding optimisation. Here

algorithms are expected to predict the number of impressions if bidding X on targeting combination Y. Again in sponsored search, the targeting rule primarily focuses on predetermined keywords. In [80] both a linear regression model and an iterative approximation are represented. It has also been discussed in the display advertising context in [81] using divide-and-conquer approach: firstly bid distribution is estimated from samples using gradient boosting decision trees; then campaign-level bid distribution is constructed using a mixture of log-normal model from the sample-level distributions.

Other factors affecting a bidding algorithm include budget constraint, campaign's live period (which effectively translates to the planning horizon), and peer competition. In [82, 83] authors propose to calculate a bid allocation plan in planning horizon (could be infinite). It is especially interesting since it breaks down the problem into *click prediction* and *market prediction* (the estimated position if bidding X at hour Y). Samples to train the model need to be purchased on the market using real money.

In fact, one can easily argue that since search engines design bidding algorithms as well as host auctions, the objective function could be diverted to optimising the overall revenue for the search engine [84, 85, 86, 87, 88], instead of performance of each individual advertiser's campaigns.

In the display advertising context, bidding algorithms are not constrained by predetermined keywords and usually use CPM pricing (ranking) models. In [58] the authors propose an algorithm that learn winning bids distribution from full or partial information of auctions. The algorithm then makes bidding decisions to achieve the best delivery (number of impressions) within the budget constraint.

# Chapter 3

# Ad Density Optimisation

Publishers (including owners of websites, apps, blogs, videos, etc.) are an important part of online advertising ecosystem. Many of them choose to provide content and services for free and seek compensation by showing display ads or commercially intended contents to users. Display advertising, as one of the primary sources of revenue for online publishers, is becoming more and more popular in recent years with the growth of programmatic advertising, especially the Real-Time Bidding (RTB) [1] because it enables publishers to sell their inventories together with audience data and as such, largely increase the value of the impressions.

However, it is common that users complain about seeing too many ads that are distracting or even annoying [89]. To my best knowledge, there is no existing research work that measures the number of ads online, their evolution over years, or if the number of ads is reasonable. This motivates the study both empirically and theoretically. This chapter presents the first-of-its-kind large-scale empirical study of display ad density evolution over recent years. To understand the findings from this study, a model is presented that optimises ad density for a given website, or several competing websites. Variations of the model and their implications are also discussed later.

In this thesis, ad density is defined to be the percentage of ads on a webpage. More precisely, it is the area of all ad slots divided by the area of the whole webpage. Note that pop-ups are ignored as they are getting less popular and blocked by default. For example, in Figure 3.1 ads are marked by red rectangles; the ad density of the same webpage has increased rapidly from less than 1.0% in 2010 to around 18.0% in 2013. It makes more sense using the percentage instead of the number of ad slots or their aggregated area (these are also reported in the chapter) since long webpages usually

contain more information thus deserve more ad slots. This chapter uses the concept *first screen ads*, too. These ads are displayed by default (without the need of scrolling) when a modern browser opens the webpage. To be more precise, an ad is considered to be in the first screen if the vertical position of its lower border is within 1000 pixels from the top of the webpage. The width of the browser has been set to 1366 pixels, the maximum of the test machine, to accommodate the responsive design (automatic width resizing according to the browser window) of most webpages, as shown in Table 3.1.

Based on the data collected over the last seven years, an empirical study is conducted in this chapter. In this study, the ad density is found to have a positive relation with the competition, but a negative relation with the unit payoff. By analysing time series of ad density and daily page-views (PVs) over recent years, it is shown that the ad density has been generally increasing, however the speed of increment is affected by the change rate of PVs. It is worth pointing out that the empirical study is user-interest-centric instead of content-centric. That is, websites are aggregated based on users' interest segments (tags); then these tags are taken as keys in analysis. This conforms with the modern RTB practice where segments are considered determinative [90].

Mathematically, finding an optimal ad density is a non-trivial task. The game theoretical model proposed in the chapter shows that it could be affected by many factors: the unit payoff of displaying ads, the maintenance cost, daily PVs and natural growth, users' preference, ad density decisions of other similar thus competing publishers, etc. The last factor is mostly overlooked in practice or existing research literature [91].

The unit payoff of display ads usually uses the CPM model [92]. There are other pricing models available, for example, CPC and Cost Per Acquisition (CPA) [54]. However, CPM is mostly adopted and becoming the de facto standard in the popular Real-Time Bidding. CPM is used throughout the chapter to remove the challenge of click through rate or conversion rate prediction.

In general, the ad density optimisation could be modelled as a control problem and solved by applying the Pontryagin's maximum principle [34]. The number of impressions is considered the *states* and the ad density the *control*. Special attention is paid to multiple publishers' case where they provide similar content (stock prices, news, etc.) and compete for users, who is assumed to visit one but only one of websites to find information on a regular basis. However, the user can also choose to leave all of these

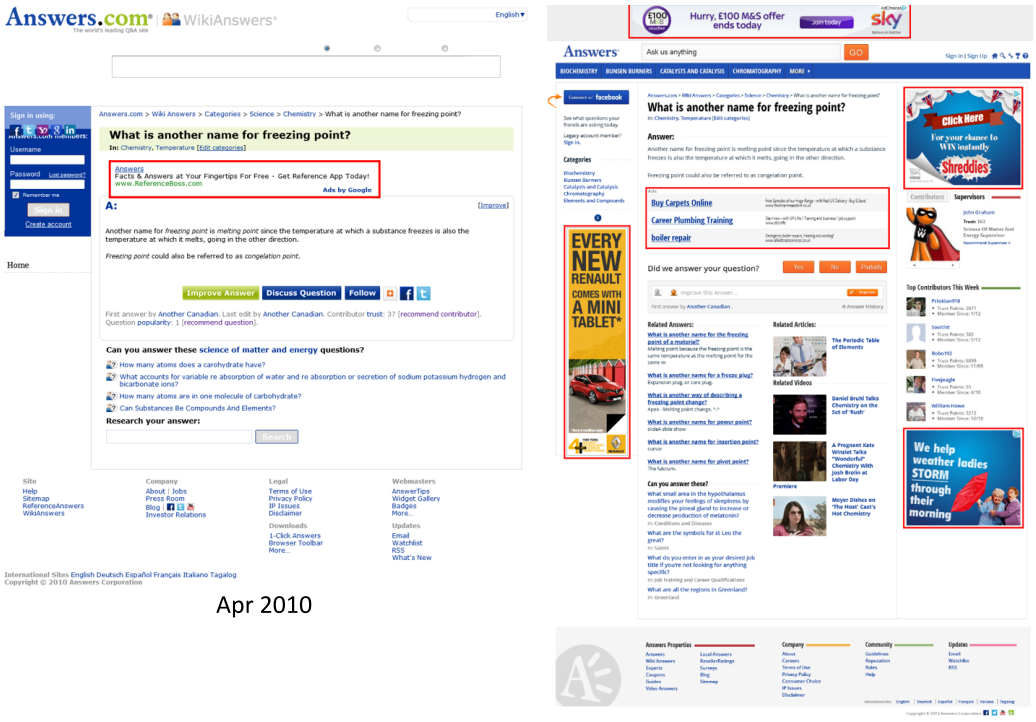


**Figure 3.1:** The ad density refers to the percentage of ads of a website. The figure shows the rapid increase of ad density (from less than 1.0% to around 18.0% in 3 years, measured by image's pixels) for a top-ranking website.

websites if there are too many ads [93]. It is assumed further that users' decision on which website to visit depends on the ad density only. This defines the state transition function of the system.

By comparing cases with or without competition, the analysis shows that the social welfare would be harmed if there are multiple publishers. Given the exact same parameter setting, the competing-case always ends up with fewer impressions and higher ad density. Thus, it is globally sub-optimal when each publisher optimises towards one's self-interest. This result corresponds to a well-known economics phenomenon: the *tragedy of the commons* [32] by considering users to be common resources. This calls for more sensible control of ad density that could return a healthier online advertising environment in the long term.

## 3.1 Related Works

The ad density problem is probably the first question for publishers who want to run an ad-supported website. In [91] the ad density problem has been studied for the first time in displaying advertising. The authors use a similar model as the thesis to capture

the relationship of revenue, traffic, and users' preferences. Then the authors evaluated the market capitalisation of websites from a finance perspective. No competition has been considered in the paper. Another piece of relevant work is [94] where a user-centric approach has been adopted to study the display ads, although little attention has been paid to the evolvement of ads over years. The importance of optimising a whole webpage instead of placements is addressed in [95].

In [96] the authors analyse traffic and revenue for big portal websites and conclude that website attract more viewers would generate more revenue, which in turn enables them to create more contents and attract even more visitors. This empirical result validates the proposed model where the websites' competition is explicitly considered.

Similar to finding a balance between content and ads, in [97, 98] the authors study the optimal combination of subscription fees and online advertising revenue. The control theory is used to optimise the subscription fee and advertising level dynamically at the same time. The model suggests to reduce subscription fees and use a low ad density in the beginning. A related work is [99] where the authors develop an economic model to examine the profitability of flat-fee banner ads and contextual advertising in both a monopolistic market and duopolistic market. Besides, the authors look at the market efficiency (social welfare) the suggests a monopoly is more efficient when the CTR of flat-fee banner ads is low. In [100] a similar analysis is performed to find the optimal strategy of selling software (flat subscription fee or advertising-supported). In [101] the authors study the optimal bias level for single and multiple gatekeepers (search engines) when they are identical or different. These concerns over competition in the marketplace inspire the development of this chapter. The optimal ad density problem has also raised attention in the sponsored search field. These studies mainly focus on fighting for users' attentions over organic search results and sponsored ads [102, 103, 104, 105, 5]. For example, in [5] the bidding strategy of an advertiser is studied when he needs to compete against the organic search results as well as sponsored ads from competitors. In [103] a bandit model is used to generate the most relevant information to be presented to limited attention users, rather than to show ads which causes distraction. The work of [102] studies how users interact with search result pages using eye-tracking techniques. Similarly, another eye-tracking based study is reported in [105] which concludes that the amount of visual attention that people devote to ads depends on their

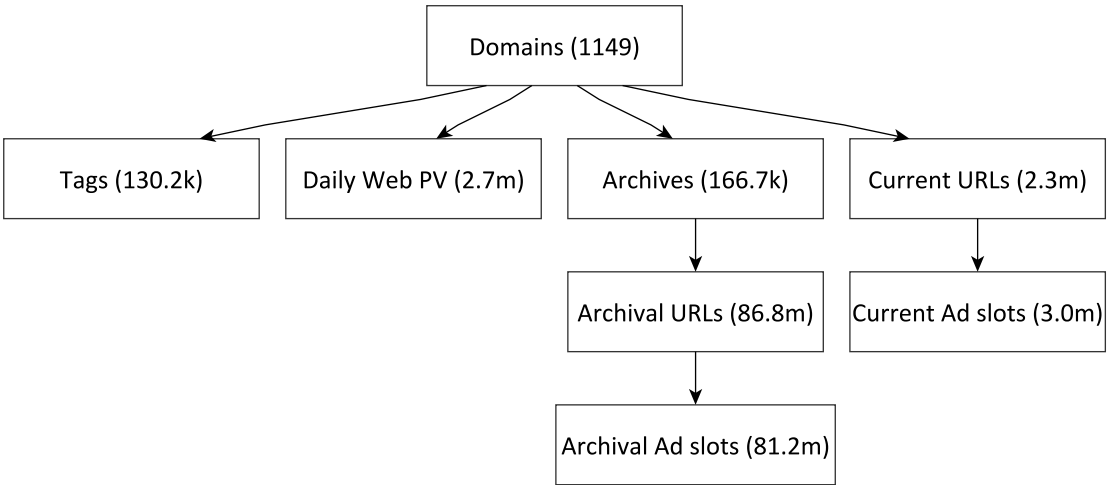


**Figure 3.2:** An overview of datasets and their relationships used in the empirical study on ad density. Numbers in parentheses are record counts.

quality, but not the type of task (informational or navigational).

## 3.2 An Empirical Study on Ad Density

This section presents an empirical study on display ad density evolution over recent years. The analysis on evolution helps to understand the online advertising business, provides insights into the whole ecosystem, especially from the users' perspective, and unveils important factors that affect the ad density decision. This section firstly describes the methodology of crawling data from various sources and then report findings by linking datasets together.

### 3.2.1 Datasets and Procedures

The empirical study consists of many datasets. An overview of datasets is given in Figure 3.2. As the start, 11k domains were selected from the top million[1]. These domains were selected because their daily impressions (page-views, PVs) and users' interest segments are available, too. Depending on when the publisher implemented the traffic analytic service, the reported daily PVs varies. The histogram of reported lifetime is illustrated in Figure 3.3. A long reported lifetime is essential since later the time series of daily PVs and ad density will be investigated. At last, 1149 domains that have reported more than seven years' daily PVs data were chosen for further analysis.

It is worth pointing out that impressions do not always equal to PVs, especially when bid requests from each ad slot are sent separately. However, this number is almost

[1] `www.quantcast.com/top-sites`

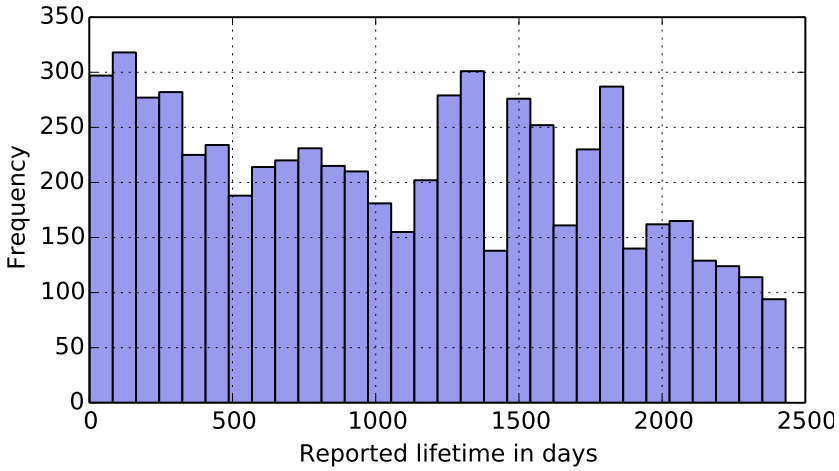


**Figure 3.3:** The time span of daily impressions (page views) for domains. Many domains report daily traffic for as long as 7 years. They were selected for further analysis.

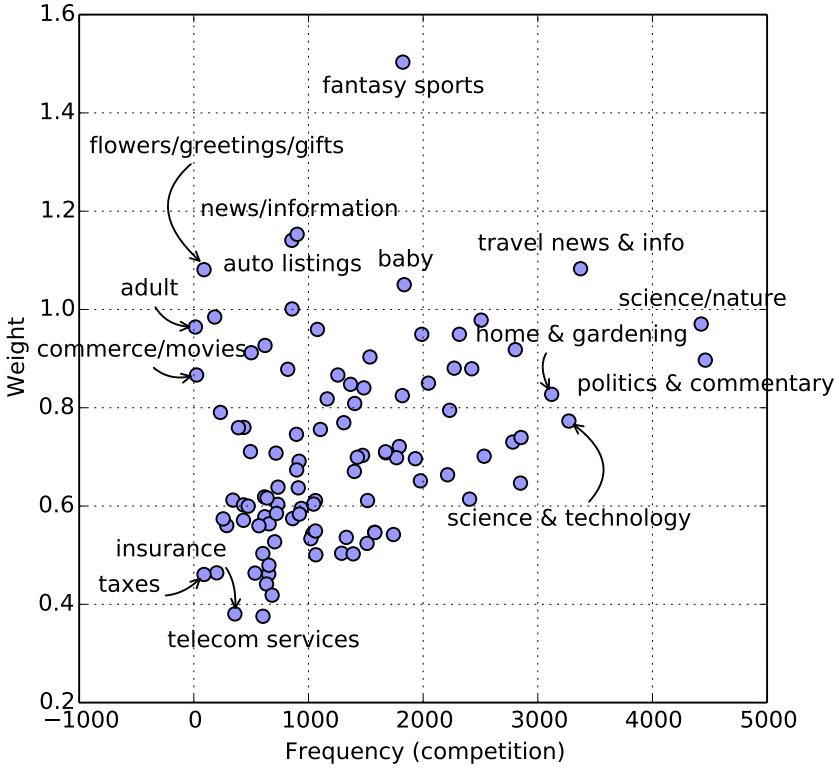


**Figure 3.4:** The second level tags' average weights (how users are addicted) and frequencies for selected domains. Both head and tail tags w.r.t. weight or frequency are annotated. A higher weight indicates that visitors are easier to target because they use the website a lot; a higher frequency means a lot of similar websites, thus, more competition on the marketplace. *Fantasy sports* players show exceptional high interest and loyalty which make the tag become almost an outlier.

impossible to calibrate. In this study, PVs are still considered to be reflecting the users' preference and determining the income of an ad-supported website.

Then for each domain the audience profile was obtained. These profiles are cre-

**Table 3.1:** Statistics for selected first level tags in 2014. Data includes width & height of webpage documents, number of ads slots and average ads densities. The last column, average competition, is computed by counting the number of domains sharing the same tag then normalised by dividing by the maximum. Note that the document width is largely a result of responsive design (automatic width resizing according to the browser window) and the 1366 pixels width of the headless browser.

| 1st level tag | Doc. Width (px) | Doc. Height (px) | URLs | Ad slots (k) | Avg. Ad area (k px^2) | Avg. Ad density (%) | Avg. competition |
|---|---|---|---|---|---|---|---|
| commerce | 1357.8 60.6 | 3612.9 6462.7 | 48584 | 496.0 | 455.5 | 9.3 | 0.28 |
| finance | 1364.1 18.0 | 3422.3 6160.7 | 8254 | 76.7 | 218.1 | 4.7 | 0.18 |
| family | 1355.2 58.0 | 3933.7 8170.3 | 73143 | 472.9 | 668.3 | 12.5 | 0.60 |
| entertainment | 1359.2 46.1 | 4316.4 8848.9 | 56129 | 550.3 | 534.4 | 9.1 | 0.44 |
| career | 1362.1 21.9 | 2833.9 2848.1 | 5661 | 56.3 | 270.3 | 7.0 | 0.46 |
| auto | 1361.4 42.1 | 3605.9 6535.1 | 25791 | 267.3 | 377.4 | 7.7 | 0.51 |
| news | 1358.4 47.3 | 4055.3 7121.1 | 40499 | 421.7 | 469.8 | 8.5 | 0.48 |
| education | 1355.8 56.9 | 4557.9 9987.8 | 52210 | 508.1 | 549.1 | 8.9 | 1.00 |

ated by a commercial Data Management Platform (DMP) based on billions of online tracking records. Instead of classifying domains based on their content, domains were tagged with users' interest segments (a.k.a., audience profiles, or tags). This approach is getting more popular in recent years, especially with the emergence of RTB and DMPs. It is fair to consider this approach beneficial in the study as more ads are displayed based on users' interest, too. By linking them together directly instead of using the website content as a proxy, users' preference could be reflected more closely. Provided by the same DMP, there are ten first level tags and 101 second level tags in total in the dataset. The frequency and average weight of tags (i.e., what is users' affinity with the website) are plotted in Figure 3.4. Interestingly, *fantasy sports* players show exceptional loyalty that makes the tag become almost an outlier.

For each domain in the dataset, the Wayback Machine [106] of the Internet Archive project was queried for archival webpages. Each domain was also accessed directly for the current snapshot. Taking those as seeds, out-links under the same domain were crawled with a depth of 3. This built the websites' archive & current snapshot dataset. The number of archives for domains are increasing over years, as shown in Figure 3.5. Although the earliest archives were from 1996, only the archival webpages from 2008 were analysed and plotted when the daily PVs data also became available.

Lastly, each archival and current webpages were analysed for ad slots. Every

**Table 3.2:** Daily PVs and effective-CPM estimation for the selected first level tags from Google AdWords in 2014. The figures were acquired by submitting keywords to the traffic estimation tool.

| 1st level tags | 2nd level tags (examples) | eCPM ($) | Daily PV (m) |
|---|---|---|---|
| commerce | commerce/movies, commerce/music, consumer electronics, consumer goods | 0.578 | 0.816 |
| finance | banking, online trading, insurance, taxes | 1.597 | 1.033 |
| family | family, baby, bridal, kids, kids education, parenting | 0.970 | 0.609 |
| entertainment | books, magazines, movies, music/radio, gaming information | 1.319 | 3.703 |
| career | career resources, job search | 0.987 | 0.228 |
| auto | auto listings, auto manufacturers, auto news & info, auto resources, car rental | 0.741 | 0.307 |
| news | news/information, business news & info, weather | 0.428 | 1.567 |
| education | educational resources, schools/universities, technology | 0.881 | 0.038 |

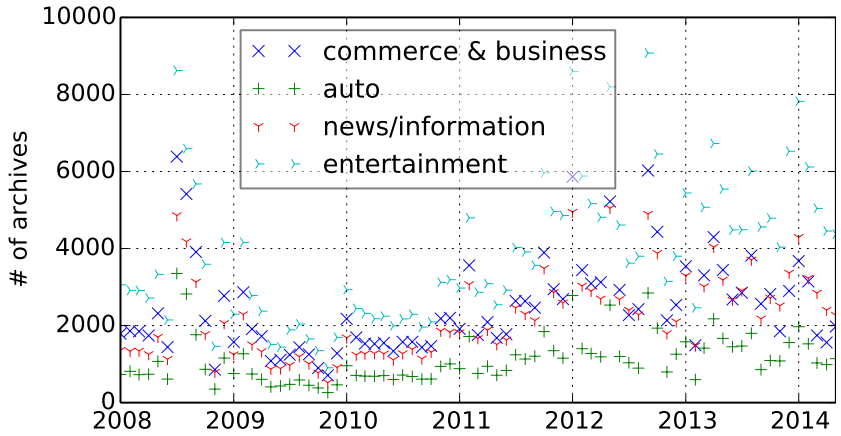


**Figure 3.5:** The number of archives for tags over years. Only the archival webpages from 2008 were analysed and plotted when the daily PVs data also became available. In fact, the dataset contain archives from 1996. Note that since the analysis was performed in the early 2014, the archives for this year is yet not complete. This would be compensated by the analysis on current URLs.

webpage was loaded in a headless browser (without the graphical interface), waited until fully rendered, then parsed for ad slots based on various rules. Commonly known advertising domains [107], typical ad unit sizes [9], and AdBlock Plus subscription lists [108] were used to identify ad slots. When an ad slot was found, a piece of JavaScript code would be injected to obtain the size and position of the slot. The size of the whole webpage was also saved. If a webpage failed to render, it would be excluded from all datasets.

Note that there are some limitation of the crawling and analysing procedure:

- For some domains, the archive may not be available (but at the tag level there are always thousands of archives);
- Some historical web servers may not be available, resulting in incompletely ren-

dering of webpages;

- Some historical ad servers may not be available, resulting in incompletely rendering of ads;
- Some uncommon ad slots could not be identified.

However, since the empirical study focuses on the trends instead of accurate ad density, these limitations do not harm the validity of the study because they always exist.

When a webpage cannot be correctly rendered it usually gives a bigger ad density, especially with the lack of CSS files which control the layout of elements and make the parallel positioning possible; when ad cannot be rendered, or ad slots cannot be identified, they give a smaller ad density. In order to minimise the impact, as many data points were collected and the URL was redirected to the current URL if the archival one was not available or not included due to *robots.txt* settings.

### 3.2.2 Results

The analysis has two parts. First the current status of display ads across user' interest segments (tags) is studied; then the ad density evolution over past years is reported and compared with historical daily PVs. Since the current snapshot does not rely on the Internet Archive service or obsolete ad servers, the figure would be more accurate.

#### 3.2.2.1 Statistics of the Current Snapshot

The main result is summarised in Table 3.1. It is not surprising to see webpages report a stable width but a considerably varying height. Websites with different tags contain very different number of URLs (thus ad slots) due to their nature of business. For example, *finance* and *career* are meant to offer services and very specific information; while *entertainment* and *news* are designed to provide as many articles as possible.

Besides, comparing with Table 3.2 and Figure 3.4 shows that the unit payoff and competition heavily affect the ad density decision. For example, *finance* reports ad density $\rho = 4.7\%$ because its big unit payoff $c = \$1.597$, average daily impressions $\bar{x} = 1.033$ million and low market competition; on the contrary, *family* reports a much larger average ad density because its smaller $c = \$0.970$, $\bar{x} = 0.609$ million, and higher competition level. This leads to an important finding of the relationship of ad density, competition, and unit payoff, as shown in Figure 3.6. The *competition* is defined as the number of websites sharing the same tag. The effective Cost Per Mille (eCPM) data is

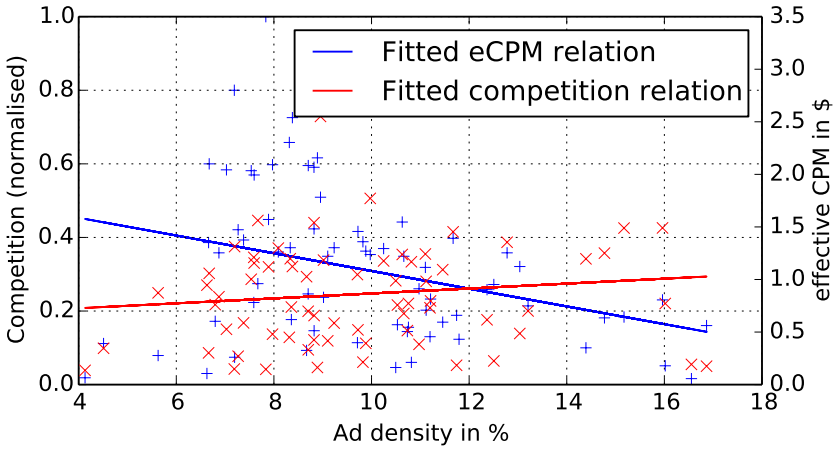


**Figure 3.6:** The relationship of competition, unit payoff and ad density plotted against all 2nd level tags. The competition is defined as the number of websites sharing the same tag (then normalised). The effective-cost-per-mille (eCPM) data is acquired from Google AdWords. The fitted lines suggest that higher competition leads to higher ad density, while higher eCPM leads to lower ad density.

acquired from Google AdWords by submitting keywords to the traffic estimation tool. The fitted lines suggest that higher competition leads to higher ad density while higher eCPM leads to lower ad density. This serves as part of the foundation of the proposed model in the later part of the paper.

Another interesting observation is on the popularity of ad slot sizes. There exists guidelines for ad slots and creative design[9]; however, there are also large number of ad slots that do not conform these guidelines, as shown in Figure 3.7. It is commonly known that ads of different size will deliver varying impact on campaigns' effectiveness, as well as users' satisfactory. However, this factor is not included in the proposed model. Interested readers may refer to [94].

#### 3.2.2.2 Ad density evolution

The evolution of ad density grouped by tags is plotted in Figure 3.8. Six tags are presented because of their representative trends.

First of all, a general declining trend of desktop PVs is found for most tags. This is consistent with the recent industrial findings [109]. It is especially true for *news* and *education*. This is mainly because of the rapid expansion of hand-held devices like smart mobile phones and tablets which take over the online traffic. People use desktops and laptops less because hand-held devices are more convenient and equally powerful in many cases. This has resulted in a traffic shifting from the desktop to

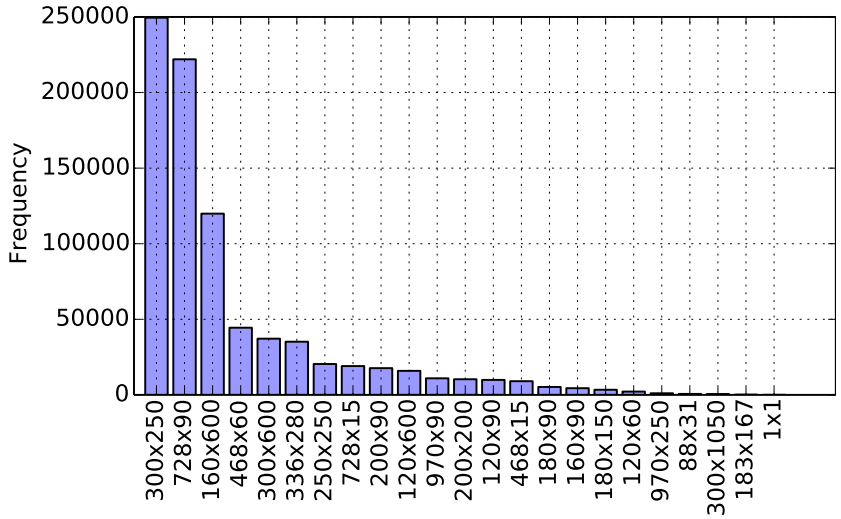


**Figure 3.7:** The distribution of top ad slot sizes in 2014.

the mobile environment. In this thesis, the desktop environment is still the focus as it is much more mature and richer in historical archives. However, it is believed that display ads (on both mobile websites and apps) are far from saturated in the mobile environment [110] that could make an interesting topic for the future works.

Secondly, despite that traffic is declining, the ad density for most categories is increasing, except for *finance* which shows a decreasing trend; and *news* which exhibits an almost flat line. Among them, the fastest growing category is *family* which gives a 17.65% annual change on average. If only considering first screens the figure becomes 27.58%.

Thirdly, ad density time series have good correlation with percentage changes of page-views, as shown below the time series plot of each first level tag. The correlation usually achieves the maximum with 1-2 time period lag, indicating the ad density usually changes after 1-2 years of observation of website traffic. By comparing the correlation of the percentage changes, it is not hard to see that page-views heavily affect the ad density decision. Although the website traffic is generally declining and the ad density is increasing, a fast PVs declining speed will slow down the increment of ads, e.g., *education*. This finding makes an important aspect of the proposed model.

Fourthly, most websites choose to have a slightly higher *first screen* ad density comparing with the whole webpage. The first screen is part of a webpage that is displayed (without scrolling) in the view-port of the browser after the webpage is loaded and rendered completely. It is seen by visitors by default, thus attracts most attention

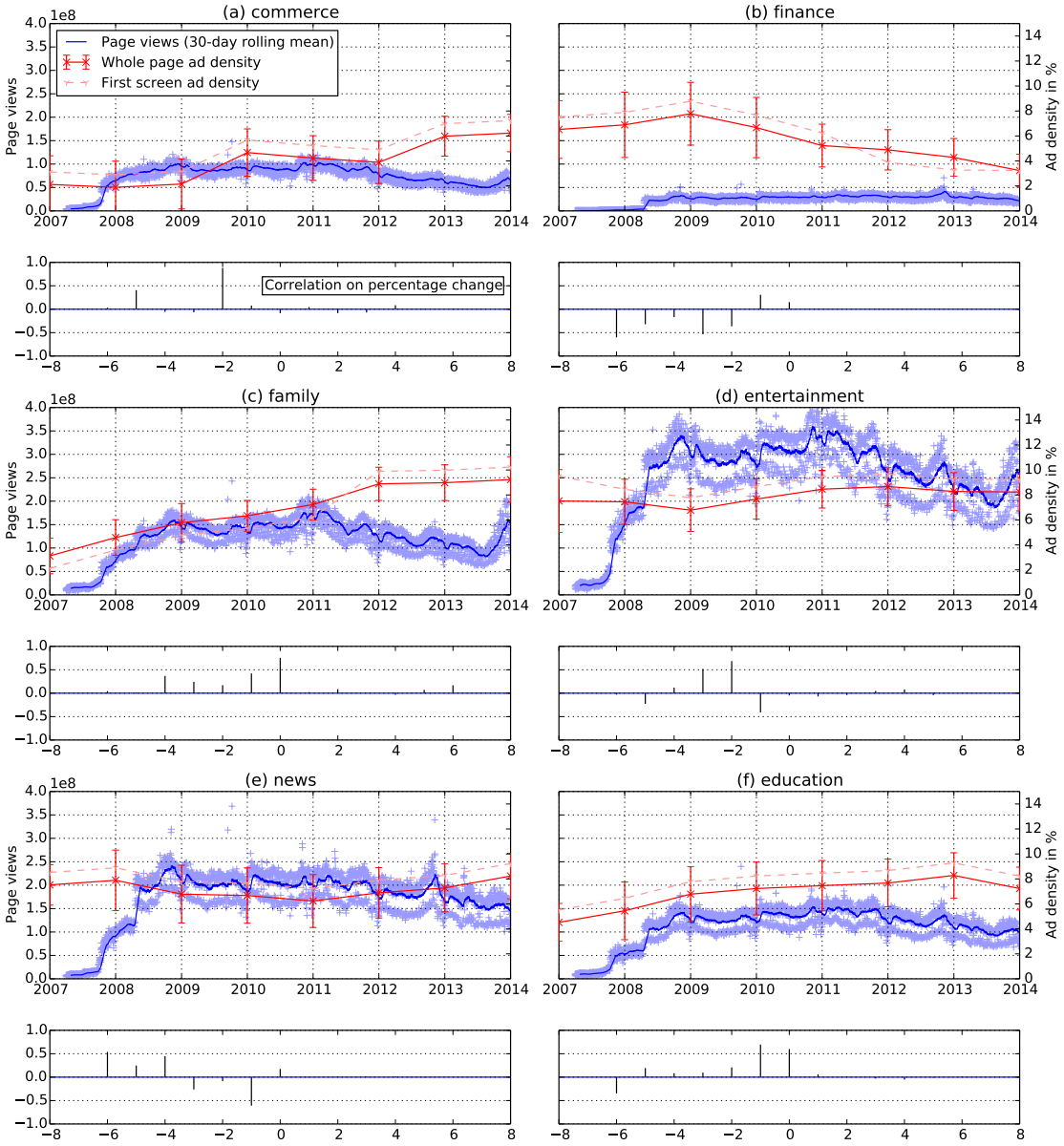


**Figure 3.8:** The evolution of monthly PVs v.s. ad density in recent years and correlation of their percentage change. Ad slots with lower border's vertical position within 1000 pixels from the top of a webpage are considered to be on the first screen. Six tags are selected and presented here to show different trends. Note that the PVs are declining for many tags. This is consistent with the recent industrial findings [109]. PVs and ad density usually have good correlation with some lags.

and usually considered *premium inventory* that generates a large proportion of revenue. In this empirical study, the first screen is defined to have a length of 1000 pixels. That is, if the vertical position of lower border of an ad is less than 1000 pixels from the top of the webpage, it is considered on the first screen.

Plots in Figure 3.8 show that most websites choose to use a higher ad density on the first screen. This is especially true for *commerce*, *entertainment*, *news*, and

*education* categories. *finance* unveils something interesting: although the overall ad density has been decreasing from 2013, the first screen one is not. This means the first screen ads are considered to be more important by publishers, thus may require separate consideration.

Lastly, there is a noticeable discrepancy when comparing ad density from historical archives of 2014 and the current snapshot (Table 3.1). As argued before, this is due to practical constraints on webpage crawling and ad slot identification. The ad density of historical archives tends to be smaller, which implies that ad tags expire more quickly than webpage content. Besides, when CSS or JavaScript files are masked by *robots.txt* the webpage content tends to occupy more space because of the loss of position and size configuration. However, these constraints exist for all archives. The overall trend is still valid and valuable.

To sum up, this empirical study exposes important trends of ad density evolution and its relationship with unit payoff and daily traffic. These directly support the proposed model in the next section.

## 3.3 Ad Density Modelling

This section discusses the ad density modelling. It is preferable to design a model that includes major factors and reflects findings from the empirical study. Consider a common case as the start: suppose there are two symmetric publishers $i$ and $j$ who run websites and subsidise the running cost by revenue from display advertising. Here the term *symmetric* means both publishers provide almost the same content or services (e.g., news, stock prices, etc.) therefore they compete for the same group of Internet users. These users have to fulfil their informational need by visiting either one of the websites, but not necessarily both of them. However, these users are advertising aversion that means they would prefer a website with fewer ads. To simplify the problem it is assumed that the movement of users is solely affected by advertising (i.e., more ads drive away users more quickly). Also note that this setup could be quickly expanded to cases with more competing publishers.

Take publisher $i$ for example. At each time step $t$ within the discrete planning horizon $0 \leq t \leq T$, the publisher decides the ad density $\rho_i$ of his website. Let the state variable $x_i(t)$ denote the number of impressions. The goal is to find a strategy

$\phi(\cdot) : t \to \rho(t)$ that maximises the cumulative payoff, which is defined as:

$$R(\boldsymbol{\rho}_i, \boldsymbol{x}_i) = \sum_{t}^{T} \left( c_i \rho_i(t) x_i(t) - \frac{k_i \left(1 - \rho_i(t)\right)^2}{2} \right) \tag{3.1}$$

where $c_i$ accounts for the unit payoff and $k_i$ for the running cost. Note that a finite planning horizon $T$ is used for the ease of derivation. By using a big enough $T$ the model could approximate the case of infinite planning horizon, which usually include a discount factor for future payoff as shown in Figure 3.11. Moreover, for the simplicity the terminal payoff of the planning horizon is assumed to be zero.

The individual state transition function for each publisher is inspired by the Lanchester's Laws [111]. They are mathematical formulae for calculating the relative strengths of a predator/prey pair. The Lanchester equations are differential equations describing the time dependence of two armies' strengths A and B as a function of time, with the function depending only on A and B. For example, suppose A and B are shooting a continuous stream of bullets at each other. Then the Lanchester's Square Law models the loss of soldiers as

$$\frac{dA}{dt} = -\beta B \tag{3.2}$$

$$\frac{dB}{dt} = -\alpha A \tag{3.3}$$

where $\alpha$ and $\beta$ are the offensive fire-power for A and B respectively.

Similarly, the ad density model considers the opposite decision as the fire-power of attracting visitors. Since these two publishers are competing with each other the state transition function (for publisher $i$) is defined as:

$$\Delta x_i(t) = n \rho_j(t) x_j(t) - m \rho_i(t) x_i(t) + h_i \tag{3.4}$$

where parameters $m$ and $n$ models users' preferences: $0 \leq m \leq 1$ controls the rate that users leaving a website because of too many ads and $0 \leq n \leq m$ controls the rate that users turn to the competitor's website.; $h_i$ is the natural growth. The objective and transition functions for publisher $j$ are similar.

This is a typical example of differential games. In this game, every publisher

tries to maximise his own payoff defined by Equation 3.1 while taking into account the interaction with competitors defined by Equation 3.4, subject to the obvious constraint:

$$0 \leq \rho_{i,j} \leq 1 \tag{3.5}$$

The objective, transition functions, and constraint together from both publishers form a system to optimise. Note that there are only two publishers in the model but it could be extended to cases of more publishers easily.

## 3.4 Solutions and Discussions

A Nash equilibrium solution to the above differential game can be found as follows. Firstly, the Hamiltonians are formed by introducing adjoint variables $\lambda_i$ and $\lambda_j$; then, two groups of partial differential equations (PDEs) could be found. For example, for publisher $i$ the Hamiltonian is:

$$\begin{aligned} &H\big(x_i(t), \rho_i(t), \lambda_i(t)\big) \\ &= c_i\rho_i(t)x_i(t) - \frac{k_i\left(1-\rho_i(t)\right)^2}{2} + \lambda_i(t)\big(n\rho_j(t)x_j(t) - m\rho_i(t)x_i(t) + h_i\big) \end{aligned} \tag{3.6}$$

where $x_j(t)$ is treated as a known parameter. From the Hamiltonian the PDEs are obtained:

$$\begin{aligned} \frac{\partial H}{\partial \rho_i(t)} &= 0 \\ &= c_i x_i(t) + k_i\big(1-\rho_i(t)\big) - x_i(t)m\lambda_i(t) \end{aligned} \tag{3.7}$$

$$\begin{aligned} \frac{\partial H}{\partial x_i(t)} &= -\Delta\lambda_i(t) \\ &= c_i\rho_i(t) - m\lambda_i(t)\rho_i(t) \end{aligned} \tag{3.8}$$

Suppose both $\lambda_i$ and $\lambda_j$ satisfy the condition that make the optimal ad densities non-zero. Then the adjoint variables are obtained by solving a boundary value problem

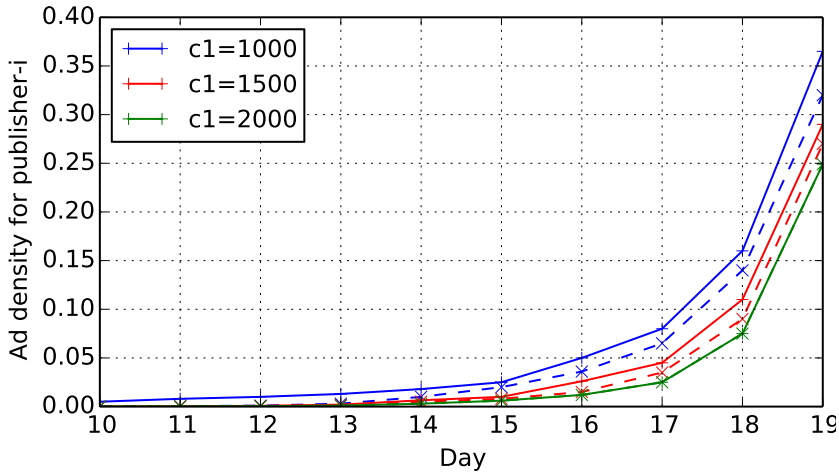


**Figure 3.9:** When the two publishers have different CPM $c$ the optimal controls become different. The solid lines are for publisher-$i$ and dashed lines are for publisher-$j$ who has a fixed $c_2 = 2000$. However, this is not validated in the empirical study due to no access to the unit payoff for each website. Note the last step always gives $\rho = 1$ due to the zero terminal value thus omitted in all plots.

with the terminal conditions:

$$\lambda_i(T) = 0 \tag{3.9}$$

$$\lambda_j(T) = 0 \tag{3.10}$$

Then the optimal controls are obtained by solving the following equations:

$$\rho_1(t) \tag{3.11}$$
$$= \frac{-\Delta\lambda_1(t+1) - \lambda_2(t)n\rho_2(t)}{c_1 - \lambda_1(t)m}$$

$$\rho_2(t) \tag{3.12}$$
$$= \frac{c_1\Delta\lambda_2(t+1) - \lambda_1(t)\big(n\Delta\lambda_1(t+1) + m\Delta\lambda_2(t+1)\big)}{c_1c_2 + m\big(c_1\lambda_2(t) - c_2\lambda_1(t)\big) - \lambda_1(t)\lambda_2(t)\big(n^2 - m^2\big)}$$

Finally, the state variables are obtained by substituting $\boldsymbol{\rho}_i$ and $\boldsymbol{\rho}_j$ into Equation 3.4 with the initial conditions:

$$x_1(0) = x_1^0 \tag{3.13}$$

$$x_2(0) = x_2^0 \tag{3.14}$$

It is expected that if two publishers have different parameters the optimal controls

become different, although this is not validated in the empirical study since there was no access to the unit payoff for each website. For example, when publisher-$j$ has higher CPM $c_1$, he could display fewer ads to build a greater user base, resulting in a higher return in the longer term while having the same payoff in the short term. This is illustrated in Figure 3.9. On the contrary, when publisher-$i$ has a lower unit payoff $c_1$ he has to display more ads to cover the maintenance cost.

### 3.4.1 Social Welfare Maximisation

This section discusses variations of the proposed model. The comparison of these variations leads to the discussion of social welfare maximisation and tragedy of the commons phenomenon. These discussions can map back to real-world cases reported by the empirical study thus are helpful for understanding the display advertising ecosystem in the long term.

Firstly, let us simplify the system by removing competition. When there is only a single (dominant) publisher on the marketplace, the solution looks different. In this case, the individual optimum achieved by the publisher is indeed the global optimum. In the real-world and in some countries, the Google search engine (67.6% in US in 2014 [112] and Facebook (71.0% in US in 2013 [113]) serve good examples to this scenario, although not perfect. The following discusses both cases that a publisher use a dynamic ad density or has to stick to a static one.

#### 3.4.1.1 Dynamic Ad Density

The publisher is still expected to decide a dynamic ad density. Then, the problem becomes a standard optimal control problem with a single state variable and a control variable. Here the state transition function is:

$$\Delta x(t) = -m\rho(t)x(t) + h \tag{3.15}$$

The users from other websites have been removed due to no competition. But users are still free to leave if getting annoyed.

The objective function remains the same:

$$R(\boldsymbol{\rho}, \boldsymbol{x}) = \sum_{t}^{T} \left( c\rho(t)x(t) - k\frac{(1-\rho(t))^2}{2} \right) \tag{3.16}$$

where the terminal value is still assumed to be zero.

To solve this problem, the Hamiltonian can be constructed by introducing an adjoint variable $\lambda$:

$$H(x(t), \rho(t), \lambda(t)) = c\rho(t)x(t) - k\frac{(1-\rho(t))^2}{2} + \lambda(t)\big(-m\rho(t)x(t) + h\big) \tag{3.17}$$

Similarly, the partial differential equations are obtained by examining the Hamiltonian:

$$\frac{\partial H}{\partial x} = c\rho(t) - m\lambda(t)\rho(t) = \lambda(t) - \lambda(t+1) \tag{3.18}$$

and applying the Pontryagin's maximum principle:

$$\frac{\partial H}{\partial \rho} = cx(t) + k\big(1-\rho(t)\big) - mx(t)\lambda(t) = 0 \tag{3.19}$$

Finally, the iterative solutions could be obtained by solving the boundary value problem with the initial and terminal conditions:

$$x(0) = x^0 \tag{3.20}$$

$$\lambda(T) = 0 \tag{3.21}$$

Here, let us briefly examine parameters of the simplified model. Parameter $c$ is the unit payoff (usually CPM) for impressions. The empirical study suggests that it has a negative relationship with ad density: when $c$ is small, the publisher has to use a higher ad density to compensate the running cost; on the contrary, good profit from ads allows the publisher to use a lower ad density to build a greater user base, which is beneficial for the long term. This relationship is confirmed in Figure 3.10 (a).

Although not presented in the empirical study due to difficulty in accessing data, it is not hard to imagine that running cost factor $k$ also affects the payoff and ad density decision. A high running cost factor would reduce the payoff and force the publisher to use a higher ad density; on the contrary, a small $k$ allows the publisher to employ a small ad density while giving higher long term payoff. This relationship is illustrated

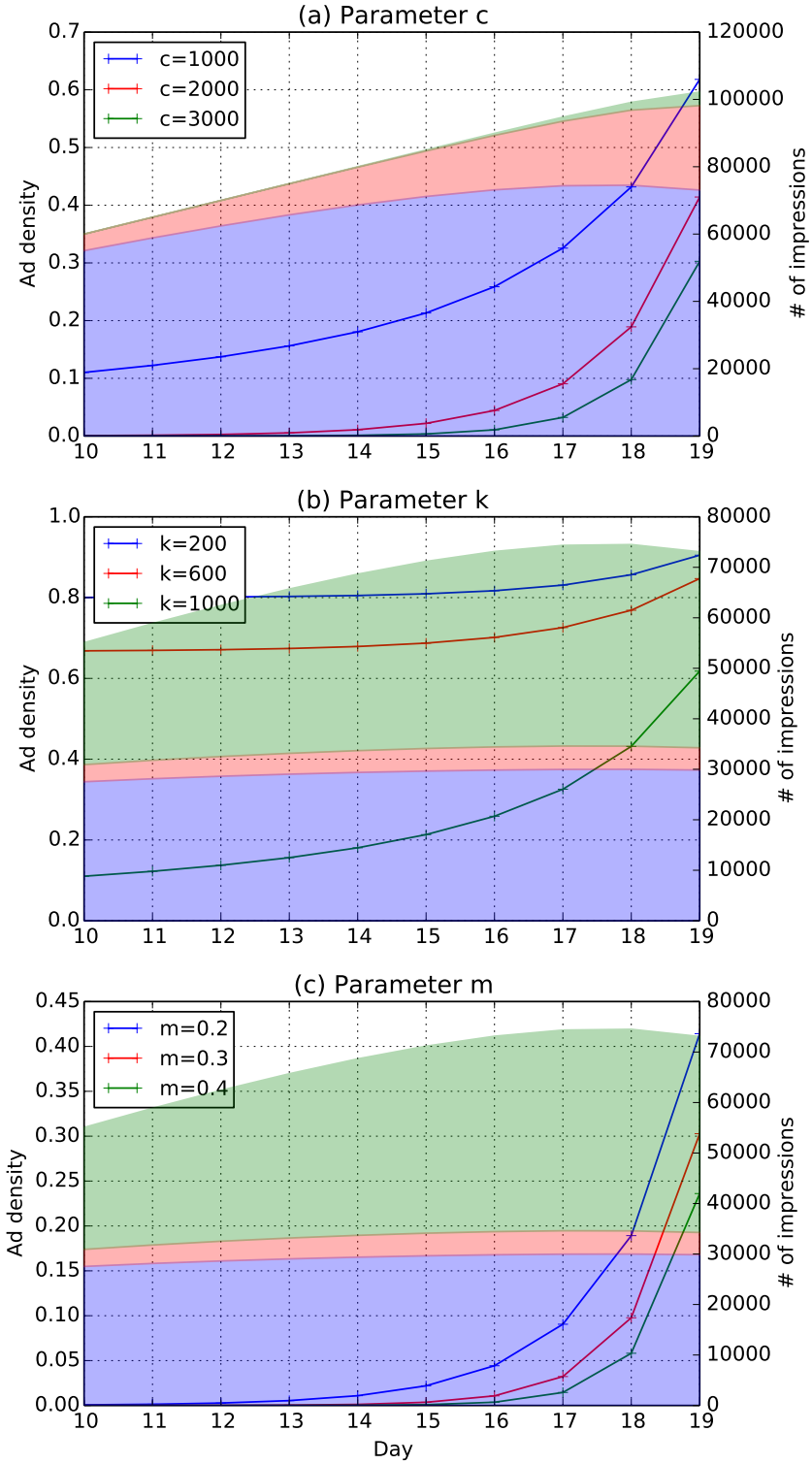


**Figure 3.10:** Parameters affect the planning. The unit payoff factor $c$ and user attrition factor $m$ have negative relationship with ad density; the running cost factor $k$ has positive relationship with ad density.

in Figure 3.10 (b).

It is worth pointing out that both parameters $c$ and $k$ are easy to measure and straightforward to understand. In order to optimise the cumulative revenue over the

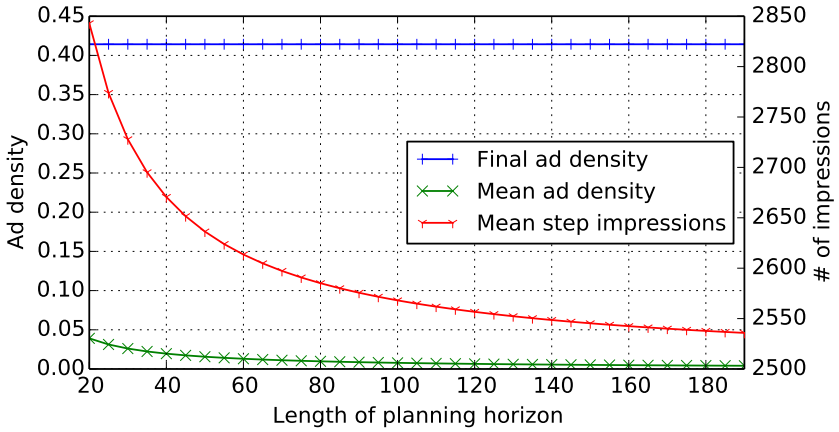


**Figure 3.11:** Parameter $T$ affects the planning. Using a large $T$ allows publishers to approximate the infinite planning horizon.

planning horizon, the publisher should try to find ad channels that give high return [114], and reduce the running cost, too. Note that here $c$ is assumed to be a known constant, while it is usually an unknown random variable in practice. In such cases, learning based techniques could be used to find $c$. Besides, deviation or confidence intervals of the estimation could be used as the risk of the planning [115]. These are not discussed in the thesis but left to the future works.

Another important factor of the model is $m$ which reflects the users' attitude over receiving ads. In this thesis, it is called the *user attrition factor*. A high $m$ means more impressions would get lost for the next time step (more users leaving) due to displaying ads. In such case, a conservative approach is naturally required as illustrated in Figure 3.10 (c).

At last let us examine the parameter $T$. Not surprisingly the $T$ would affect cumulatively payoff and average impressions. However, it is worth noting that $T$ does not change the final ad density given other factors remaining the same as shown in Figure 3.11. This finding confirms that the proposed model is practical in real-world applications: a publisher could start with a big $T$ and re-evaluate parameters in later stages. This is, in fact, a standard practice given the rapid development of display advertising ecosystem.

#### 3.4.1.2 Tragedy of the Commons

The term tragedy of the commons refers to the case that individuals act independently and rationally to maximise one's own interest, however these actions are contrary to the

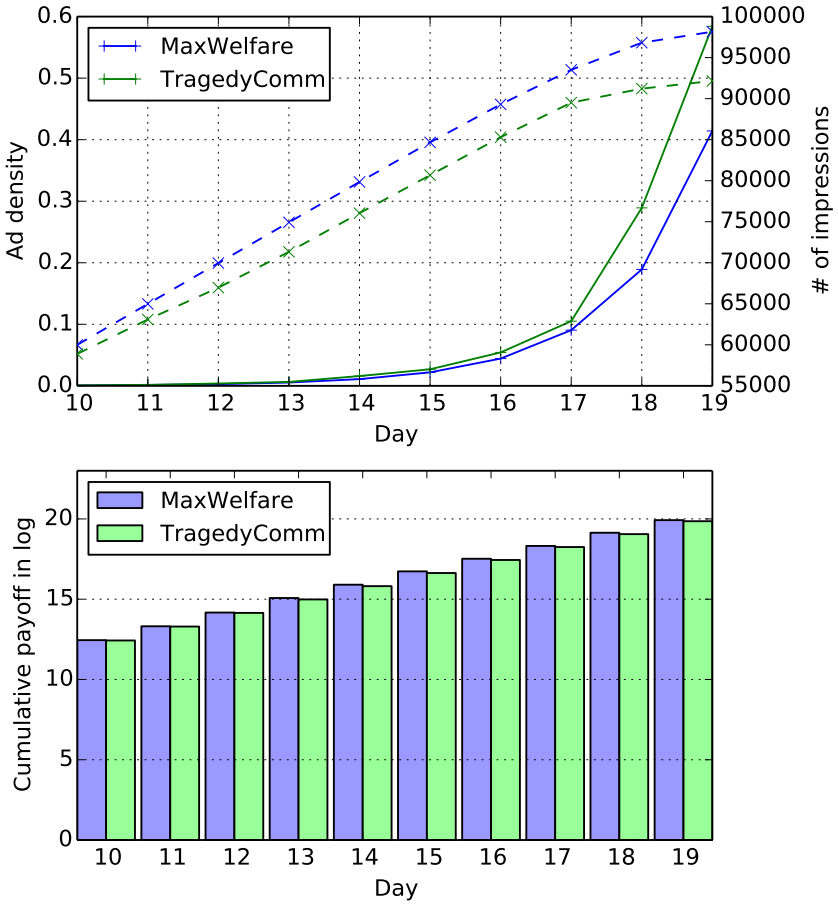


**Figure 3.12:** Comparing ad density decisions, cumulative payoff, and public resources (impressions) for cases with or without competition. The case with competition leads to tragedy of the commons while the no-competition case represents the social welfare maximisation. For the former one, the ad density (solid lines) is higher while the impressions (dashed lines) are lower than the global optimal which is obtained from a single publisher case. As a result, users are less satisfied, and the cumulative payoff is smaller.

whole group's long-term best interest by depleting some common resource. In online advertising case, competing publishers want to maximise one's cumulative payoff. Although each publisher's decision is optimal within the planning horizon, the decisions as a whole are sub-optimal to the group. The common resource being depleted here is impressions (users): website visitors get less satisfied because of being shown more ads; at last, they choose to leave both websites.

It has been shown in the empirical study that competition has a positive relationship with ad density. This assumption is not difficult to validate by comparing two cases. The ad density, impressions, and cumulative payoff for both models are plotted in Figure 3.12 and consider the single publisher case to be global optimal. The following parameters are used for the MAXWELFARE model: $x^0 = 10000, h = 5000, m =$

$0.2, c = 2000, k = 100$

For the TRAGEDYCOMM model it is assumed that there are two competing publishers and simply split the initial user base and natural growth, and take into account the users' transition: $c_1 = c_2 = 2000, k_1 = k_2 = 100, x_1^0 = x_2^0 = 5000, h_1 = h_2 = 2500, m = 0.4, n = 0.2$

Note that impressions and cumulative payoff are aggregated for the MAXWELFARE case. Both publishers follow the same ad density curve since they are symmetric. The plot shows that the TRAGEDYCOMM case builds a smaller user base and receives less cumulative payoff almost at every time step. It indicates that ads are overcrowded and users' attention has been overgrazed, thus, users are less satisfied. This could harm the online advertising ecosystem in the long term. It is necessary to develop countermeasures to promote more sensible ad density [32].

#### 3.4.1.3 Static Ad Density

Lastly, the model is simplified further by assuming that a static ad density is used throughout the planning horizon. In fact, this is never a rare case in real-world since changing templates is costly for most websites. Thus, publishers have to do the change infrequently and rely on static ad densities between changes.

The solution to this variation is obtained by inspecting the first order derivative of the payoff function. Firstly, let us review the payoff and transition function for this case:

$$R(\rho, \boldsymbol{x}) = \sum_{t}^{T} \left( c\rho x(t) - k\frac{(1-\rho)^2}{2} \right) \tag{3.22}$$

$$\Delta x(t) = -m\rho x(t) + h \tag{3.23}$$

the latter yields:

$$x(t) = (1 - m\rho)^t x^0 + \frac{1 - (1 - m\rho)^t}{m\rho} h \tag{3.24}$$

Again, the goal of the publisher is to maximise the total payoff $R$ within the planning horizon $1 \leq t \leq T$.

The optimal $\rho$ could be found by taking the derivative of $R$ and letting it equals to

zero:

$$\frac{dR}{d\rho} = 0 \tag{3.25}$$

which gives:

$$\begin{aligned}\rho \approx & \frac{-Tkm + 2ch + 2cx^0}{6cmx^0} \\ & + \frac{\sqrt{Tkm\big(TKm + 4c(3mx^0 - h - x^0)\big) + \big(2c(h + x^0)\big)^2}}{6cmx^0}\end{aligned} \tag{3.26}$$

by assuming $(1 - m\rho)^T \approx 0$. The optimality could be proved by examining the second order derivative which is omitted here.

## 3.5 Conclusion

The ad density problem has been studied in this chapter from both empirical and theoretical perspectives. Results from a large-scale real-world dataset are presented which unveils the general growing trend of online display ads in recent years. Besides, important findings are reported including: higher unit payoff leads to a lower ad density but higher competition lead to a higher one; ad density has been gradually increasing over years and the incremental speed is affected by the change rate of traffic. Based on these findings, the ad density model is formulated and solutions are provided for different cases using control theory and game theory. The model and solution map to the results of empirical study well. Specifically, the maximisation of social welfare and tragedy of the commons phenomenon is discussed. It shows that online display ads are overcrowded and users' attention has been overgrazed. Due to the complexity of real-world business the model is not perfect (e.g., some publishers mainly rely on subscription fees thus have few ads). It could also be improved by making unit payoff and cost parameters stochastic. These are left to the future works.

## Chapter 4

# Sequential Ads Selection

Given an ad density, a publisher can estimate the number of impressions based on the historical website traffic. However, as illustrated in

Besides an ad density and the expected number of impressions, a publisher has more control over payoff maximisation over time by selecting ad sources. In practice, it can be challenging because of abundant options great volatility of payoff.

First, one has to decide whether to make contracts with advertisers or agencies directly or to register with public ad networks or exchanges to reach more demands. For example, in [40] a queuing system is proposed to find the optimal policy of selecting advertisers to make private contracts, whereas in [16, 99], the focus is on pricing ads properly in either of the two settings.

Second, the payment scheme could be different, and the publisher needs to choose from CPM, CPC, and possibly other models, c.f. definitions in Appendix A. To achieve the maximum revenue, a balance of such choices can be established in a static setting as reported in [44, 43, 45].

Lastly and most importantly, the publisher has to decide on which page [24, 116] and which users [117, 118] these ads should be matched with, probably in a real-time fashion using text summaries [21]. These are also known as contextual and behaviour targeting. Traditionally the matching is done by ad networks and advertisers are allowed to choose which keywords they intend to bid. Now publishers are getting more involved and can actively switch in nearly real-time between different pricing schemes and different sources in order to increase their ad payoffs as demonstrated in the Google Double Click For Publisher [119] and some ad exchanges like AppNexus [120].

The nearly real-time switching ability essentially allows online publishers to inte-

grate the above decisions all together in a more general framework to optimise their ad revenue. However, the following challenges remain unsolved because of its dynamic settings: First, the prior analysis of the matching between content and ads provides a fine start but a perfect match does not guarantee the maximum revenue. With the limited display impressions, publishers also need to balance the need of selecting ads and their pricing schemes to learn the true payoffs (exploration) with that of allocating ads to generate high immediate payoffs based on the current belief (exploitation). Second, online publishers have more opportunities to explore from different ad networks and pricing schemes provided ads are correlated. The key question is how to make use of the correlations embedded in the data to improve the efficiency of the exploration and increase the ad incomes in the long term.

This chapter studies the above two issues and formulate the sequential ad selection problem by applying Partially Observable Markov Decision Processes (POMDPs). To provide a basic understanding of publisher revenue problem when dealing with multiple ad sources, various trade mechanisms and pricing models are not distinguished. They are simply expected to contribute revenue once selected. Moreover, to make the research focused, the problem is formulated by considering the correlation of ads only while the same principle and result can be applied to the correlation of webpages and users. The mathematical derivation shows that the expected payoff and variance of correlated ads can be naturally updated using a formula similar to collaborative filtering (i.e., to learn the preference from the similar users). Then the model is examined on a carefully collected dataset, and the results show that the proposed models, particularly with the new belief updates, outperform other strong baselines.

## 4.1 Related Works

This section discusses the related works to the ad selection and keyword extraction problems which are studied in this chapter.

### 4.1.1 Ad Selection

Traditionally, the research question is limited to how to choose advertisers to make (private) contracts with, as elaborated in [121, 40, 14]. In [40] the authors use a queueing system to accommodate advertisers and add constraint that advertisers are impatient

and would leave if their ads cannot be displayed right way. A dynamic programming solution is provided in [14] by combining the available ad inventories and dynamically delivery of promised advertising contract to the viewers. This inspires the development of this chapter.

In terms of techniques, this chapter is closely related to multi-arm bandit with dependent arms discussed in [122]. Because it also dels with a multi-period selection problem with exploration and exploitation dilemma. In [122] the dependencies of candidates are modelled by clustering them first; after that, a multi-arm bandit algorithm runs twice: first selects a cluster and then a candidate in that cluster. Our paper is different in that the dependency is directly modelled using a covariance matrix. The impact of correlation is well formulated and illustrated by rigorously deriving the belief updates once other correlated ads have been selected. Specifically, the problem is formulated by applying Partially Observable Markov Decision Processes (POMDPs) with discrete action (selection of ads), continuous observations (payoffs), and continuous hidden states (performance of ads). The proposed model is a special case of continuous POMDPs [123, 124], where two-stage Gaussian generative processes and no transit of hidden states are considered during the planning horizon. To provide an optimal ad selection, similar with [124, 125], the Monte Carlo sampling is used to deal with continuous observations and to approximate a Dynamic Programming solution in the finite planning horizon. As a complement of the approximation approach, the Upper Confidence Bound algorithm [126] is extended by integrating with the belief update.

### 4.1.2 Keyword Extraction

In the later part of this chapter an adaptive keyword extraction tool is presented. It is especially useful when an ad source allows (or expects) an input of keywords. There are commercial tools generating high co-occurrence or similar phrases as suggestions based on seed terms provided by user, like Google AdWords Keyword Tool[1]. There are two problems for these tools: 1) it still requires fair amount of manual work to input seeds and choose from suggestions; 2) the topics of generated phrases are based on user query logs, existing bid phrases, and lexical analysis, and may easily drift from the original from webpages.

[1] `http://adwords.google.com/select/KeywordToolExternal`

Keyword extraction is typically considered a supervised learning problem [127, 128, 129, 130] where the algorithms learn to classify as positive or negative examples of keywords based on training sets. These algorithms need expensive human labelled dataset in advance, and usually perform the learning process offline. The model could become inaccurate when users' interests change over time. In [128] linguistic knowledge is introduced such as noun-phrase-chunks (NP-chunks) and part-of-speech (POS) to outperform using statistics features only. These linguistic features are also used in the proposed system. Besides, query logs have been used as a good reflect of users' interests [131]. However, such logs are not available in this chapter.

There are also extensive research on keyword suggestion. In [132] the keywords are suggested based on concept hierarchy mapping therefore the suggestions are not limited to the bag-of-words of the webpage, and may expand to non-obvious ones which are categorised in bigger concepts. The work of [133, 134] recognise that the bid prices for hot keywords are high therefore would cost more, and try to find related non-obvious keywords that are cheaper. Although these keywords may have lower traffic, but when combined the traffic could match that of a hot one, while these keywords cost less in total.

In [20] the authors propose a classifier that uses multiple text features, including how often the term occurs in search query logs, to extract keywords for ad targeting. The system discussed in [130] first generates candidates by several methods including a translation model capable of generating phrases not appearing in the text of the pages. Then candidates are ranked in a probabilistic framework using both the translation model favouring relevant phrases, as well as a language model favouring well-formed phrases. Another relevant piece of work is [135]. The authors propose an exploration-exploitation algorithm of sorting keywords in an descending order of profit-to-cost ratio and adaptively identify the set of keywords to bid on based on historical performance with a daily budget constraint. The proposed system tries to find the keywords from the given webpage with additional web knowledge, rather than find profitable ones from a very large set (like 50k). Then the matching of keywords and ads is left to display ad networks/exchanges, such as Google AdSense.

In [136] the authors propose a combination algorithm of using Upper Confidence Bound (UCB) and $\epsilon$-greedy to solve the exploration-exploitation dilemma. Their target

is to select high profit ads which would be fed in contextual advertising platforms. The feature vectors of ads are not used in their system as side information, instead, they use standard bandits considering that the reward follows an unknown stochastic process. This partially inspires the development of this chapter.

## 4.2 The Sequential Payoff Model

This section formulates the sequential ad selection problem. Suppose there is an online publisher who wants to put ads on their hosting webpages to generate profits. The ads could be obtained from various sources either by making contracts with advertisers directly, by registering with ad networks or by employing a SSP [119]. Moreover, some ad networks allows (or expects) an input of keywords to return relevant ads. To help to address this challenge an adaptive keyword extraction system is proposed in the later part of this chapter.

Suppose there are $N$ ads from various ad networks or exchanges available for the publisher. For each display impression, the publisher needs to decide which ad source to select. Without loss of generality, let us consider making our selection decisions every $M$ impressions and denote the decision times as $t \in [1, ..., T]$. To stay focused, let us assume that impressions from the same webpage are independent while bearing in mind that the scenario of multiple impressions can be addressed by incorporating user click-through models to remove rank bias [76].

For each time step $t \in [1, ..., T]$, let us define

$$\Psi(t) = \begin{bmatrix} s(1), & x(1) \\ \dots, & \dots \\ s(t-1), & x(t-1) \end{bmatrix} \tag{4.1}$$

as the available information up to time $t$, where $s(t) \in N$ denotes the decision, i.e. the index of ad selected for the time step $t$. The random variable $X(t)$ is used to denote the payoff gained at time step $t$ and $x(t)$ its realization.

Let $\pi$ be an arbitrary choosing policy according to the information obtained so far.

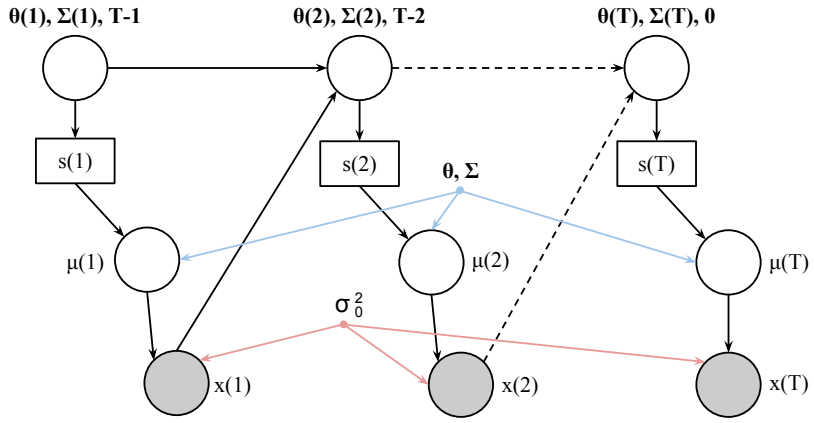


**Figure 4.1:** The payoff model illustrated by an influence diagram representation with generative processes of a finite horizon POMDP. In this diagram, the non-filled circular nodes represent variables including belief states; the shaded nodes represent rewards as well as observations of the system; the black point nodes represent non-random values; the non-filled square rectangular nodes represent actions. The dotted lines indicate indirect dependency and intermediate nodes are not drawn. Note that $s(\cdot)$ also depends on $\sigma_0^2$ but lines are not drawn for simplicity.

Let us define

$$s(t) \equiv \pi(\Psi(t)) \tag{4.2}$$

To simplify the notation, policy $\pi$ and decision $s(\cdot)$ are used interchangeably in the rest of the chapter. The cumulative payoff over time $T$ with certain policy $\pi$ is

$$R_\pi(T) \equiv M \sum_{t=1}^{T} X_{s(t)}(t) \tag{4.3}$$

The goal is to choose the optimal policy to maximise $R_\pi(T)$. However, there is no observation on any future $x_{s(t)}(t)$ (where $t \geq t'$) before making a decision about $s(t)$ (at $t = t'$). For a given webpage, let us assume two generative processes to generate the payoffs as shown in Figure 4.1. First, the matching between ads and the webpage is considered the true but unknown payoffs of ads over a webpage as $\boldsymbol{\mu}$, a $N$-dimension vector. This vector is generated from a multivariate Gaussian distribution governed by mean vector $\boldsymbol{\theta}$ and covariance matrix $\boldsymbol{\Sigma}$ as the following

$$\boldsymbol{\mu} \sim \mathcal{N}(\boldsymbol{\theta}, \boldsymbol{\Sigma}) \tag{4.4}$$

where $\boldsymbol{\theta}$ and $\boldsymbol{\Sigma}$ are the parameters of the model and can be estimated beforehand from data. Meanwhile, considering the fact that the payoffs are affected by either the visiting

users, some unexpected factors, or the uncertainty that has not been well modelled from the Gaussian, the observed payoffs are generated from the true payoffs by

$$\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma_0^2 \cdot \boldsymbol{I}) \tag{4.5}$$

where $\boldsymbol{X}$ is a $N$-dimension observed payoff vector and $\boldsymbol{I}$ is the identity matrix. A simple variance $\sigma_0^2$ is used to model the noise. A universal constant is used to describe the possible noise, as the treatment of the uncertainty from the underlying users. The experiments will show that the noise factor, although simple, plays an important role in controlling the sensitivity of the proposed model. For more advanced treatment about user modelling, interested readers may refer to [118]. Given the two-stage process (i.e., decide and update), the objective of the publisher is to find the optimal policy which maximises the expectation of the overall ad income through the period, i.e.,

$$\begin{aligned}
\pi^* &= \arg\max_{\pi} \mathbb{E}\left[R_\pi(T)\right] \\
&= \arg\max_{\pi} \mathbb{E}\left[\sum_{t=1}^{T} X_{s(t)}(t)\right] \\
&= \arg\max_{\pi} \sum_{t=1}^{T} \mathbb{E}\left[X_{s(t)}(t)\right] \\
&= \arg\max_{\pi} \sum_{t=1}^{T} \int_x x_{s(t)}(t) p(x_{s(t)}(t)|\Psi(t)) dx \\
&= \arg\max_{\pi} \sum_{t=1}^{T} \theta_{s(t)}(t)
\end{aligned} \tag{4.6}$$

where $M$ is dropped from Equation 4.3 because the decision is independent of it. Note that $\theta_{s(t)}(t)$ is the shorthand for $\hat{\mu}_{s(t)}|\Psi(t)$, denoting the estimated expectation of $\mu_{s(t)}$ at time step $t$ giving all available information $\Psi(t)$. It presents publisher's belief at time $t$. The above formulation is, in fact, a special case of continuous POMDPs [123, 124], where $\boldsymbol{\mu}$ is the hidden state, and $\boldsymbol{X}$ is the observation over time and the belief about $\boldsymbol{\mu}$ would be sequentially updated using a posterior probability. The next task is to provide the estimation $\theta_{s(t)}(t)$ at each time step $t$ after an ad has been selected, and its payoff has been observed. The belief update for correlated ads is particularly interesting and important, which will be discussed in the next section.

### 4.2.1 Belief Updates

At each time step the publisher makes a decision, observes the payoff, and then updates expected payoffs of all ads. In order to calculate the expected payoffs the publisher needs to calculate the value of $\boldsymbol{\theta}(t+1)$ and $\boldsymbol{\Sigma}(t+1)$ according to observation $x_{s(t)}(t)$ and the previous belief. This section derives the update equation using Bayesian inference. Let us first look at the brief update of the same ads. Suppose the publisher has two ad sources. The first was selected at time step $t$ and received a payoff of $x_1(t)$. With Bayes' theorem and marginalizing $\mu_1$ out, the p.d.f. of $X_1$ conditioned on the new observation $x_1(t)$ and previous available information $\Psi(t)$ is obtained:

$$\begin{aligned} & p\left(x_1|x_1(t), \Psi(t)\right) \\ & = \int p\left(x_1|x_1(t), \Psi(t), \mu_1\right) p\left(\mu_1|x_1(t), \Psi(t)\right) d\mu \end{aligned} \tag{4.7}$$

where

$$\begin{aligned} p\big(\mu_1|x_1(t), \Psi(t)\big) & \propto \; p(x_1(t)|\mu_1, \Psi(t))p(\mu_1|\Psi(t)) \\ \propto \exp\Big\{-\big(x_1(t)-\mu_1\big)^2 - \big(\mu_1 - \theta_1(t)\big)^2\Big\} & \end{aligned} \tag{4.8}$$

and by inspecting the exponent part, find the posterior distribution of $\mu_1$ is found as

$$\begin{aligned} \mu_1|x_1(t) &\sim \mathcal{N}\left(\theta_1(t+1), \sigma_1^2(t+1)\right) \\ \theta_1(t+1) &= \frac{\sigma_1^2(t)x_1(t) + \sigma_0^2\theta_1(t)}{\sigma_1^2(t) + \sigma_0^2} \\ \sigma_1^2(t+1) &= \frac{\sigma_1^2(t)\sigma_0^2}{\sigma_1^2(t) + \sigma_0^2} \end{aligned} \tag{4.9}$$

where similarly $\sigma_i^2(t)$ is used as the shorthand for $\sigma_i^2|\Psi(t)$.

Substituting the posterior of $\mu_1$ into Equation 4.7 gives the expected payoff of the selected ad as

$$X_1|x_1(t), \Psi(t) \sim \mathcal{N}\left(\theta_1(t+1), \sigma_0^2 + \sigma_1^2(t+1)\right) \tag{4.10}$$

where recall that the prior noise $\sigma_0^2$ is assumed known.

### 4.2.2 Correlated Ads

In the real world situation, the payoffs of ads are correlated. Similar products, using similar creative, or targeting similar potential customers will generate similar payoffs. By considering the correlation of ads the publisher could find a more efficient way of identifying best candidates because not only the beliefs of the selected ads themselves can be updated using Equation 4.10, but so do the other correlated ads. Again with Bayes' theorem and marginalizing $\mu_1$ out, the p.d.f. of $X_2$ conditioned on the observation $x_1(t)$ and previous available information $\Psi(t)$ is obtained:

$$\begin{aligned} & p\left(x_2|x_1(t), \Psi(t)\right) \\ & = \int p\left(x_2|\mu_2, x_1(t), \Psi(t)\right) p(\mu_2|x_1(t), \Psi(t)) d\mu_2 \end{aligned} \tag{4.11}$$

where

$$\begin{aligned} & p(\mu_2|x_1(t), \Psi(t)) \\ & \propto p(x_1(t)|\mu_2, \Psi(t)) p(\mu_2|\Psi(t)) \\ & = p(\mu_2|\Psi(t)) \int p(x_1(t)|\mu_1, \Psi(t)) p(\mu_1|\mu_2, \Psi(t)) d\mu_1 \end{aligned} \tag{4.12}$$

With the covariance known, the conditional distribution of $\mu_1$ on $\mu_2$ is

$$\begin{aligned} \mu_1|\mu_2 &\sim \mathcal{N}(\theta_1|\mu_2, \sigma_1^2|\mu_2) \\ \theta_1|\mu_2 &= \theta_1 + \frac{\sigma_{1,2}}{\sigma_2^2}\left(\mu_2 - \theta_2\right) \\ \sigma_1^2|\mu_2 &= \sigma_1^2 - \frac{\sigma_{1,2}^2}{\sigma_2^2} \end{aligned} \tag{4.13}$$

where $\sigma_{1,2}$ is the covariance of $\{\mu_1, \mu_2\}$. Substituting them into Equation 4.12 gives

$$\begin{aligned} \mu_2|x_1(t) &\sim \mathcal{N}(\theta_2(t+1), \sigma_2^2(t+1)) \\ \theta_2(t+1) &= \theta_2(t) + \sigma_{1,2}\frac{x_1(t) - \theta_1(t)}{\sigma_1^2(t) + \sigma_0^2} \\ \sigma_2^2(t+1) &= \sigma_2^2(t) - \frac{\sigma_{1,2}^2}{\sigma_1^2(t) + \sigma_0^2} \end{aligned} \tag{4.14}$$

Similarly substituting the posterior of $\mu_2$ from Equation 4.14 to Equation 4.11 the

expected payoff of the non-selected ad is obtained:

$$X_2|x_1(t), \Psi(t) \sim \mathcal{N}\left(\theta_2(t+1), \sigma_0^2 + \sigma_2^2(t+1)\right) \tag{4.15}$$

Note the correctness of the equation can be verified by setting the first and second ads equal – if $\theta_1 = \theta_2$ and $\sigma_1^2 = \sigma_2^2 = \sigma_{1,2}$, Equation 4.14 becomes exactly Equation 4.9. Therefore, Equation 4.14 is taken as the unified equation covering both self and correlated updates. The objective function in Equation 4.6 is now completed with the belief update formulas (constraints), all of which are now summarised together as the following

$$\pi^* = \arg\max_{\pi} \sum_{t=1}^{T} \theta_{s(t)}(t) \tag{4.16}$$

subject to

$$\theta_{s(t+1)}(t+1) = \theta_{s(t+1)}(t) + \sigma_{s(t),s(t+1)} \frac{x_{s(t)}(t) - \theta_{s(t)}(t)}{\sigma_{s(t)}^2(t) + \sigma_0^2} \tag{4.17}$$

$$\sigma_{s(t+1)}^2(t+1) = \sigma_{s(t+1)}^2(t) - \frac{\sigma_{s(t),s(t+1)}^2}{\sigma_{s(t)}^2(t) + \sigma_0^2} \tag{4.18}$$

It is worth noticing that the update in Equation 4.17 is closely related to the *word of mouth* heuristic adopted in collaborative filtering [137, 138]. In the collaborative filtering, particularly, the user-based ones, the rating of a target user is estimated by looking at other similar users; The more similar a user is, the more contribution he or she would have to the prediction. Using the heuristics, the final rating prediction is a weighted average across all similar users [138] and the similarity is usually measured by cosine similarity or Pearson's correlation coefficient and user means are used to remove the bias of mean ratings among users [137]. In this thesis, using a simple Gaussian model, a collaborative update mechanism across correlated ads is naturally derived. The major difference is that the update is in a sequential way. As seen in Equation 4.17, the similarity measure here is the correlation normalized by the variance, and $\theta_{s(t+1)}(t)$ and $\theta_{s(t)}(t)$ are used to remove the bias from the mean payoffs between different ads. Moreover, Equation 4.18 naturally provides the confidence of the predictions.

Optimising Equation 4.16 leads to the exploration and exploitation dilemma. The

publisher would like to earn more by selecting the best known $\theta_{s(t)}(t)$ so far. However, some ads with higher variances might potentially have higher payoffs. They also require to be selected in order to gather the feedback. Not selecting the local best may result in a loss of the immediate reward, but the loss, however, could be compensated if any better alternatives is found in later stages. Besides, the model deals with a changing dataset naturally: the new coming ads could be assigned high variances to encourage the exploration on them.

## 4.3 Solutions

This section provides both exact and approximate solutions to the proposed selection problem. A toy example is introduced to illustrate the benefit of using correlation.

### 4.3.1 Value Iteration

The revenue optimisation problem in Equation 4.16 could be solved exactly by following a value iteration approach using Dynamic Programming [139]. Recall that $\boldsymbol{\theta}$ and $\boldsymbol{\Sigma}$ denote the belief of $N$ ads' true payoffs before $t = 1$. Let $V^*(\boldsymbol{\theta}, \boldsymbol{\Sigma}, T)$ denote the max possible revenue the publisher could gain in $T$ time steps. The following statement and the Bellman equation [139] are obtained.

For any given priori $\boldsymbol{\theta}$ and $\boldsymbol{\Sigma}$, there exists an optimal policy $\pi^*$ to the problem in Equation 4.16, and $V^*(\boldsymbol{\theta}, \boldsymbol{\Sigma}, T)$ is achievable. More over $V^*(\boldsymbol{\theta}, \boldsymbol{\Sigma}, T)$ satisfy the following condition

$$\begin{aligned} & V^*\left(\boldsymbol{\theta}, \boldsymbol{\Sigma}, T\right) \\ & = \max_{s(1)\in N} \mathbb{E}\left[X_{s(t)}(1) + V^*\left(\boldsymbol{\theta}|X_{s(t)}(1), \boldsymbol{\Sigma}|X_{s(t)}(1), T-1\right)\right] \end{aligned} \tag{4.19}$$

By solving this equation recursively the optimal policy is found. If $T = 1$, the optimal revenue is

$$\begin{aligned} V^*(\boldsymbol{\theta}, \boldsymbol{\Sigma}, 1) &= \max_{s(1)\in N} \mathbb{E}\left[X_{s(1)}(1)\right] \\ &= \max_{s(1)\in N} \theta_{s(1)}(1) \end{aligned} \tag{4.20}$$

which indicates that the publisher should simply choose the ad with highest expected payoff. This choice is straightforward because no more time steps exist, thus no need

of exploration.

For $T = 2$ the optimal revenue is written as

$$\begin{aligned}
&V^*(\boldsymbol{\theta}, \boldsymbol{\Sigma}, 2) \\
&= \max_{s(1)\in N} \mathbb{E}\left[X_{s(1)}(1) + V^*\left(\boldsymbol{\theta}|X_{s(t)}(1), \boldsymbol{\Sigma}|X_{s(t)}(1), 1\right)\right] \\
&= \max_{s(1)\in N} \int p(x_{s(1)}) \left(x_{s(1)} + U^*(\boldsymbol{\theta}|x_{s(t)}(1), \boldsymbol{\Sigma}|x_{s(t)}(1), 1)\right) dx_{s(1)} \\
&= \max_{s(1)\in N} \int p(x_{s(1)}) \left(x_{s(1)} + \max_{s(2)\in N} \left(\theta_{s(2)}(2)\right)\right) dx_{s(1)} \\
&= \max_{s(1)\in N} \left(\theta_{s(1)}(1) + \int \max_{s(2)\in N} p(x_{s(1)})\theta_{s(2)}(2) dx_{s(1)}\right)
\end{aligned} \tag{4.21}$$

The difficulty lies in the last integral as it depends on the $\max$ operator. Using Chasles' Relation [140], it could be expanded to several regional integrals according to the random vector $\boldsymbol{\theta}(2)$. For instance, if the publisher has to choose from only two ads, and by solving $\theta_1(2) > \theta_2(2)$ the following answer is obtained

$$\theta_1(2) > \theta_2(2) \text{ when } x_i(1) > k \tag{4.22}$$

which indicates the publisher should choose the first ad when the observation from the first time step is bigger than some value $k$. Then the last integral of Equation 4.21 could be broken into two regional integrals with exact solution

$$\begin{aligned}
V^*(\boldsymbol{\theta}, \boldsymbol{\Sigma}, 2) = \max_{i\in 1,2} \Big(&\theta_1(1) + \int_{-\infty}^{k} p(x_i)\theta_2(2) dx_i \\
&+ \int_{k}^{+\infty} p(x_i)\theta_1(2) dx_i\Big)
\end{aligned} \tag{4.23}$$

The above equation could be easily extended to $N$ ads cases provided the solution of $N$ inequalities for vector $\boldsymbol{\theta}$, where the only variable considered as unknown is the observation from the last time step. For simplicity, let us define that a region is *dominated* by some ad when the ad should be selected if the observation falls in the region. This is similar with [125] where the value function is expressed as a linear combination of $\alpha$-functions. Formally the region dominated by $i$-th ad at time step $t$ is denoted as

$[m_{i,t}, n_{i,t}]$. For a general case of $N$ ads, Equation 4.21 is written as

$$V^*(\boldsymbol{\theta}, \boldsymbol{\Sigma}, 2) = \max_{s(1)\in N} \Big(\theta_{s(1)}(1) + \sum_{s(2)}^{N} \int_{m_{s(2),2}}^{n_{s(2),2}} p(x_i)\theta_{s(2)}(2)dx_i\Big) \tag{4.24}$$

where the regional integral could be solved as

$$\int_m^n xp(x)dx = -\sigma^2\Big(p(n) - p(m)\Big) + \mu\Big(\phi(n) - \phi(m)\Big) \tag{4.25}$$

where $\phi(x)$ is the c.d.f. for the Gaussian random variable $X$. Note that for some ads at some time steps, their dominant regions could be empty; simply indicating the ads would never be selected under such circumstances.

#### 4.3.1.1 A Toy Example

This section gives an example to demonstrate the sequential selection mechanism with embedded correlated belief update. Assume a publisher have 2 ads to select from. The Gaussian noise from users is given by

$$\sigma_0^2 = 0.1 \tag{4.26}$$

Random state $\boldsymbol{\mu}$ is defined by a bivariate Gaussian, i.e.

$$\boldsymbol{\mu} \sim \mathcal{N}(\boldsymbol{\theta}, \boldsymbol{\sigma}) \tag{4.27}$$

where

$$\boldsymbol{\theta} = \begin{bmatrix} 1 \\ \\ 0.95 \end{bmatrix} \quad \boldsymbol{\Sigma} = \begin{bmatrix} ten & 0.2 \\ \\ 0.2 & 50 \end{bmatrix} \tag{4.28}$$

Considering only one time step gives the expected revenue

$$\begin{aligned} V^*(\boldsymbol{\theta}, \boldsymbol{\Sigma}, 1) &= \max_{s(1)\in N} \mathbb{E}\big[X_{s(1)}(1)\big] \\ &= \max(1, 0.95) \\ &= 1 \ \text{ with } s(1) = 1 \end{aligned} \tag{4.29}$$

Now consider $T = 2$. Suppose the first ad at the first time step is selected, which yields the following update for time step $t = 2$

$$\begin{aligned} &\text{when } s(1) = 1 : \\ &\theta_1(2) = 1 - \frac{ten \times (1 - x_1(1))}{0.1 + ten} \\ &\theta_2(2) = 0.95 - \frac{0.2 \times (1 - x_1(1))}{0.1 + ten} \end{aligned} \tag{4.30}$$

which gives

$$\theta_1(2) > \theta_2(2) \text{ when } x_1(1) > 0.95$$

Thus, the optimal reward for $t = 2$ when choosing $s(1) = 1$ could be derived as

$$\begin{aligned} &V^*_{s(1)=1}(\boldsymbol{\theta}, \boldsymbol{\Sigma}, 2) \\ =&1 + \int_{0.95}^{+\infty} p(x_1)\theta_1(2)dx_1 + \int_{-\infty}^{0.95} p(x_1)\theta_2(2)dx_1 \\ =&3.1993 \end{aligned} \tag{4.31}$$

A small defect is that variable $x$ stands for the payoff observed from the first time step, and it cannot be smaller than zero. However, due to the Gaussian assumption, when calculating $V^*$, $x$ is integrated over the Real space and sometime results in a negative expected reward. However, such policy would vanish in later comparisons due to its low value and has little effect on our decision process.

Similarly selecting the second ad at the first time step yields the following update for step two

$$\begin{aligned} &\text{when } s(1) = 2 : \\ &\theta_1(2) = 1 - \frac{0.2 \times (0.95 - x_2(1))}{0.1 + 50} \\ &\theta_2(2) = 0.95 - \frac{50 \times (0.95 - x_2(1))}{0.1 + 50} \end{aligned} \tag{4.32}$$

which gives

$$\theta_1(2) > \theta_2(2) \text{ when } x_2(1) < 1.00 \tag{4.33}$$

Thus

$$\begin{aligned} & V^*_{i=2}(\boldsymbol{\theta}, \boldsymbol{\Sigma}, 2) \\ =& 0.95 + \int_{-\infty}^{1} p(x_2)\theta_1(2)dx_2 + \int_{1}^{+\infty} p(x_2)\theta_2(2)dx_2 \\ =& 4.7291 \end{aligned} \tag{4.34}$$

Finally the maximum expected payoff over two time steps is

$$V^*(\boldsymbol{\theta}, \boldsymbol{\Sigma}, 2) = \max(3.1993, 4.7291) = 4.7291 \tag{4.35}$$

and the corresponding optimal policy is

$$\pi^* = \{s(1) = 2, s(2) = 1 \text{ if } x_1(1) < 1, s(2) = 2 \text{ otherwise}\} \tag{4.36}$$

The result shows that the optimal policy is clearly different from a myopic one, which tries to maximise the immediate reward only. Following the myopic policy, the publisher would choose $s(1) = 1$ and receive a smaller payoff. One of the reasons of selecting the second is because it has a higher variance (taking into account the correlations as well). In the experiment section, the real-world data suggests that high variances are quite common.

### 4.3.2 Approximate Solution

To understand the approximations let us rewrite the objection function as a combination of expected immediate reward and exploration function, which utilizes the available information up to the decision time step, denoted by $\xi(\Psi(t), i)$ for the $i$-th candidate. The decisions maximise some the objective value function $V_{s(t)}(\Psi(t))$, i.e.,

$$\begin{aligned} s(t) &= \arg\max_{s(t) \in N} V_{s(t)}(\Psi(t)) \\ &= \arg\max_{i \in N} \left(\bar{x}_i + \xi(\Psi(t), i)\right) \end{aligned} \tag{4.37}$$

**Algorithm 1** The VI-COR algorithm using value iteration with Monte Carlo sampling.

**function** VALUEFUNC($\boldsymbol{\theta}, \boldsymbol{\Sigma}, t$)
  array $V \leftarrow 0$ ▷ Expected reward vector.
  **loop** $i \leftarrow 1$ to $N$
    $V[i] \leftarrow \theta_i(t)$ ▷ Expected immediate reward.
    **if** $t < T$ **then**
      **for all** $s$ in SAMPLE($\boldsymbol{\theta}, \boldsymbol{\Sigma}$) **do**
        $[\boldsymbol{\theta}', \boldsymbol{\Sigma}'] \leftarrow$ UPDATEBELIEF($\boldsymbol{\theta}, \boldsymbol{\Sigma}, s, i$)
          ▷ New belief after selecting $i$ and observing $s$.
          ▷ Equations 4.17 and 4.18.
        $V[i] \leftarrow V[i] + \frac{1}{M_0}$VALUEFUNCTION($\boldsymbol{\theta}', \boldsymbol{\Sigma}', t+1$)
      **end for**
    **end if**
  **end loop**
  **return** [MAX($V$), MAXINDEX($V$)]
**end function**

**Algorithm 2** The UCB1-NORMAL-COR algorithm using multi-armed bandit with correlated update.

**function** PLAN($\boldsymbol{\theta}, \boldsymbol{\Sigma}, \Psi(t)$)
  array $V \leftarrow 0$
  **loop** $i \leftarrow 1$ to $N$
    **if** $t_i < \lceil 8 \log t \rceil$ **then**
      **return** i
    **end if**
  **end loop**
  $[\boldsymbol{\theta}', \boldsymbol{\Sigma}'] \leftarrow$ UPDATEBELIEF($\boldsymbol{\theta}, \boldsymbol{\Sigma}, \Psi(t)$)
    ▷ New belief of all ads with all available information.
    ▷ Equations 4.17 and 4.18.
  **loop** $i \leftarrow 1$ to $N$
    $V[i] \leftarrow \theta'_i + \sqrt{16 \cdot \frac{q_i - t_i \theta'^2_i}{t_i - 1} \cdot \frac{t-1}{t_i}}$
      ▷ Expected reward.
  **end loop**
  **return** [MAX($V$), MAXINDEX($V$)]
**end function**

For example, the exploration function in Equation 4.19 is

$$\xi = \int p(x_{s(1)}) V^* \left( \boldsymbol{\theta} | x_{s(t)}(1), \boldsymbol{\Sigma} | x_{s(t)}(1), T-1 \right) dx \tag{4.38}$$

of which the computation is expensive due to recursive calling and integral. The following of this section presents two approximate methods.

#### 4.3.2.1 Value Iteration with Monte Carlo Sampling

For $T \geq 3$ the solution for $N$ inequalities cannot be obtained easily. Instead Monte Carlo sampling is used to deal with the integral and avoid solving the inequalities. The exploration function is written as

$$\xi_{\text{VI-COR}} = \frac{1}{M_0} \sum_{x \in \mathbb{S}} p(x) V^* \left(\boldsymbol{\theta} | x, \boldsymbol{\Sigma} | x, T-1\right) dx \tag{4.39}$$

where $\mathbb{S}$ is the sample set and $M_0$ is the normalizing factor from sampling. The algorithm for a general $T \geq 3$ case is represented in Algorithm 1. This algorithm is named VI-COR in our experiments.

#### 4.3.2.2 The UCB1-Normal-COR Algorithm

A problem to the above value iteration solution (with or without the MC sampling) is the computational complexity. The run time of the algorithms is actual of $O(N^T)$ where $N$ is the number of candidates and $T$ is the planning horizon. Usually in real-world such complexity is unacceptable. In order to tackle this challenge this section introduces a multi-armed bandit based approximation.

The multi-armed bandit is a popular model dealing with exploration-exploitation dilemma in sequential optimisation problems [141]. Similar to the problem discussed above, in the multi-armed bandit scenario a player must decide which arm to play at each time step to maximise the cumulative reward over the entire planning horizon.

Most multi-armed bandit algorithms reduce the computational cost by approximating the exploration function. This thesis improve the performance of a deterministic policy UCB1-NORMAL [141] by adding the correlated update. The original algorithm assumes the Gaussian distribution of the reward, independence between arms, and underlying mean and variance for reward distribution are unknown but fixed. The exploration function of UCB1-NORMAL is written as

$$\xi_{\text{UCB1-NORMAL}} = \sqrt{16 \cdot \frac{q_i - t_i \theta_i^2(t)}{t_i - 1} \cdot \frac{t-1}{t_i}} \tag{4.40}$$

where $q_i$ is the sum of squared reward obtained from $i$-th arm, and $t_i$ the times $i$-th has been played so far. The algorithm is extended for by adding the correlated updates. The exploration function would remain the same form, but $\boldsymbol{\theta}(t)$ at each time step is updated

according to Equations 4.17 and 4.18, instead of only updating the selected candidate. The algorithm, referred to as UCB1-NORMAL-COR in the experiments, is represented in Algorithm 2.

## 4.4 Experiments and Results

This section describes a real-world dataset collected from Google over a six months period, then compare various algorithms mentioned before on this dataset. Some interesting findings are reported in the end.

### 4.4.1 Dataset

The dataset was collected from Google AdWords Keyword Planner service [142]. Let us consider the scenario that advertisers deploy campaigns through an advertiser (demand) side platform, whereas online publishers retrieve ads and earn revenue from a related supply side platform. Generally, online publishers share the ad revenues with their chosen ad networks or exchanges with a fixed percentage. For instance, online publishers with Google AdSense gain 68% of advertisers' spending, and the ratio has remained the same since 2003 [143]. Thus, it is reasonable to consider online publishers' ad revenue to be proportional to the cost of the advertisers.

The test data was collected from 12/2011 to 5/2012. The Google AdWords Keyword Planner service [142] provides nearly real-time data to help advertisers to adjust budgets and select appropriate keywords. Given a keyword, budget, and targeting rules, the service returns a list of fields including daily clicks, global and local impressions, average position (of the ad list which usually has five to eight ads), average CPC, and total cost. In addition, the US and UK markets are separated using the geographical targeting option. Estimation of keywords' stats were collected across the Google Sponsored Search and Display Networks. During the collection period, 521 different keywords from various categories were used, and 310 of them have non-zero mean payoffs. The mean and variance of all keywords are plotted in Figure 4.2, 4.3, and 4.4. As shown in Table 4.3, keywords are clustered into eight categories. In fact, it is not necessary for publishers to try out all the available ads. Instead, they should specify their target categories (based on hosting webpages content) and make optimal decisions within the targeted categories. In the experiments, ads associating with the collected

**Table 4.1:** The sample mean and variance of keyword prices in *People & Organization* dataset. It clearly shows that some candidates have lower mean but very high variance, implying short and strong bursts due to commercial advertising campaigns deployed by advertisers.

| **Ad candidate** | **Sample Mean** | **Sample Variance** |
|---|---:|---:|
| selena gomez | 9.69 | 13567.30 |
| eminem | 22.44 | 74386.88 |
| michael jackson | 23.58 | 3003.34 |
| justin bieber | 27.70 | 27412.89 |
| wayne rooney | 31.06 | 350.87 |

**Table 4.2:** The sample correlation matrix for *People & Organization* category. The high correlations made the UCB1 and UCB1-Normal inefficient.

| | eminem | justin bieber | michael jackson | selena gomez | wayne rooney |
|---|---:|---:|---:|---:|---:|
| eminem | 1.00 | -0.43 | -0.58 | -0.50 | -0.73 |
| justin bieber | - | 1.00 | 0.80 | 0.74 | 0.70 |
| michael jackson | - | - | 1.00 | 0.97 | 0.71 |
| selena gomez | - | - | - | 1.00 | 0.63 |
| wayne rooney | - | - | - | - | 1.00 |

keywords are considered as candidates and decision making is on a daily basis.

### 4.4.2 Baselines and Experiments

The following policies have been compared in the experiments:

- RANDOM policy, which selects candidates randomly (uniform);
- MYOPIC policy, which selects candidates based on expected immediate reward;
- UCB1 policy, which assumes independent between candidates and is model-free of reward distribution [141]; and
- UCB1-NORMAL policy, which assumes independent between candidates and the reward following Gaussian distribution;

And the proposed algorithms:

- VI-COR policy, which uses value iteration with Monte Carlo sampling (Algorithm 1); and

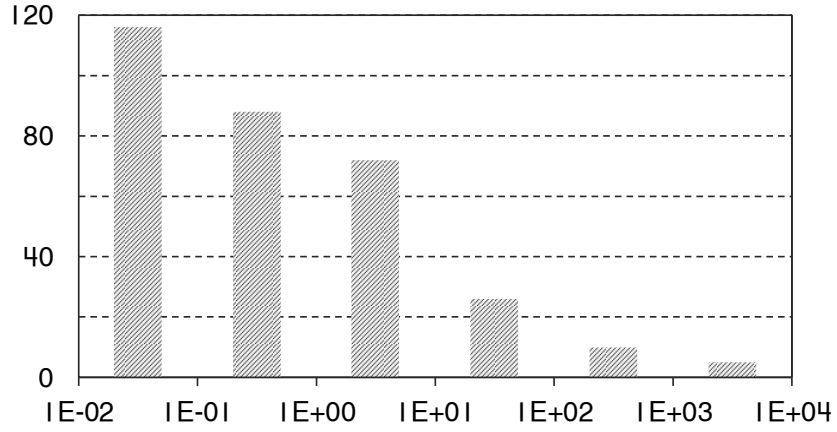


**Figure 4.2:** The histogram of expected payoff of all candidate keywords.

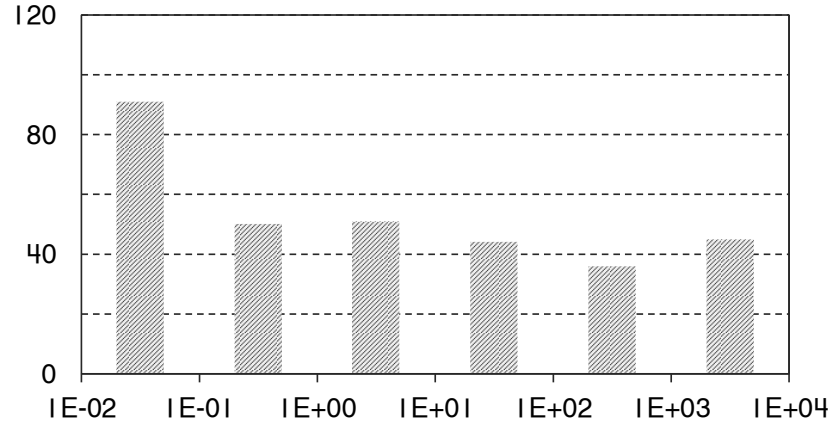


**Figure 4.3:** The histogram of variance of payoff of all candidate keywords.

- UCB1-NORMAL-COR policy, which consider the correlation of candidates (Algorithm 2).

For each keyword, there are about $150$ daily payoff data points (some keywords have less due to a later start of collecting). For each category (except *Uncategorised*) keywords are selected with close mean payoffs to form a dataset. In order to test the statistical significance of algorithms performance, the daily payoff time series were divided into eight chunks for each dataset (with overlap). For each chunk, $20\%$ was used as the training set to get the prior belief of ad performances, i.e., $\boldsymbol{\theta}(0)$ and $\boldsymbol{\Sigma}(0)$. Then the remaining $80\%$ was run with each algorithm reporting the cumulative reward at each time step. Besides, the averaged results are reported with Wilcoxon signed-rank test [144] for the significance of the best algorithm outperforming the second best within each dataset.

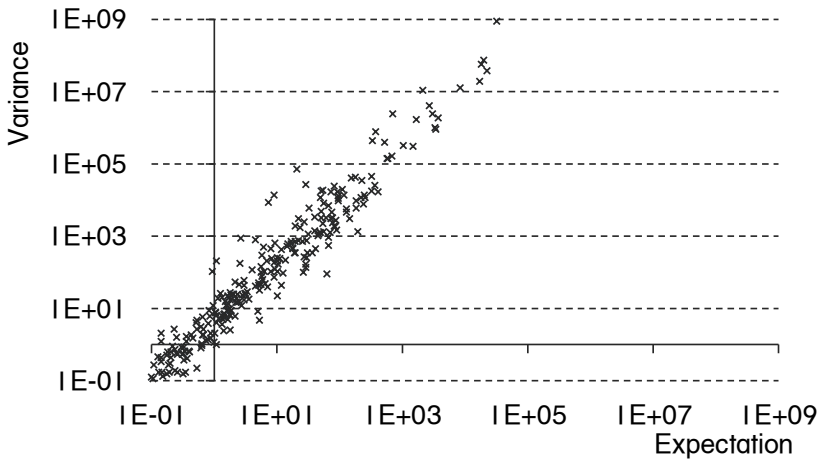


**Figure 4.4:** The scatter plot of mean and variance of payoff of all candidate keywords.

**Table 4.3:** Categorisation of the collected 310 keywords with non-zero mean payoffs.

| **Category** | **Count** | **Category** | **Count** |
|---|---|---|---|
| Education | 34 | Person & organization | 29 |
| Shopping | 86 | (Personal) finance | 51 |
| Product & service | 20 | Information | 52 |
| Medical | 25 | Uncategorised | 13 |

For each cluster candidates were chosen with the similar expectation, because when the payoff differs too much the problem becomes trivial. For instance, in Education category the highest sample mean is $56.94$ for *android app developer* while the lowest is $0.29$ for *training indesign*. Publisher would select the first one without any need of exploration, resulting in little discrimination of tested algorithms. Therefore, candidates were chosen with relatively close means to emphasise exploration, for instance, *People & Organization* dataset listed in Table 4.1 and Table 4.2, which is genuine in real-world provided that available ads are way more than $310$ tested here.

### 4.4.3 Results and Discussions

The proposed two algorithms are compared with the others across the ten categories. In order to compare performances across categories, the cumulative revenues have been normalised against the GOLDEN solution (i.e., an oracle that always makes the best decision). The results are summarized in Table 4.4 and are compared in Figure 4.5 and 4.6. They show that, within the ten different datasets, the proposed VI-COR algorithm

**Table 4.4:** The overall performance comparison. The cumulative payoffs are averaged on 8 chunks then normalized w.r.t. the GOLDEN policy for a better representation. In each row the one with highest cumulative payoff is in bold and with an $*$ if the difference with the second best is significant.

| **Datasets** | MYOPIC | RANDOM | UCB1 | UCB1-NORMAL | VI-COR | UCB1-NORMAL-COR |
|---|---|---|---|---|---|---|
| Education | 21.9 | 23.0 | 30.9 | 30.9 | **41.2*** | 27.6 |
| Finance-1 | 38.5 | 27.8 | 40.9 | 26.4 | **44.5** | 27.4 |
| Finance-2 | 22.1 | 16.5 | 30.6 | 22.8 | **38.0*** | 22.9 |
| Information | 14.1 | 12.9 | 27.8 | 15.9 | **29.4** | 15.9 |
| People & Organization | 41.6 | 30.4 | 50.5 | 31.4 | **72.9*** | 63.3 |
| Shopping-1 | 17.4 | ten.6 | **42.3** | 16.1 | 40.2 | 16.4 |
| Shopping-2 | 29.9 | 14.5 | 34.3 | 75.3 | 52.9 | **79.2*** |
| Shopping-3 | 9.7 | 4.3 | 21.9 | 18.3 | **27.3** | 19.4 |
| Product & Service | 24.7 | 26.0 | 47.2 | 57.1 | **67.9*** | 59.9 |
| Medical | 30.5 | 19.6 | 52.7 | 32.2 | **58.0*** | 33.5 |

performed the best for 8/ten with 5/8 significantly better. The UCB1-NORMAL-COR algorithm performed the best for 1/ten and was significantly better in that trail. With the *Shopping-1* dataset, the UCB1 algorithm performed the best, but the VI-COR had the comparable performance, and the difference was not significant.

In Figure 4.5 and 4.6 the daily performance comparison is given on *Education* and *People & Organization* datasets where algorithm runs on entire payoff data series. The interesting findings are discussed in the following sections.

#### 4.4.3.1 The Importance of Exploration

First let us study the importance of exploration. Figure 4.5 and 4.6 compares the daily accumulated revenues over time. As illustrated in Figure 4.5, in the beginning (between day-0 and day-10) the MYOPIC policy achieved an excellent result, and its cumulative payoff was the best until day-65. This is explained by the fact that there is no exploration involved in this policy and it exploits the current belief directly. However, the proposed algorithms with exploration quickly outperform it in the late stage as more profitable ads have been discovered from the exploration from the early stage. In the end, the MYOPIC policy fails to win due to no exploration in the beginning, and later stuck to suboptimal ads. It is similar in Figure 4.6 where the MYOPIC policy outperforms others between day-25 and day-35, but is caught up and passed very soon. These conclude that the exploration is valuable and necessary in the ad selection task. Note

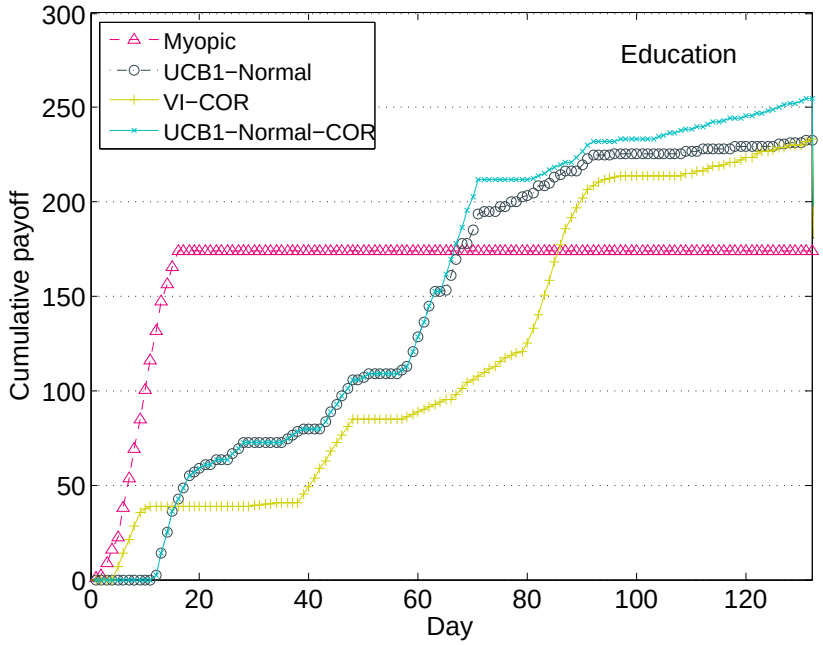


**Figure 4.5:** Comparing algorithms with correlated update with their baselines on *Education* which has nine keywords.

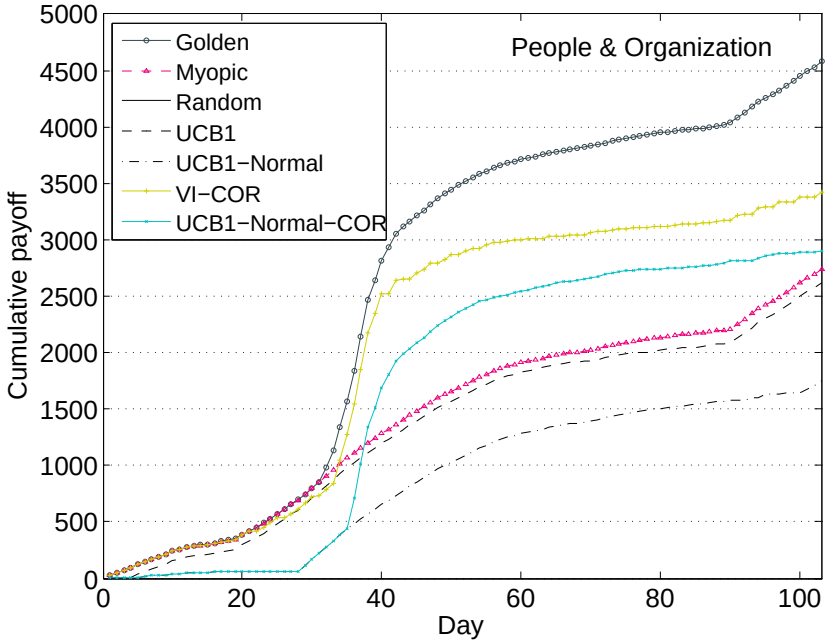


**Figure 4.6:** The performance of algorithms on *People & organization* which has five keywords.

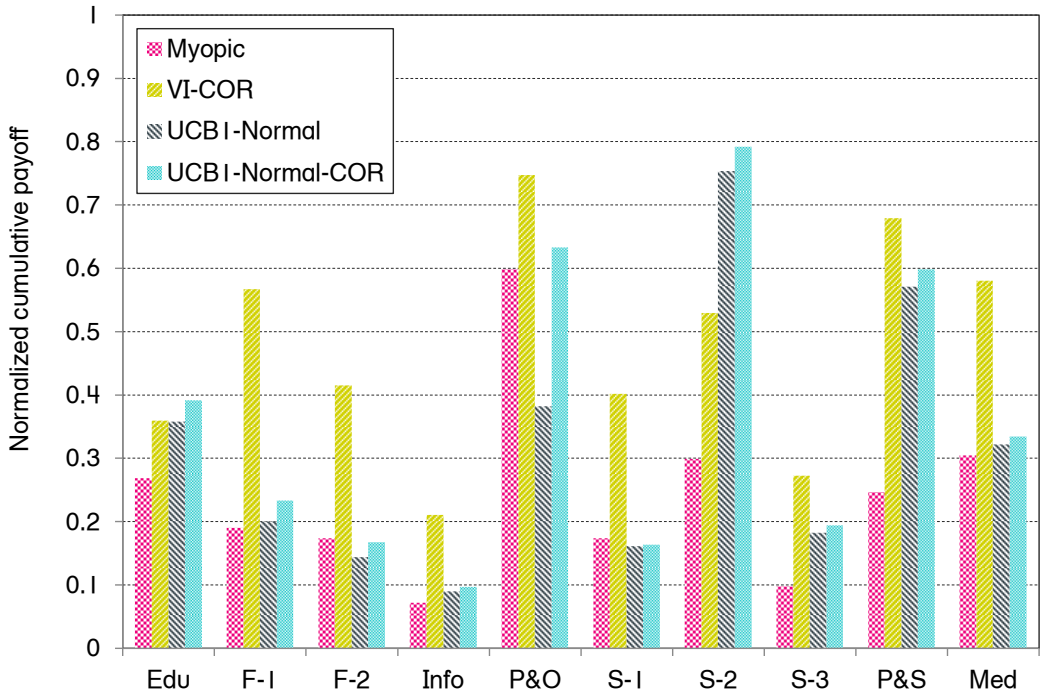


**Figure 4.7:** Comparing accumulated payoffs on all ten categories. VI-COR always performed better than MYOPIC and UCB1-NORMAL-COR always performed better than UCB1-NORMAL across all datasets. The results prove the benefit of correlated update.

that only *Education* and *People & organization* are illustrated, while the algorithms behaved consistently across all categories.

#### 4.4.3.2 The Importance of Correlation

Recall that a Gaussian noise constant is used for all webpages and users. The consequence is that $X_1$ and $X_2$ are conditional independent when $\mu_1$ and $\mu_2$ are known. The relationship between covariances of $X1$ and $X_2$ and that of $\mu_1$ and $\mu_2$ can be further derived as,

$$\begin{aligned} \text{Cov}[X_1, X_2] &= \mathbb{E}\big[\,\text{Cov}[X_1, X_2|\mu_1, \mu_2]\big] \\ &\quad + \text{Cov}\left[\mathbb{E}[X_1|\mu_1, \mu_2], \mathbb{E}[X_2|\mu_1, \mu_2]\right] \\ &= \text{Cov}[\mu_1, \mu_2] \end{aligned} \tag{4.41}$$

which enables publishers to use either of correlations within the model. In experiments $\text{Cov}[X_1, X_2]$ is used.

Like most multi-armed bandit models, the UCB1 algorithm assumes independence between candidates. Therefore, when candidates have relatively low correlations, e.g.,

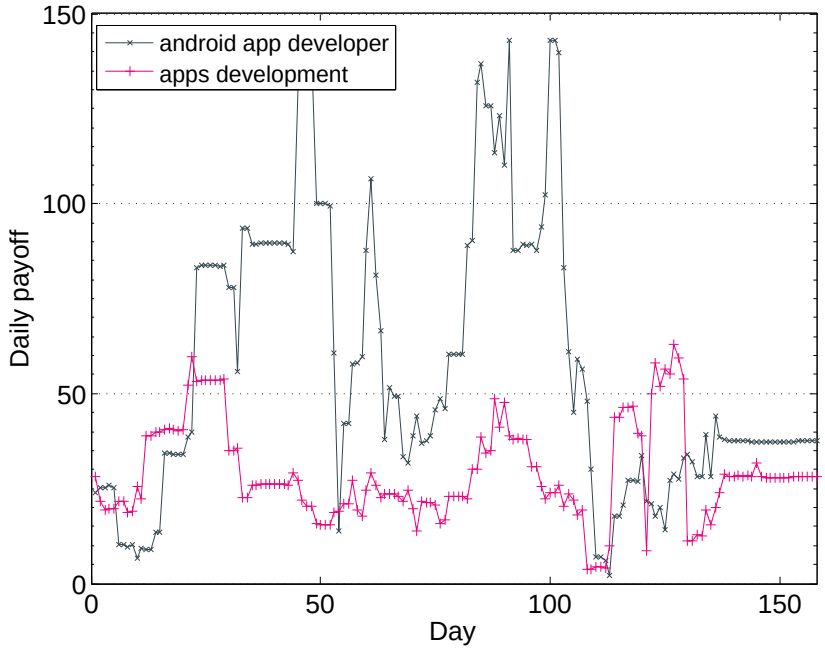


**Figure 4.8:** The daily payoffs of two mobile development related ad candidates.

in *Shopping-1*, the algorithm would perform well. This is confirmed by the observation that the UCB1-NORMAL-COR algorithm reports only 0.3% improvement over UCB1-NORMAL on that dataset. However, when the correlation of ads are high such as *People & Organization* (a sample correlation matrix is shown in Table 4.2), the proposed algorithms show better results. For instance the VI-COR algorithm shows 22.4% improvement over the UCB1 and the UCB1-NORMAL-COR shows 31.9% improvement over the UCB1-NORMAL on that dataset. In Figure 4.6 it is clear that UCB1-NORMAL-COR discovers the better options much quicker than UCB1-NORMAL. The same conclusion is obtained by comparing VI-COR with MYOPIC. Figure 4.7 shows significant improvement by utilising correlation of ads. The maximum uplift was obtained in *Product & service* (43.3%), and on average it was 22.2% across all experiments.

#### 4.4.3.3 The Impact of the Noise Factor $\sigma_0^2$

The introduction of the noise factor $\sigma_0^2$ is an essential part of proposed models. On one hand, it helps to capture the uncertainty which is unforeseeable and has not been properly modelled by the underlying $\boldsymbol{\theta}$ and $\boldsymbol{\Sigma}$ as discussed before. The assumption about the Gaussian distribution may not be accurate in practice. This can be seen from Figure 4.8 and Figure 4.9. The noise factor provides the flexibility of tuning proposed algorithms towards particular situations. In the experiments, the noise factor

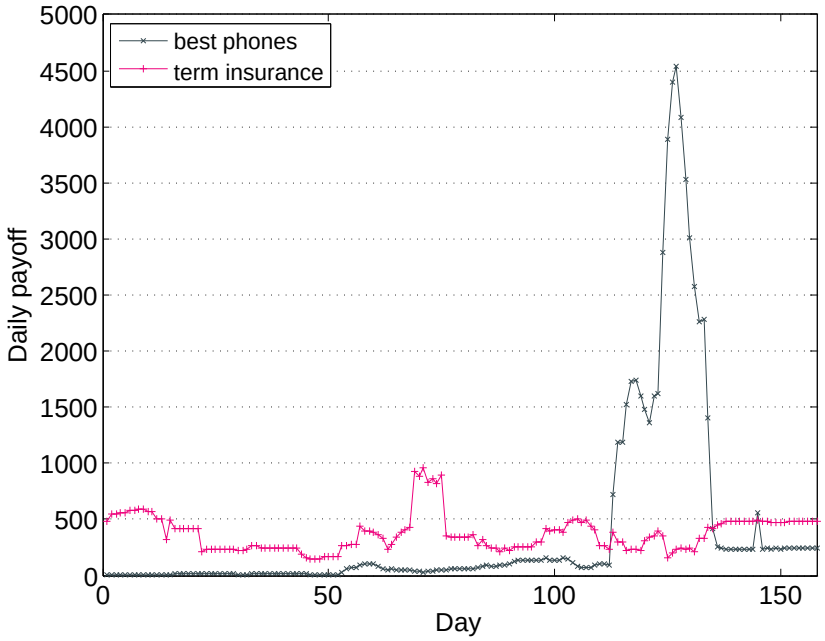


**Figure 4.9:** The daily payoff of two candidates with a sudden change.

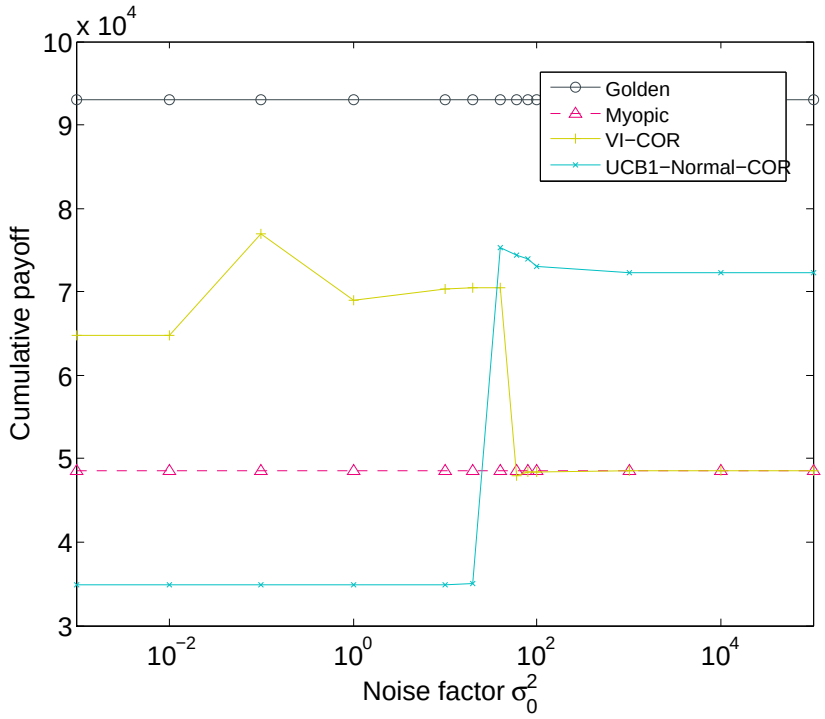


**Figure 4.10:** The impact of the noise factor $\sigma_0^2$ for the situation in Figure 4.9.

was obtained by tuning using training datasets (20% of all data points).

On the other hand, the noise factor is also a control of the sensitivity of the algorithms towards the unexpected daily payoff fluctuation. Equations 4.17 and 4.18 show that the noise factor is in the denominator, and a high noise factor leads to a steady

policy while a low one leads to a highly responsive (sensitive) policy. A smaller value, for example, $\sigma_0^2 = 0.01$, would make the algorithm switch probably too much, whereas a larger value would not be able to capture the fluctuation of the data. The dataset suggests that sometimes there could be sharp bursts, led by commercial activities. For instance, at the beginning of May, 2012 the *Sumsang Galaxy S III* and *Nokia 808 PureView* were launched for pre-ordering or purchasing, and both claimed to be the 'best' on the market. The competition of commercial campaigns cause the daily payoff of *best phones* became very high between 15/04/2012 and 05/05/2012 (Figure 4.9). In order to response to such a short and strong abnormal activity a small noise factor is required. Figure 4.10 shows that with $\sigma_0^2 \leq 40$ the VI-COR algorithm is able to identify and switch to *best phones* when the burst happened; but with $\sigma_0^2 \geq 60$ the algorithm is not able to switch, resulting in a loss of payoff.

It is worth noticing that, as illustrated in Figure 4.10, the setting of the noise factor is algorithm-dependent as well. UCB1-NORMAL-COR requires a large noise value to deal with bursts — the best cumulative payoff was obtained at $\sigma_0^2 = 40$, and as the value of the noise factor decreased the performance dropped greatly. The different behaviour of two algorithms is due to the different structures of the exploration function. As shown in Equation 4.40, the exploration function of the UCB1-NORMAL-COR contains the squared expectation of the payoffs in the past, indicating that the candidate with a history of low payoffs would not be favoured especially with a sudden burst. Using a high noise value would increase the chance of selecting and exploring such candidates.

## 4.5 Adaptive Keywords Extraction

In some cases, publishers could retrieve different ads from the same source by using different keywords. This creates an additional problem after selecting an ad source: to explore and exploit keywords that generate the most revenue. The proposed system was created to accomplish a real-world task (keyword extraction for parked domains), however, the idea could be easily adopted by any general keyword extraction from various sources.

From the publishers' perspective, the process of extracting keywords is illustrated in Figure 4.12. It is an iterative process of selecting candidates and update the model according to the feedback. This task of extracting keywords against user feedback

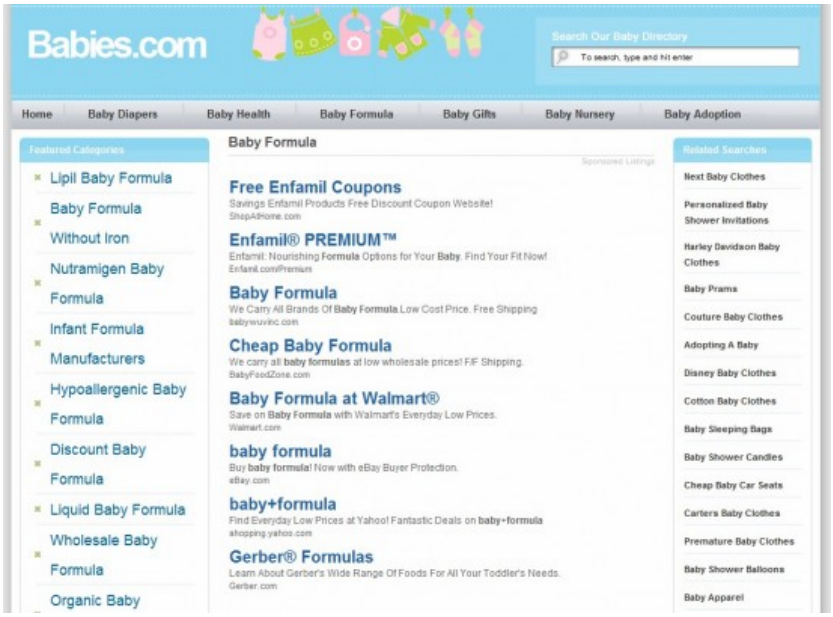


**Figure 4.11:** An example of advertising on parked domains. Instead of showing an error or *under construction*, relevant ads are displayed.

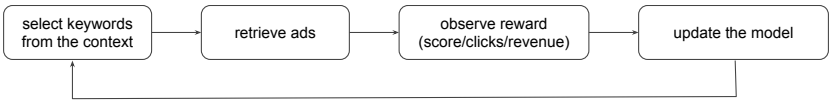


**Figure 4.12:** The keyword extraction task could be an iterative process. The publisher would like to exploit current optimal keywords as well as to explore potentially better ones.

directly links to the multi-armed bandits problem. The linear multi-armed bandits [145, 146, 147, 148] used in this thesis is a special case of (contextual) bandits. It has the same setting with standard bandits except the availability of the side information. The side information of each arm determines the reward for pulling the arm, therefore, could be used to make decisions. This section mainly exploits text features from webpages under domains being evaluated, including TF-IDF scores in title, content, keyword and description in HTML meta tag, header, anchor text of in-links, and part of speech.

First of all let us define reward of iteration step $t \in [0, T]$ as a real number in [0,1],

$$r(t) \in [0, 1] \tag{4.42}$$

The reward could be in various form, for example,

- Relevance scores of selected keywords against the webpage;
- The click-through rates (CTR) of ads, or

- Profit gained in each iteration.

Apparently the relevance scores, CTRs and profit are linked, however not guaranteed with any linear/non-linear relationship.

This section uses human judged relevance score for simplicity, also due to the difficulty of acquiring necessary data of CTR and profit. Besides, the crowdsourcing approach is employed to get the feedback in a cheap and fast way. Each webpage is considered a K-armed bandit machine and every arm a word (or multi-word phrase). In each iteration, an arm will be pulled resulting in the selection of the corresponding word/phrase to be the keywords of the webpage. Different from standard bandits, each arm ($i \in K$) of the linear bandits is associated with some $D$-dimension feature vector $\mathbf{x_i} \in \mathbb{R}^{D\times 1}$ which is already known. The expected reward (user feedback) is given by the inner product of its feature vector $\mathbf{x_i}$ and some fixed, but initially unknown parameter (column) vector $\mathbf{w}$. That is, the reward is a linear function of the feature vector and unknown parameters,

$$r_i = \mathbf{x_i}' \cdot \mathbf{w} \tag{4.43}$$

The goal here is to get the optimal reward. The proposed solution is the LINREL algorithm [148] highlighting upper confidence bound (UCB) to deal with the exploration-exploitation dilemma.

### 4.5.1 Upper Confidence Bound Algorithm

The idea of LINREL algorithm is to estimate the reward for the $i$-th arm from the linear combination of the historical reward received, with a high probability. Here the LINREL algorithm is expanded to help the thesis self-contained. First let us write the feature vector of the $i$-th arm as the linear combination of the previous chosen feature vectors (this is always possible except for the initial $d$ iteration steps),

$$\mathbf{x}_i(t) = \mathbf{X}(t) \cdot \mathbf{a}_i(t) \tag{4.44}$$

where $\mathbf{a}_i(t) \in \mathbb{R}^{t\times 1}$ is the coefficient of the linear combination and $\mathbf{X}(t)$ is the feature matrix selected in past iteration steps with dimension $D \times t$,

$$\mathbf{X}(t) = [\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(t-1)] \tag{4.45}$$

where the $\mathbf{x}(t)$ denotes the feature vector used in $t$ iteration step (the decisions are known but the subscripts are dropped for simplicity). With the same coefficient the reward could be written as,

$$r_i(t) = \mathbf{x}_i(t)' \cdot \mathbf{w} = (\mathbf{X}(t) \cdot \mathbf{a}_i(t))' \cdot \mathbf{w} = \mathbf{R}(t)' \cdot \mathbf{a}_i(t) \tag{4.46}$$

where $\mathbf{R}(t) \in \mathbb{R}^{t\times 1}$ is a vector of historical reward,

$$\mathbf{R}(t) = [r(1), r(2), \ldots, r(t-1)]' \tag{4.47}$$

This gives a good estimate $\mathbf{R}(t) \cdot \mathbf{a}_i(t)$ for $r_i(t)$. The algorithm keeps the variance small to maintain a narrow confidence interval of the estimate. By assuming i.i.d. of $r_i(t)$ (which is true for the keywords extraction task) and since $r_i(t) \in [0, 1]$ the variance of this estimate is bounded by $\|\mathbf{a}_i(t)\|^2/4$.

To get $\mathbf{a}_i(t)$, let us calculate the eigenvalue decomposition,

$$\mathbf{X}(t) \cdot \mathbf{X}(t)' = \mathbf{U}(t)' \cdot \mathbf{\Delta}[\lambda_1, \lambda_2, \ldots, \lambda_d] \cdot \mathbf{U}(t) \tag{4.48}$$

where $\mathbf{U}(t)$ is a unitary matrix and $\lambda_1, \ldots, \lambda_k \geq 1$ and $\lambda_{k+1}, \ldots, \lambda_d < 1$. Then for each feature vector $\mathbf{x}_i(t)$ write,

$$\mathbf{z}_i(t) = \mathbf{U}(t) \cdot \mathbf{x}_i(t) = [x_{i,1}(t), x_{i,2}(t), \ldots, x_{i,d}(t)]' \tag{4.49}$$

$$\mathbf{u}_i(t) = [x_{i,1}(t), \ldots, x_{i,k}(t), 0, \ldots]' \tag{4.50}$$

$$\mathbf{v}_i(t) = [0, \ldots, x_{i,k+1}(t), \ldots, x_{i,d}(t)]' \tag{4.51}$$

The coefficient $\mathbf{a}_i(t)$ is calculated as,

$$\mathbf{a}_i(t) = \mathbf{u}_i(t)' \cdot \mathbf{\Delta}[\frac{1}{\lambda_1}, \cdots, \frac{1}{\lambda_k}, \cdots, 0] \cdot \mathbf{U}(t) \cdot \mathbf{X}(t) \tag{4.52}$$

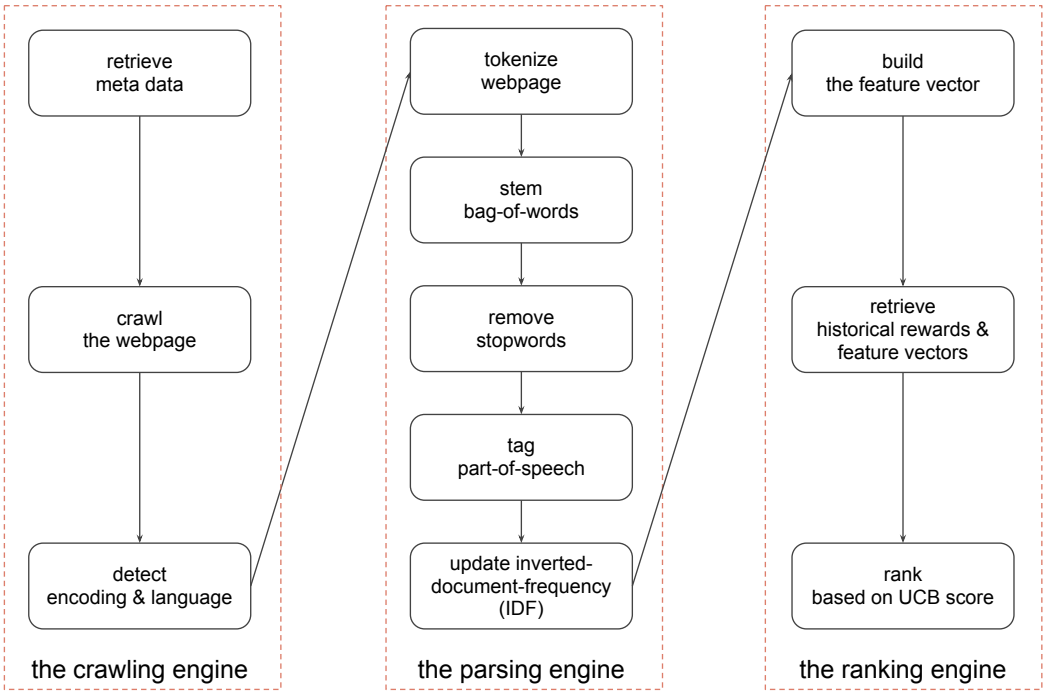


**Figure 4.13:** The architecture of the RAZORCLAW prototype system. The system is divided into three main parts. In every part, there are loosely bounded modules with each for a single task. These modules could be changed or updated flexibly. This thesis intents to create an open framework for the keywords extraction task.

Then with probability of $1 - \delta/T$ the expected reward of $i$-th arm is,

$$ucb_i(t) = \mathbf{R}(t) \cdot \mathbf{a}_i(t) + \sigma \tag{4.53}$$

where $\sigma$ is the width of the confidence bounds by using Azuma-Hoeffding bound [149],

$$\sigma = \|a_i(t)\|(\sqrt{ln(2TK/\delta)}) + \|v_i(t)\| \tag{4.54}$$

where $\delta$ is used to confine a narrow confidence interval and recall $K$ is the number of arms. The arm with highest $ucb_i(t)$ score will be selected. Apparently the arm got selected because the combination of its expected reward ($\mathbf{R}(t) \cdot \mathbf{a}_i(t)$) or potential reward ($\sigma$) is high. The former leads to exploitation and the latter results in exploration.

### 4.5.2 Prototype System

A prototype system RAZORCLAW was built according to the model discussed above. The system is implemented in Java. Its architecture is illustrated in Figure 4.13. The prototype system is made of three main parts: a crawler, a parser, and a ranker. In the

parsing engine, different modules will be used for different languages.

In the crawling module, the RAZORCLAW first loads meta information from the parked domain database, including the anchor texts and referrer URLs collected from the Internet. The webpage is crawled, too. The next step is to detect the correct encoding of the text (sometimes not presented in HTML) and the language. The language detector used in the system[2] is based on the naive Bayesian filter and has reported 99% precision over 49 languages.

Once the encoding and language are detected the content is converted to UTF-8 and invoke corresponding NLP processor. OpenNLP[3] was used for major western languages and IKAnalyzer[4] for Chinese-Japanese-Korean-Vietnam languages. The language support is not the main point of the system; however, it processes more than 50 languages. This is especially useful for free domain services, where a good amount of registrations are found from south east Asia, India, and Arabic countries.

After stemming, removing stopwords, tagging part-of-speech (POS), and saving the inverted-document-frequency (IDF), each phrase in the bag-of-words is processed to generate its local feature vector and to retrieve historical features and rewards. According to the ranking result, the system then gives the required number of keywords.

### 4.5.3 Datasets and Competing Algorithms

The experiments used 165 domains from DOT.TK's database and collected all possible side information. The proposed prototype system was compared with BM25F algorithm [150], KEA algorithm[5] [151], Yahoo! Term Extraction service[6] and Google Targeting Idea service[7].

The parameters of BM25F algorithm were obtained from the work of [150]. KEA is a famous keywords extraction tool in academic research. It is capable of extraction keywords with domain-specific knowledge, for example with Wikipedia article names. However, in the experiments domains were selected randomly and not restricted to any particular area.

Both web services require registration as a developer and incur a cost if exceeding

[2] http://code.google.com/p/language-detection
[3] http://incubator.apache.org/opennlp
[4] http://code.google.com/p/ik-analyzer
[5] www.nzdl.org/kea
[6] goo.gl/KXaPw
[7] goo.gl/CpNgp

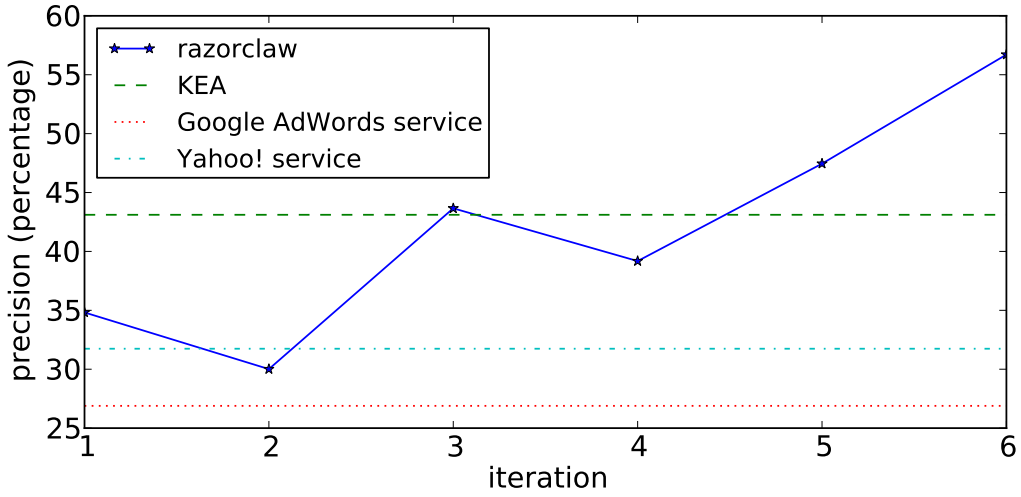


**Figure 4.14:** The performance comparison of competing algorithms and services.

the free quota. According to the specification, text content of webpages (including the title and meta) were sent to the Yahoo! service and URLs to the Google one.

### 4.5.4 Crowdsourcing Based Experiments and Results

To carry out the evaluation in an efficient and cheap fashion, a popular crowdsourcing platform was employed to judge the results of multiple algorithms. A test was conducted on oDesk.com and 23 human assessors who passed the test were randomly selected. Then they were asked to rank the relevance between the domain and keywords on a one (least relevance) to five (most relevance) scale. Very detailed instructions were given to help keep the ranking as consistent as possible among different assessors and in different iteration steps. For broken links or empty keywords (some algorithm failed to give the result) the assessors were asked to give a zero. oDesk.com was chosen for its simplicity, openness, big candidate pool, and various tools to monitor the progress. Traps were designed, screenshots and remote desktop monitoring were used to reduce fraud and malicious behaviours. Among the returns three users were marked careless (by trigging three out of ten traps) and their results were completely removed.

There were six iterations of experiments in total, on 7th June, 17th July, 4th August, 11th August, 15th August, and 20th August 2011. The comparison of competing algorithms is reported in Figure 4.14. As introduced above, the relevance ranking ranges from 0-5 with 0 is specifically for failures (website did not open, encoding or language was not recognised, or other issues resulting in failure of extracting key-

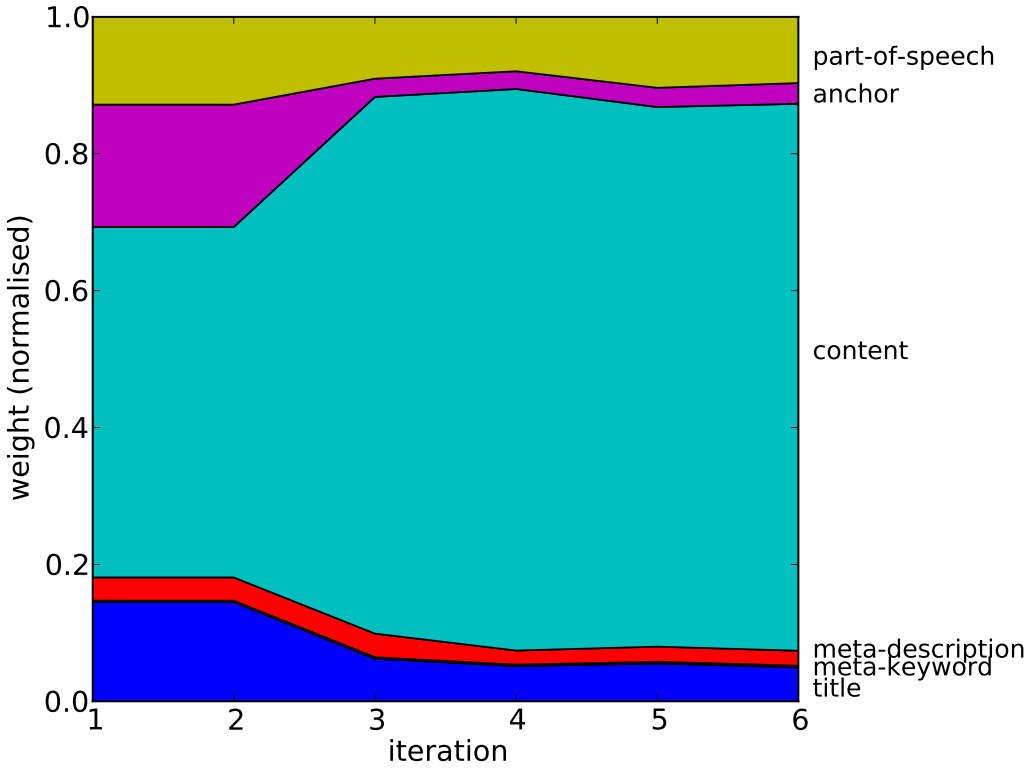


**Figure 4.15:** The evolution of weights of some features of the proposed system. Features with too small weights are not included here. The initial weights were obtained from the work of [150]. The result showed that the *content* of webpage is the most influential factor, followed by *part-of-speech* and *title*. The *keyword* and *description* in the HTML meta tag are in fact not trustworthy.

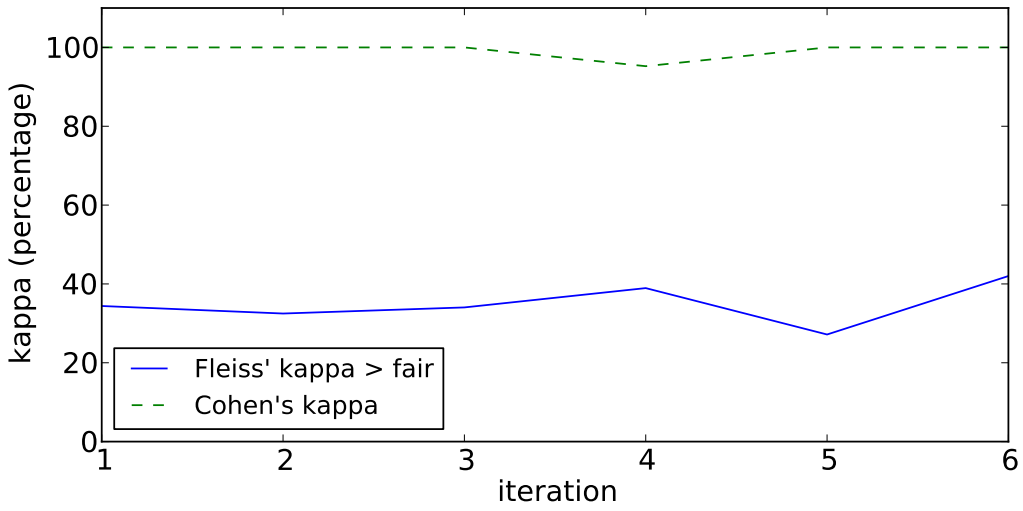


**Figure 4.16:** Measuring the agreement for crowdsourcing ranking results. The *fair* in Fleiss' kappa corresponds to 0.2.

words). In every iteration, each domain was ranked by at least two assessors.

First the overall precision (cumulative relevance ranking against the highest possi-

ble score) is compared. Figure 4.14 shows that RAZORCLAW performed better than the second best (KEA), with 8.29% behind in the first iteration, but 13.6% improvement in the last one.

From the precision plot the improvement of RAZORCLAW is clearly observable. At iteration 2 and four the algorithm performed worse than the previous round, suffering from the exploration. However, the algorithm was able to pick up new sets of weights of features as anticipated. Figure 4.15 shows that the corresponding evolution of the weights. In the figure, top 6 features are listed but ones with too small weights are not included. The *content* was the most influential factor, followed by *part-of-speech* and *title*. The *keyword* from the HTML meta tag is, in fact, not trustworthy, which is consistent with the no-use decision in major search engines like Google [152]. The measurement of agreement is plotted in Figure 4.16. The average of Cohen's kappa of any two assessors is reported for each iteration. Similarly the Fleiss' kappa of all assessors of each iteration is plotted. The agreement of results is generally satisfying.

## 4.6 Conclusions

This chapter presents a model of optimally selecting ads in an online setting. Based on POMDPs, the belief updates are formulated by taking the correlation of ads into account. It is mathematically show that the belief update across ads is similar to the *word of month* principle employed in collaborative filtering. To make use of the correlated belief update two approximate methods are proposed: one is derived from the Value Iteration and Sampling approach, whereas the other is based on the Upper Confidence Bound solution. The empirical experiments compare the proposed algorithms with various baselines using a real-world dataset and show that the Bayesian inference with correlations makes the exploration more efficient and significantly uplifts the payoff.

This chapter also proposes and evaluates an adaptive keyword extraction system, as a complement for pulling ads from various sources, since there are still many ad sources that allow manipulating the output by providing keywords. The system is designed based on BM25F and multi-armed bandits models and has been evaluated by crowdsourcing techniques. The iterated experiments showed good performance of proposed algorithm and exposed important most features (content and part-of-speech).

# Chapter 5

# Reserve Price Optimisation

Revenue from ad exchanges [153] which use Real-Time Bidding (RTB) is getting more and more important to online publishers. In RTB, a publisher generates a bid request for each impression in real-time and then sends a request to advertisers via ad exchanges. Advertisers then calculate and submit bids for this impression using bidding algorithms.

In this process, the reserve price is one of the most powerful tools for the supply side to manipulate auctions [31]. A reserve price defines the minimum that a publisher would accept from bidders. It reflects the publisher's private valuation of the inventory; bids will never be accepted if they are below the reserve price. In the second price auction, which is commonly used in RTB, the reserve price could potentially uplift the revenue. Figure 5.1 illustrates how the final price is calculated in an auction with a reserve price. Let $b_1, \ldots, b_K$ denote the descending bids and $\alpha$ the reserve price. Then, the desirable case is $b_1 \geq \alpha > b_2$ where the publisher gains additional revenue of $\alpha - b_2$; the neutral case is $b_1 > b_2 \geq \alpha$ where the publisher has no extra gain; and the undesirable case is $\alpha > b_1$ where the publisher suffers from a loss of $b_2$. These cases directly motivate the work of this chapter.

Note that in practice, publishers can choose to disclose or hide reserve prices in bid requests. When a reserve price is present, most advertisers would still return bids even when the bids are lower than the disclosed reserve. This helps to avoid the *time-out penalty* (a penalty for bidders not responding in time, used to protect bidders from being flooded) which would reduce the bid request volume over time.

This optimisation problem has been studied in the context of sponsored search [154, 155, 37, 38] by using the optimal auction theory or machine learning techniques. However, the problem in the RTB context is different and unique. First, the

optimal auction theory requires knowing the distribution of the advertisers' private yet true assessments of the impression before calculating the optimal reserve price [154]. In RTB, it becomes a lot harder to learn the distribution. An advertiser is required to submit a bid for each impression using proprietary algorithms, which are never disclosed to publishers and could rely heavily on privately-owned audience profiles (a.k.a., users' interest segments). Besides, various practical constraints such as the budget, campaigns' lifetime, irrationality, divert advertisers from bidding at private values. This difference makes the private value based algorithm inefficient in practice. Thus, it is of great interest to empirically study the subject and examine several commonly adopted algorithms in the real-world as reported in this chapter.

Second, unlike sponsored search, an advertiser does not have the keyword constraint and faces almost unlimited supply of impressions in RTB. Setting up an aggressive reserve price could easily drive the advertisers away from those placements and force them to look for something similar but cheaper. This possible consequence is analysed in the online experiments and the attrition hypothesis is rejected in today's RTB marketplace, which could be due to the complexity of the experiment scheduling or less sensitivity of common bidding algorithms. The little attrition implies good chance of implementing the optimisation in the current ecosystem.

This chapter presents the first field study of RTB auctions and discuss the reserve price optimisation in this context. The empirical study is based on the analysis of real-world data from a production platform. Thorough discussions about the commonly adopted algorithms for reserve price optimisation are included, including both the private-value-free and the private-value-based ones [36, 154, 37], and their variations. At last, large-scale online experiments were conducted to test these algorithms. The results suggest that the proposed game theory based ONESHOT algorithm performed the best, and the superiority (12.3% on average) is significant in most cases.

## 5.1 Related Works

Up to now the reserve price problem has mainly been studied in the sponsored search (SS) field. For instance, in [37], the authors have studied the reserve price problem in a real-world online advertising system (Yahoo! sponsored search). The authors test optimal auction theory on 450k keywords and the results diverge by number of

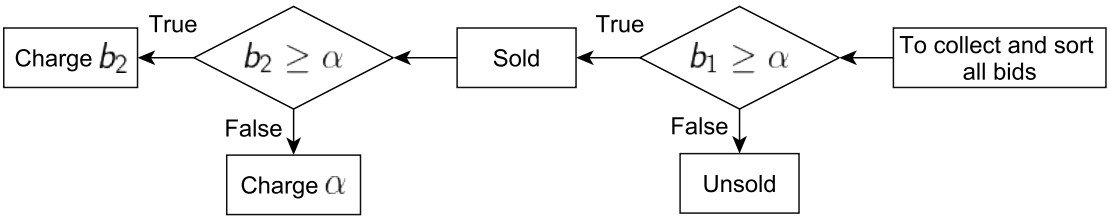


**Figure 5.1:** The decision process of second price auctions on the publisher side. The desirable case is $b_1 \geq \alpha > b_2$ where the publisher gains extra payoff of $\alpha - b_2$. The *soft floor prices* is ignored to avoid making the process a lot more complicated.

impressions: for keywords with high search volume the uplift on revenue is good and statistically significant. However, for other keywords the total revenue declines by 2.2% on average. This result is consistent with the experiment results in this chapter, that is, the optimisation may not be helpful when there are very few bidders. Additionally, the bidders' attrition analysis was conducted which could be more important in RTB.

In [154] the optimal auction problem is characterised for sponsored search. The authors demonstrate the calculation of optimal reserve price for multi-item auctions and conclude that the optimum is independent of the number of bidders. However, in [38] the authors suggest that both the number of bidders and ad links affect the optimal reserve price. Some assumptions are made in the first paper [154]: advertisers know their value per click; they have the same Click-Through Rate (CTR) at a given position; they share the common knowledge of position CTRs; they all maximise the expected profit. These assumptions might be realistic when search engines bids for advertisers, but however are far from achievable in RTB. Also in the SS context, in [155] the advertiser-specific reserve price is discussed in both the generalised second-price auction (GSP) and the Vickrey-Clarke-Groves (VCG) auction [8] settings. As proved in the paper the advertiser-specific reserve prices could result in losing truthfulness in the VCG pricing; but could lead to envy-free equilibrium in the GSP pricing.

The reserve price study is planned in the RTB environment. In [153] the abstract model of an ad exchange is proposed and several problems are discussed, including auction mechanism, call out optimisation [156], publisher revenue optimisation [157], arbitrage bidding and risk analysis [158]. As suggested by the authors, the reserve price problem takes an essential role in yield optimisation yet remains unsolved. In this thesis, the problem is studied at placement-level (a group of impressions); the impression-level dynamic reserve prices (e.g., user-specific) is left to the future works.

There is also rich literature from bidders' perspective on how to bid when a randomised reserve price in ad network is present [58, 159]. The optimal auction design is studied in [159] with two layers: central seller to intermediaries and intermediaries to bidders. The authors conclude that revenue-maximizing intermediaries will use an auction with a randomized reserve price chosen from an interval in equilibrium, and the optimal reserve price decreases with the number of buyers. In this chapter, although no intermediary is employed, the relationship of the optimal reserve price and number of bidders is observed and discussed.

Due to the similarity in mechanisms, RTB also can borrow a lot from established research on auction theories [160, 161, 162], esp. electronic commerce [163, 164]. In [165] the detection and reaction towards reserve prices cheating are first discussed for online auctions. The authors of [162] discuss the situation that an item could be resold if previous auctions fail due to high reserve price. An equilibrium reserve price was proven to approach the one in an optimal static auction. Meanwhile, there are papers on scoring inventories [67] considering unique characteristics of RTB auctions (e.g., number of bidders, bid time, bidders' identity). However, few of them take the reserve price into account.

## 5.2 An Empirical Study of RTB Auctions

An empirical study was conducted to understand the unique features of RTB auctions. The study looks at the bidders' lifetime, bidding pattern across a day, the measurement of advertising performance, etc. The knowledge gained from the study is especially helpful to understand the drawbacks of optimal auction theory in RTB practice. It also inspires the design of the one-shot game based reserve price optimisation model in the following section.

This section first reports the statistics of dataset used in the empirical study and some general analysis on bidders and bids. Then it discusses three aspects (bidding behaviour, bids' distribution, frequency and recency) and associated problems (reserve price detection, daily pacing, and selective bidding).

### 5.2.1 Datasets

The study was carried out in a production ad exchange based in the UK. Datasets were created both for demand and supply side. However, these two datasets are independent thus not linked together in the empirical study.

For advertisers, impression, click, and conversion logs from Feb to May 2013 were sampled. In total there are 52,850,635 impressions, 72,958 clicks, and 37,978 conversions. Note that only the conversions were sampled then back traced to find the associated clicks and impressions. Since these datasets were sampled, various reports (e.g., analytical performance, site domain, attributed conversions, frequency/recency distribution, and so on) were also used to measure the performance of advertisers.

Auction logs were recorded, too, for placements from registered publishers of the same ad exchange. In total, there are sampled 12,965,119 auctions from 50 placements, ranging from Dec 2012 to May 2013. These placements are from 16 websites of different categories, including *news*, *finance*, *pc & console games*, *gadgets*, *sports*, *entertainment*, and so on. Auction logs were sampled to suit the limited computational and storage capability. Therefore, various reports were used to measure the overall performance of publishers.

#### 5.2.1.1 Periodic Patterns

Due to the typical human daily routine there are always periodic patterns of activity. Figure 5.2 and Figure 5.3 show clear daily patterns of both impressions and clicks from a single website. For the number of impressions, there was also a weak weekly pattern that during weekends fewer visitors came to the website. There was also a daily pattern for conversions: during daytime the numbers of conversions were larger and the CVRs were higher compared to sleeping hours, c.f. Figure 5.5 and Figure 5.4.

Note that there are two types of conversions: *post-click* that the user saw an impression, clicked, and finally converted (while remaining on the advertiser's website and without any interruption); *post-view* that the user saw an impression, but did not click, but still converts (by visiting the website later on directly). These two types were plotted separately.

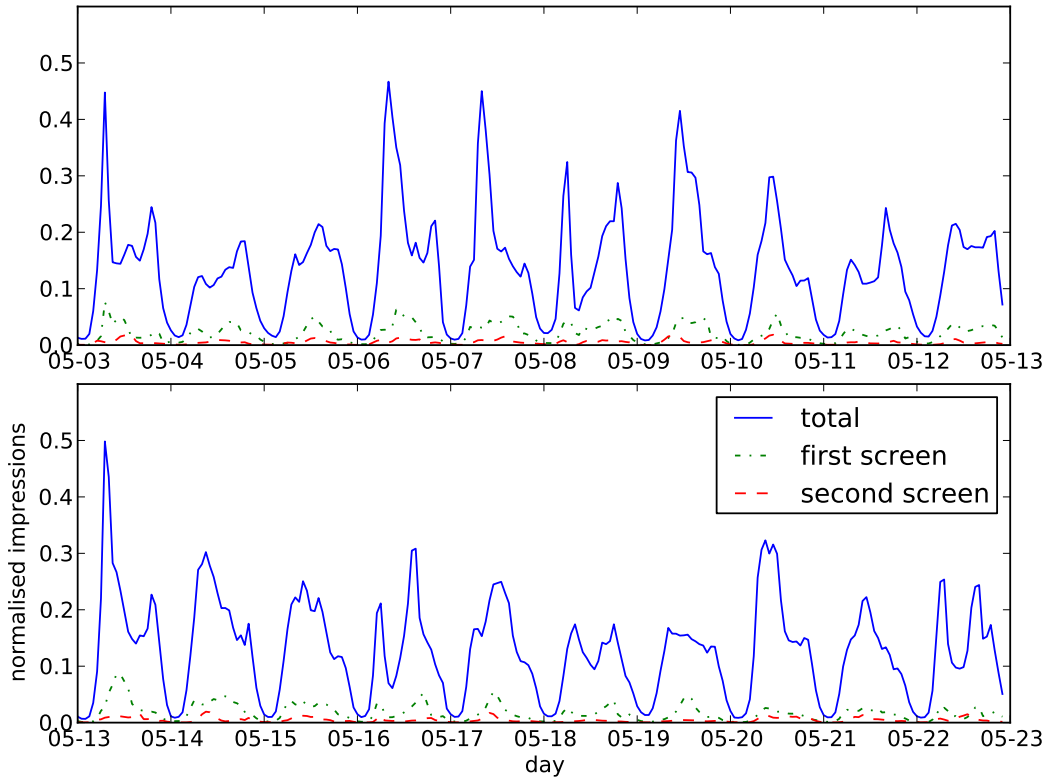


**Figure 5.2:** The normalised impression time series snippet by hour. A definite daily pattern and less clear weekly pattern, that fewer visitors come to the website, can be observed. The data is from a single website, but the pattern is very similar for all websites which were evaluated. The green dotted line denotes the impressions from the front page; the red segmented line denotes the impressions from the second-level pages (i.e., those are linked directly from the front page).

#### 5.2.1.2 The Soft Floor Price

In modern ad exchanges, publishers are usually given the option to set a soft floor price $p_s$ along with a hard floor price (the traditional reserve price) $p_h$. The auction process is illustrated in Figure 5.6. However, note that the business requires that the soft floor to be either higher than the hard floor or zero. By setting a high soft floor price (e.g., $1000 CPM), the publisher can change the mechanism from the second price to the first price auction, which always charges the winner the amount that he bids. Unlike the hard floor, advertisers usually are not aware of the existence of soft floor prices, especially in RTB environment where the available inventory is unknown before bidding.

The ratio of the *effective* first price auction and the second price auction in the dataset was checked, using impression logs that include bid price and price paid. There was no explicit indicator of the auction mechanism being employed. Thus, a simple heuristic was used: when `bid_price` = `price_paid`, it was considered first price

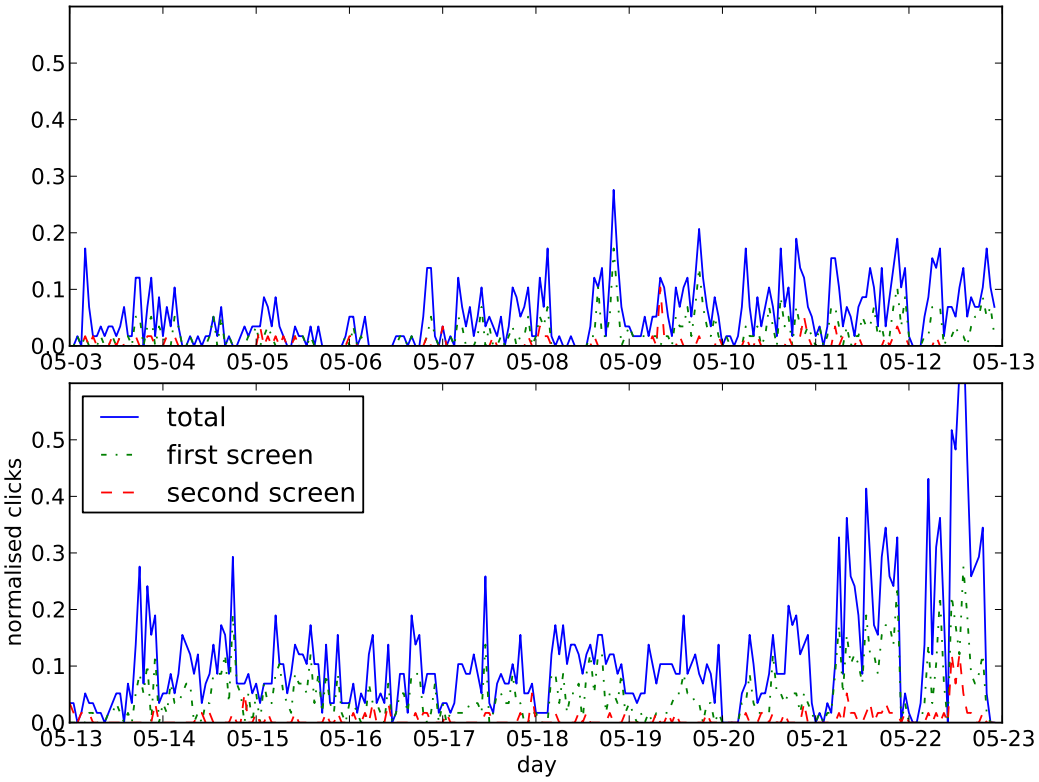


**Figure 5.3:** The normalised click time series snippet by hour from the same website as of Figure 5.2. A daily pattern with lots of noises exists, too. Other websites had the very similar pattern.

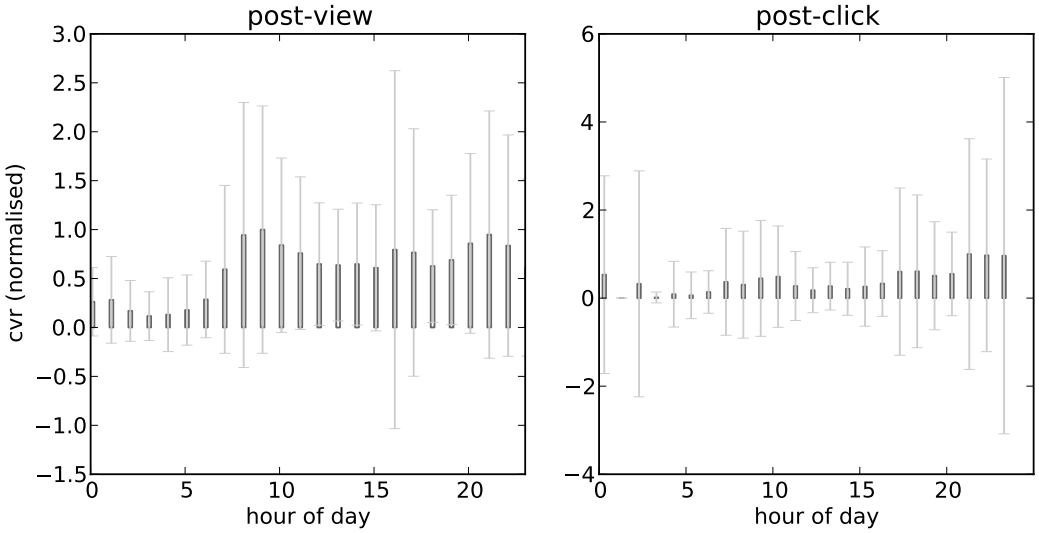


**Figure 5.4:** The distribution of normalised post-view and post-click Conversion Rate (CVR) against time. The plot was created using all conversions in the demand-side dataset. The error bar shows the standard deviation. The plot shows that the CVRs were higher during wake hours.

auction. In total, there were about 40% impressions bought in first price auctions. However, these impressions accounted for 55.4% of the total advertisers' budget.

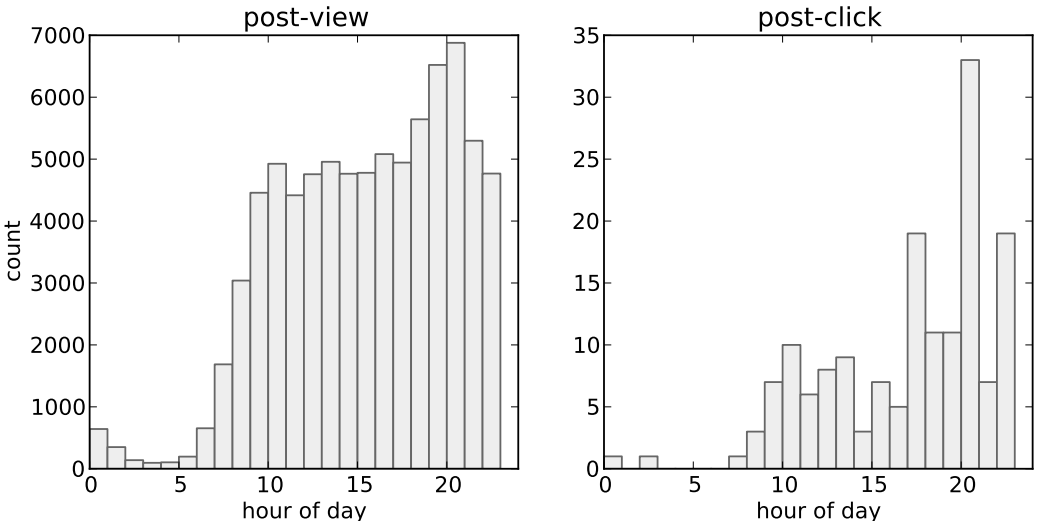


**Figure 5.5:** The distribution of number of post-view and post-click conversions against time. The plot was created using all conversions in the demand-side dataset.

To my best knowledge this mixture of auction mechanisms is mostly unaware of. Needless to mention the counter measure from advertisers. In different types of auctions advertisers have different optimal bidding strategies [166]. Specifically, in a second price auction an advertiser should bid his private value; while in a first price auction he should bid lower than the private value. Thus, the mixture, as illustrated in Figure 5.6, puts advertisers in a not favourable position (i.e., bidding sub-optimally by assuming it is always the second price auction). It hurts the campaign performance and could damage the advertising eco-system in long term. From the advertisers' perspective, it is worth exploring the floor prices setting of different publishers and act accordingly. For example, the winner may choose to lower his bid (while still winning) to reduce the cost in a high soft floor scenario (the first price auction). Meanwhile, the advertiser who could react to its competitors' moves fastest has a substantial advantage [8]. However, if it's the second price auction, the exploration activities only incur unnecessary cost.

### 5.2.2 Bidding Behaviours

This section studies the advertisers' bidding behaviour to provide a better understanding of the RTB ecosystem. Particularly, it analyses the bids' distributions and the pacing settings of a daily budget.

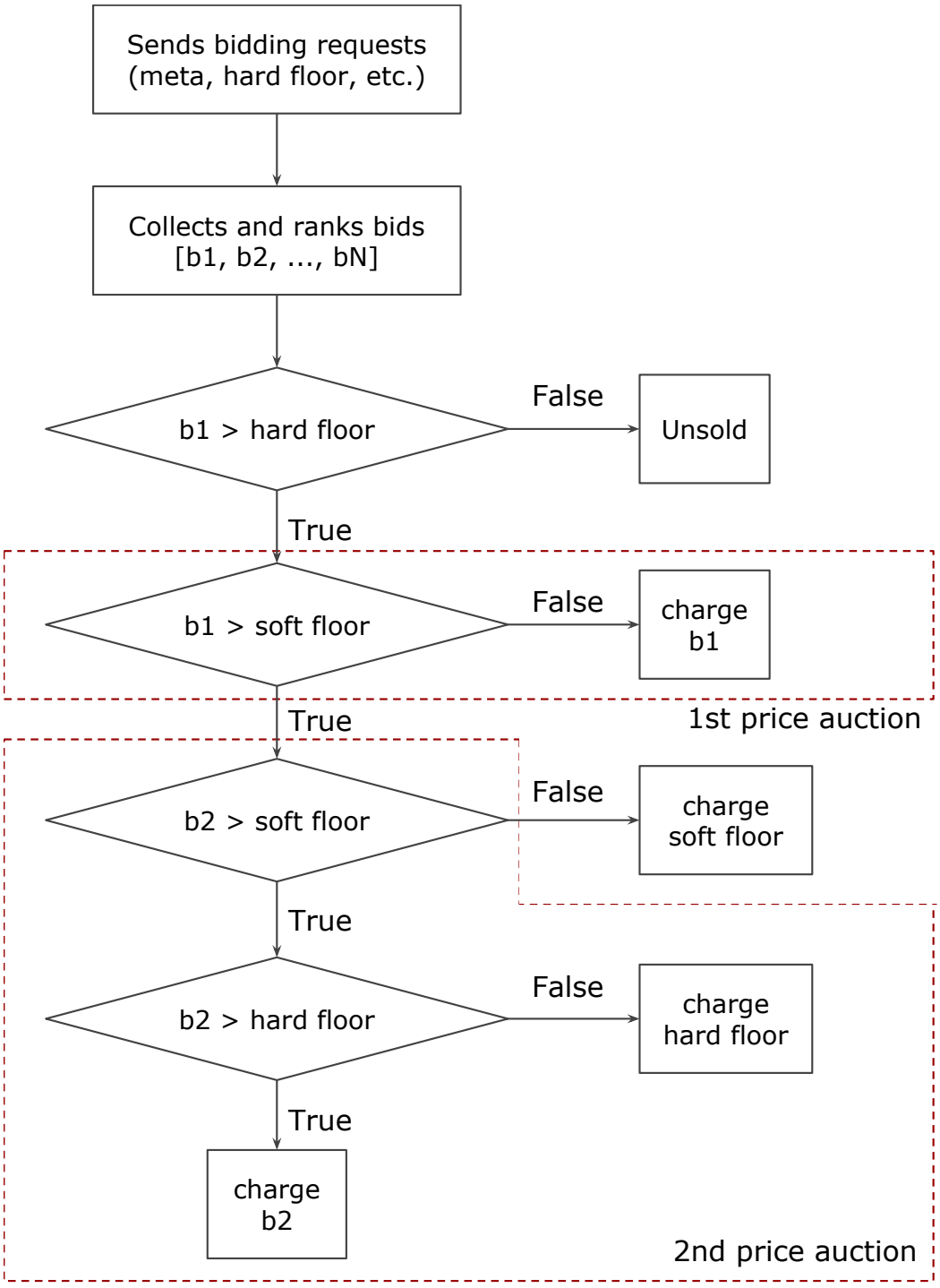


**Figure 5.6:** An illustration of the auction process in modern ad exchanges with both hard and soft floors. The business requires that the soft floor to be either higher than the hard floor or zero. The auction mechanisms are mixed and extended by introducing the soft floor price. The impact of such mixture is mostly unaware of. This complicated setting puts advertisers in an unfavourable position and could damage the ecosystem.

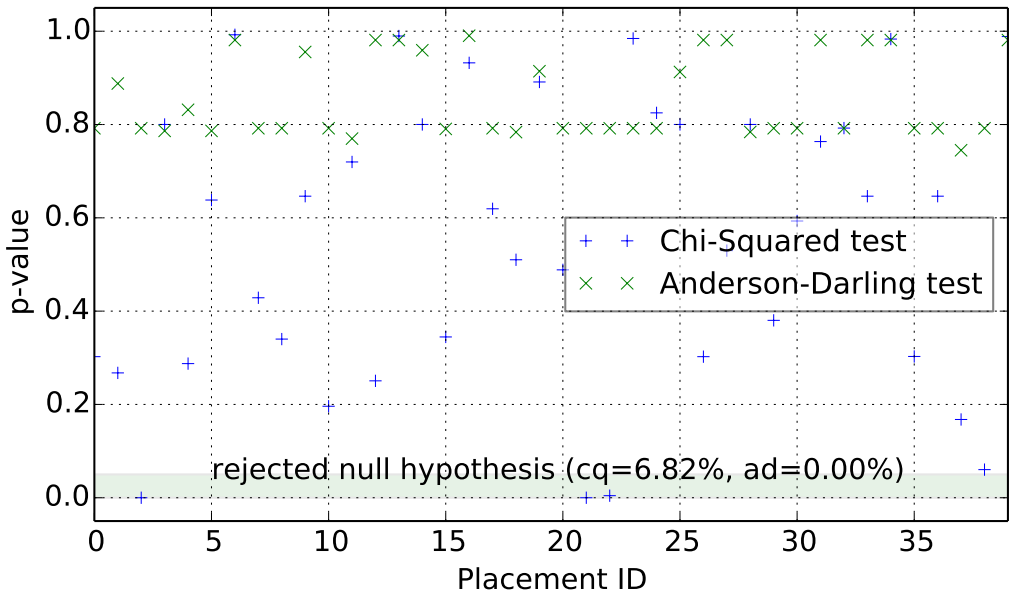


**Figure 5.7:** The bids' distribution test at placement level. Only bids from 3 out of 44 placements (6.82%) accept the Uniform distribution hypothesis. The Uniform distribution is tested by Chi-Squared test and the Log-normal distribution is tested by Anderson-Darling test.

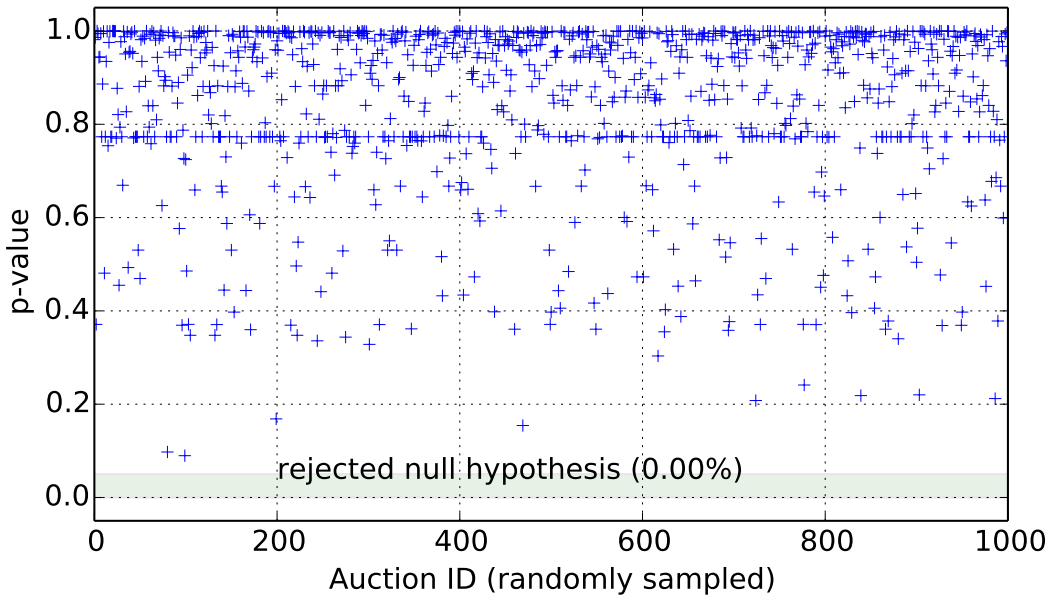


**Figure 5.8:** The bids' distribution test at the auction level. Only bids from less than 0.1% of all auctions accept the Log-normal distribution hypothesis. The plot shows a randomly sampled of 1000 auctions. Only the Log-normal distribution is tested by Anderson-Darling test.

#### 5.2.2.1 Bids' Distribution Test

A key assumption made by the optimal auction theory is that bidders draw their private values from a monotonically increasing distribution. In second price auctions, they bid at their private values; thus, the distribution of bids reflects their individual valuations

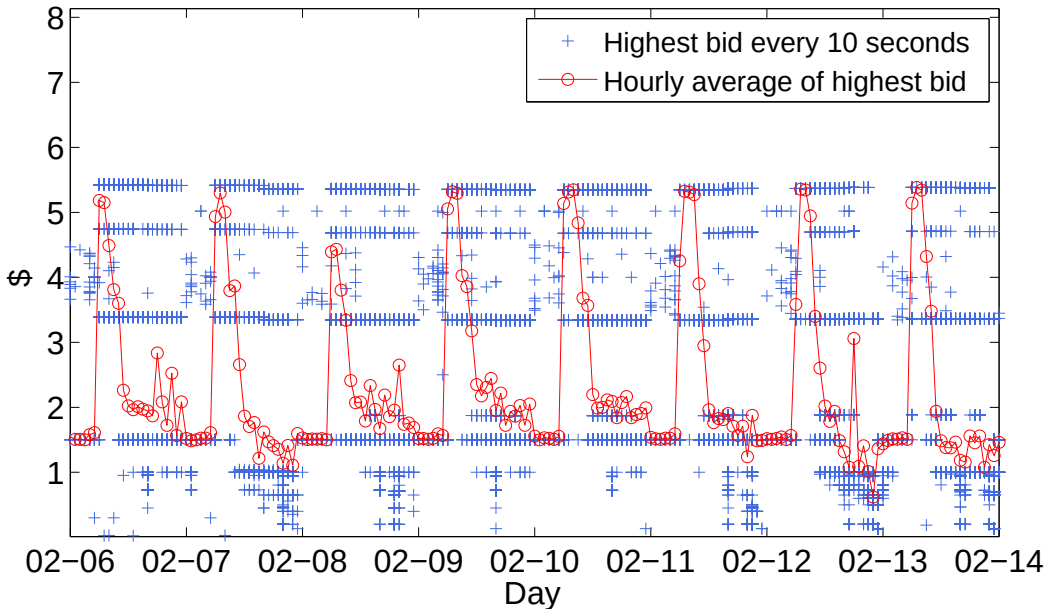


**Figure 5.9:** Hourly average winning bids and bursts on a placement of high level of competition. Since the placement is static, the change of bids is mostly likely due to the usage of audience data.

of an impression. As introduced in Chapter 2 researchers often assume bidders are symmetric, and bids follow Log-normal or Uniform distribution. This assumption then becomes the core of optimal auction theory based algorithms of computing reserve prices [36, 37, 38].

In this empirical study, bids were fitted to these two distributions. For the Log-normal distribution, bids' normality was checked using Anderson-Darling test [167] after taking the logarithm. This test was performed at both the auction level and placement level. For the Uniform distribution, the Chi-squared test [168] was used only at the placement level. P-value$< 0.05$ is used to reject the null hypothesis.

The results are reported in Figure 5.8 and 5.7. Clearly only the acceptance of Uniform distribution assumption at placement level is noticeable yet still at a low ratio of 6.82%. The poor results are probably due to the bursts and randomness of bids as illustrated in Figure 5.9. However, to be consistent with the research literature and due to the limited time, the Log-normal assumption was adopted when implementing the optimal auction theory based algorithm in the experiments.

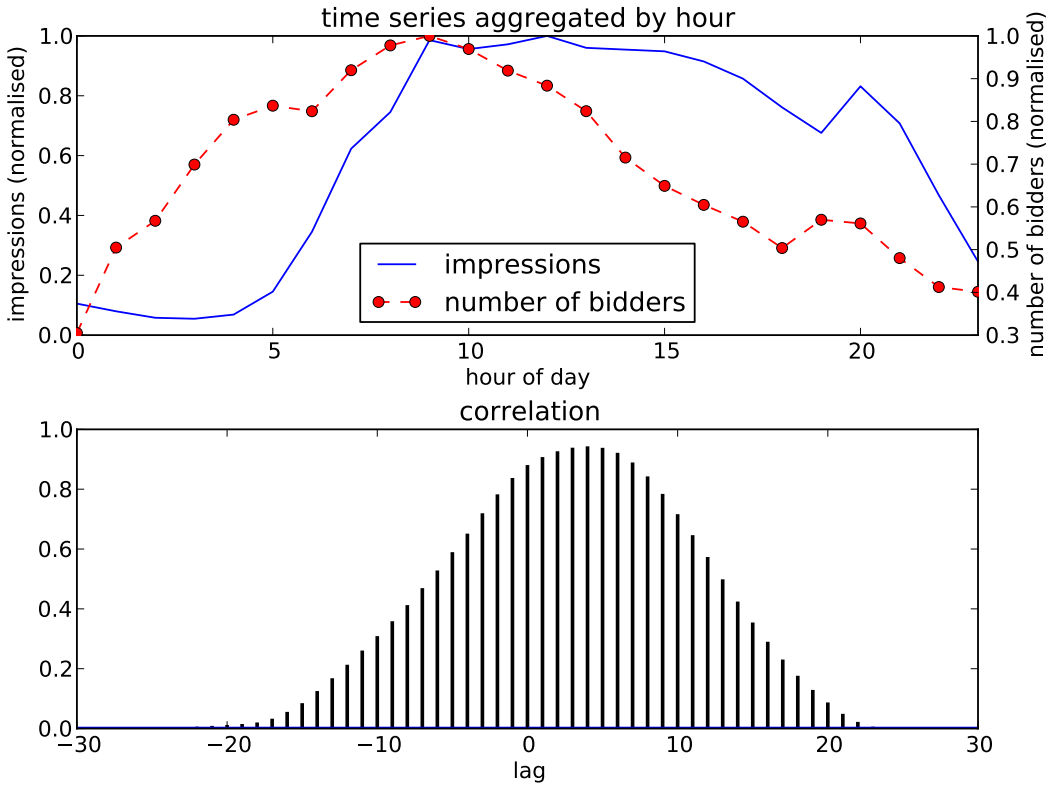


**Figure 5.10:** The distribution of the number of bidders and impressions against hour-of-day. Their correlation plot shows a clear lag when they reach the maximum in a day. This lag indicates the unbalance of supply and demand of the market in some hours. Besides, the fact that there are more bidders in the morning may be due to a mixture of hour-of-day targeting and no daily pacing setup. The plots used three months worth of data sampled from a single placement. Note that for some placements the lag was not very clear.

#### 5.2.2.2 The Daily Pacing

The daily pacing refers to the way that advertisers spend their budget in a single day. Usually an advertiser submits a daily budget for his campaign, and chooses from spending it evenly throughout a day (uniform pacing) and as fast as possible (no pacing). The *no pacing* option may lead to a premature stop easily, which means the budget depletes too quickly so advertisers cannot capture traffic later in the day, that may have high-quality impressions. An instance of premature stop is illustrated in Figure 5.11 (day 1). The *uniform pacing* also suffers from the traffic problem: if high quality impressions appear in early part of the day, the pacing setup would not be able to capture all of them; if there is not enough traffic in the late part of the day, the pacing setup would not be able to spend all the budget (usually called under-delivery). In sum, although being widely used in DSPs they are not good daily pacing strategies.

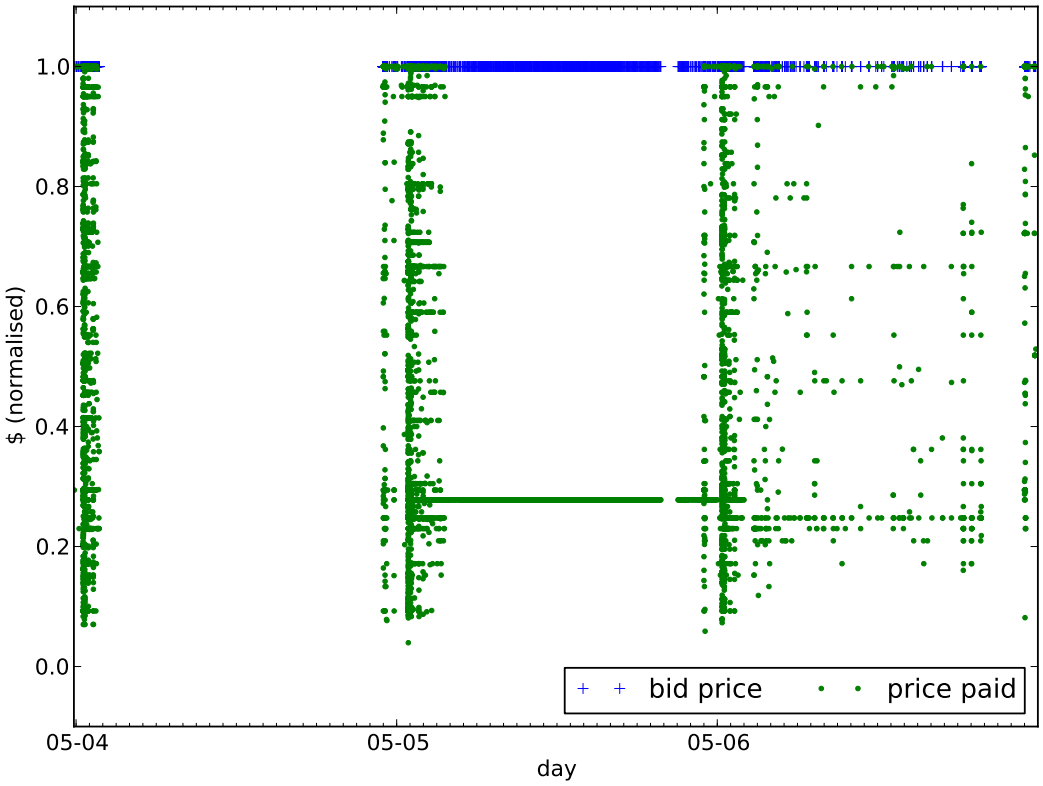


**Figure 5.11:** An interesting instance: an advertiser switched from no pacing to even daily pacing. He was bidding at a flat CPM. With ad exchanges, the large amount of available inventories could deplete the budget quickly. However, even daily pacing is far from optimal because it does not consider the performance in different time slots. Note that in practice the pacing engine would spend slightly more at the beginning of the day to learn the available impressions and to calculate the spending speed, especially for campaigns with small budgets.

Figure 5.10 shows the hourly mean of the number of bidders and impressions, which are normalised respectively. There is an obvious lag when these two series reach their maximum in a day. Generally speaking, the number of bidders peaks in the morning but the number of impressions peaks afterwards. This lag indicates the unbalance of supply and demand of the market in particular hours. For example, there are more bidders competing over limited impressions in the morning, resulting in high winning bids as shown in Figure 5.10, which in turn costs more of advertisers. However, this higher cost does not necessarily lead to higher performance. Figure 5.4 shows that both post-view and post-clicks CVR peak in the evening, which argues that intense bidding activities in the early hours are not reasonable.

This distribution of bidders throughout a day may be due to a mixture of hour-of-day targeting and no daily pacing setup. Most of the advertisers wish to skip the last

night hours because of the low CVR. Some of them may use the no pacing option to avoid the risk of under-delivery as discussed before. When these bidders start to bid in the morning, they win lots of impressions with high bids, and they quit as their budgets depletes quickly. The validation of this hypothesis is left to the future work.

A reasonable alternative is the dynamic pacing against the performance. It is intuitively correct to spend more budget in hours that generate more clicks or conversions, and less in low performing hours.

Note that this problem is not the same as a typical budgeted multi-arm bandit problem that has been discussed extensively in the literature [169, 170, 171]:

1. In this allocation problem the advertiser can only explore the current time step;
2. There is potential over or under-delivery in a time step due to the latency in the practical implementation.

Therefore, re-distributing the remaining budget is required after every time step.

### 5.2.3 Conversion Rates and Performance Measurement

Considering the maintenance cost (e.g., servers and bandwidth) it is not wise for an advertiser to submit a bid every time he receives a bidding request. Among various factors that he uses to decide to bid or not, the frequency and recency factors are common and important ones.

#### 5.2.3.1 The Frequency Factor

The frequency factor (or frequency cap, FC) defines how many times ads (a.k.a., creatives) would be displayed to a single user. It can be applied to campaign groups, campaigns, or creatives separately. For example, given a campaign with 3 ads, an advertiser could set FC(campaign) = 6, FC($ad_1$) = 2, FC($ad_2$) = 3, and FC($ad_3$) = 4 and these settings would work together. The same user would at most see $ad_1$ twice, $ad_2$ three times, $ad_3$ four times, and all these ads displayed to him together would not exceed six times.

The frequency factor is normally set based on historical data and needs to be constantly adjusted during the flight time of the campaign, because different campaigns ask for very different FCs as illustrated in Figure 5.12. The campaign 1 (left) received the highest CVR with 6-10 impressions, which is also true for the cumulative CVR metrics. However, the campaign 2 (right) received the highest CVR with 2-5 impressions. If the

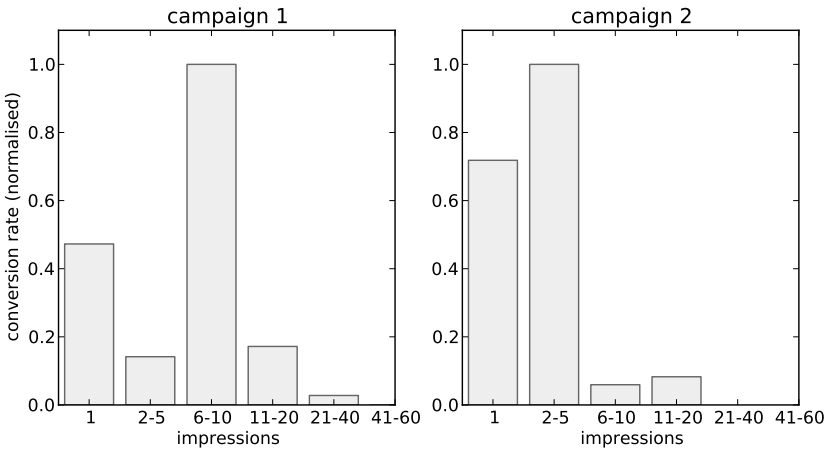


**Figure 5.12:** The frequency against CVR from two different campaigns. Campaign 1 (left) received the highest CVR with 6-10 impressions, which is also true for the cumulative CVR metrics. However, campaign 2 (right) received the highest CVR with 2-5 impressions. If the campaign 1 sets a frequency cap of 2-5 impressions, most of conversions would not be achieved. If the campaign 2 sets a frequency cap of 6-10, nearly half of impressions could be wasted.

campaign 1 sets a frequency cap of 2-5 impressions, most of the conversions will not be achieved. If the campaign 2 sets a frequency cap of 6-10, nearly half of impressions could be wasted. Therefore, a good FC setting is crucial to the efficiency of advertising.

Setting an optimal FC requires finding the right metrics to measure the efficiency of different FCs. See the example in Table 5.1 where popular metrics are compared. Assume there are 100 users; the CPM is fixed at $10; the advertiser's conversion goal is worth $500. The table shows that using different metrics could lead to very different decision. If the advertiser uses CVR as most of advertisers do, he would go for FC=3; if the advertiser cares more about the total number of conversions, he would use FC=5 (seven conversions); if the advertiser cares more about CPA he would use FC=3, too; if the advertiser measures the performance by ROI, he would go for FC=2.

Using FC = 2 seems the most profitable. However, choosing FC = 3 is reasonable, as well, especially when considering the long-term impact: FC = 3 gives more conversions at low cost, and once these users are attracted they have a higher chance of converting again in the future.

#### 5.2.3.2 The Recency Factor

The recency factor (or recency cap, RC) helps to decide to bid or not, based on how recently the ad was displayed to the same user. It also works at different levels including campaign groups, campaigns, and creatives. For example, an advertiser can set

**Table 5.1:** The comparison of different metrics against frequency caps (FC). Note *cvr* and *convs* are *fc* specific, i.e. the extra value could be gained by increasing the frequency cap from the previous level to the current one. The *cumulative cvr* is the CVR advertisers use: total conversions divided by total impressions. The *cpa* gives the cost-per-acquisition. The *roi* gives the return-on-investment based on the advertiser's conversion valuation. These values are calculated by assuming there are 100 users; the CPM is fixed at $10; the advertiser's conversion goal is worth $500.

| FC | CVR | (Cumulative) CVR | Convs | CPA | ROI |
|---|---|---|---|---|---|
| 1 | 0.0000 | 0.0000 | 0 | - | 0.00 |
| 2 | 0.0150 | 0.0150 | 3 | 667 | 1.50 |
| 3 | 0.0067 | 0.0167 | 2 | 600 | 1.25 |
| 4 | 0.0025 | 0.0150 | 1 | 667 | 1.00 |
| 5 | 0.0020 | 0.0140 | 1 | 714 | 0.88 |
| 6 | 0.0000 | 0.0117 | 0 | 857 | 0.70 |
| 7 | 0.0000 | 0.0100 | 0 | 1000 | 0.58 |

RC(campaign) = (1 hour) so that all ads from this campaign would be displayed to the same user only once in every hour. Similar to the frequency factor, RCs are useful to achieve the best advertising efficiency. For example, displaying ads intensively incurs a high cost but little effect (or even getting people annoyed). Users need to think, compare, and then make decisions. A better strategy is to display the same ad right after the *thinking time* to remind users (e.g., financial services). However, for some campaigns (e.g., booking flight tickets) users would convert very quickly or not convert at all, which requires relatively intense advertising.

Figure 5.13 plots the RC against CVR of two different campaigns in the dataset. Both campaigns show the highest CVR at the 1-5 minutes level. However, for campaign 2 (below) the CVR is still not negligible after a long time (14-30 days). If the advertiser uses a more strict RC setting (e.g., do not display ads to users who were first exposed 14 days ago) he could lose potential conversions. On the other hand, using a loose RC setting for campaign 1 (above) would only waste the advertising budget since the CVR is very low after 14 days.

Another interesting observation is given in Figure 5.14. Similarly to the frequency factor, the analysis of efficiency of RCs requires metrics based on the understanding of advertising goal, which is not repeated here.

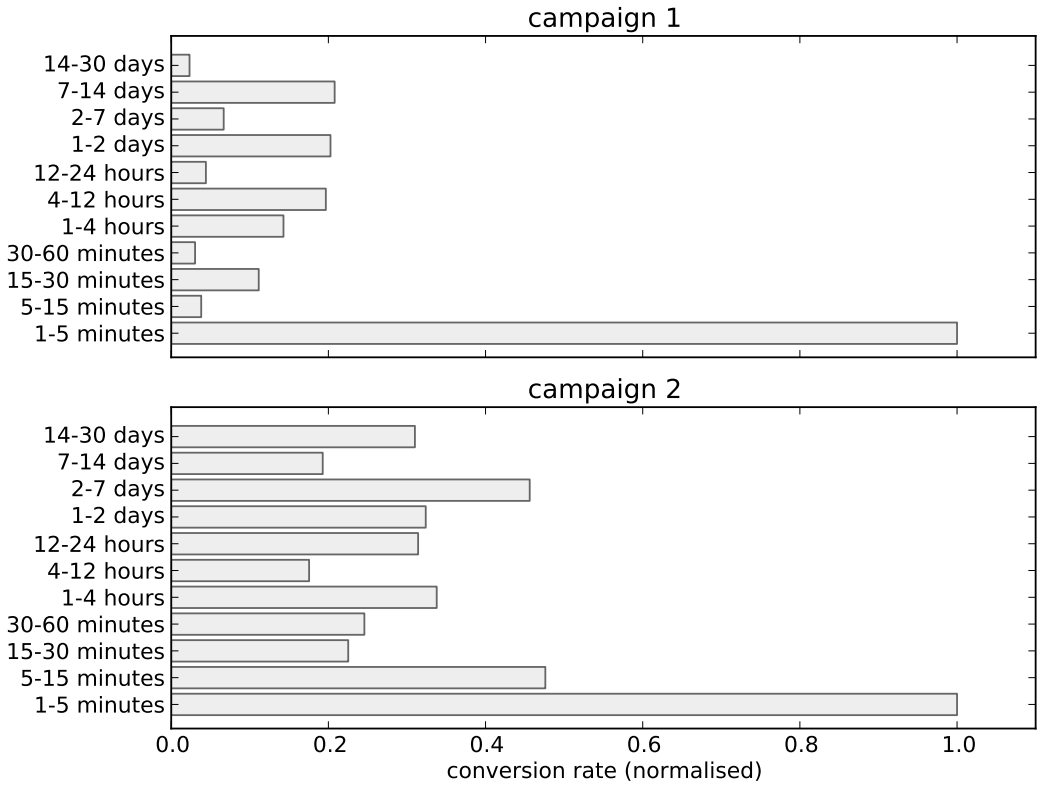


**Figure 5.13:** The recency factor against CVR from two different campaigns. Both campaigns show the highest CVR at the 1-5 minutes level. However, for campaign 2 (below), the CVR is still not negligible after a long time (14-30 days). If the advertiser uses a more strict RC setting (e.g., do not display ads to users who were first exposed 14 days ago) he could lose potential conversions. On the other hand, using a loose RC setting for campaign 1 (above) will only waste budget since the CVR is very low after 14 days.

The above analysis shows the importance of setting up proper FCs and RCs. At present, the finest granularity of these settings is the creative level. With RTB, advertisers can push them to a finer granularity: setting FCs and RCs for individual users.

## 5.3 The Reserve Price Problem and Solutions

This section discusses the reserve price optimisation problem, proposes a solution, and compare its performance with other baselines in real-world online experiments.

Suppose a publisher tries to maximise the ad revenue for a single placement (i.e., single-item). Impressions from this placement are sold using the second price auction in RTB. The publisher could set a reserve price $\alpha$ *before* an auction is conducted.

To simplify the discussion, let us assume there is only one impression at each step $t$ and the publisher is optimising the revenue over horizon $T$. Let us also assume that at each step $t$ there are $K \geq 2$ bidders participating in the auction. If there is only

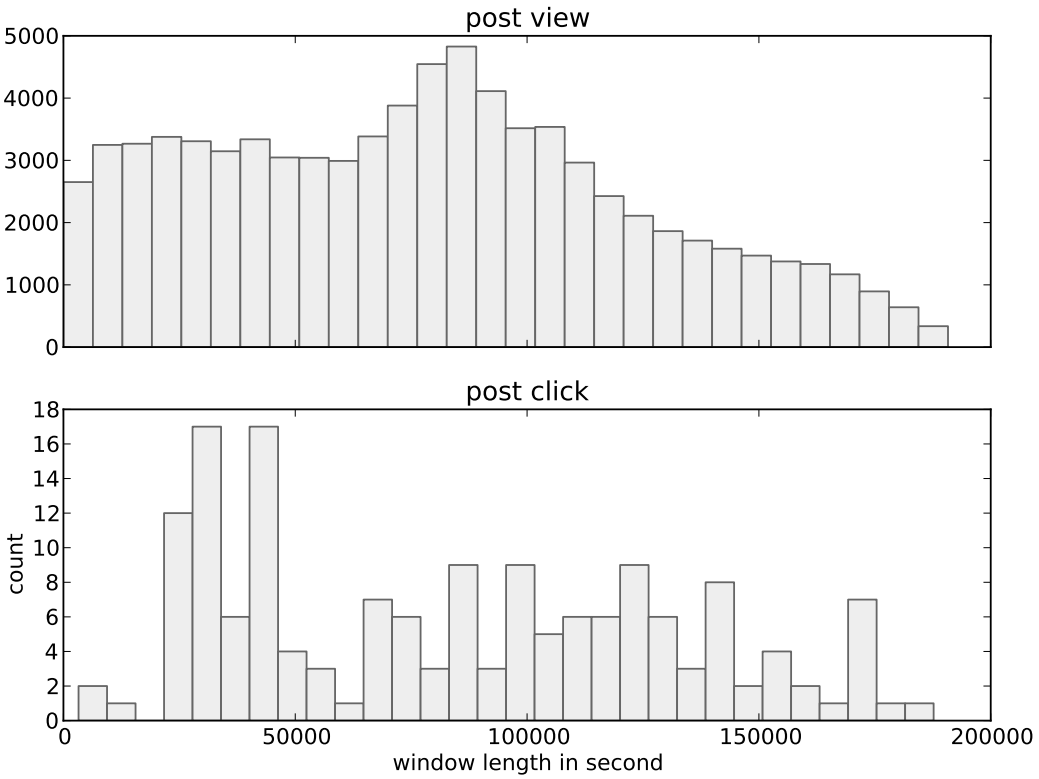


**Figure 5.14:** The histogram of conversion window lengths since the first exposure. Post-view and post-click conversions are drawn separately but are from the same campaign. The plot for the post-view conversions roughly distinguishes people into two groups: *impulse purchaser* who converted quickly and *rational purchaser* who took some time to think after seeing the first ad. In order to maximise conversions, the advertiser would consider setting up RCs to target these two types of users. Interestingly the plot of post-click conversions suggests that the time needed to fill the form and complete the purchase varied a lot for different users (or they could leave the page open for a while).

one bidder it is equivalent to having the second highest bid equals to the minimal bid (usually \$0.01). Note bidders could be different each time and could change quite a lot—this introduces great noise into the system and challenges the estimation of the distribution of the private evaluation values from the advertisers, where most auction theories assume there is a fix distribution of bids.

For each auction, let us denote the final bids of the placement as $b_1(t), b_2(t), \cdots, b_K(t)$. Without loss of generality let us assume $b_1(t) \geq b_2(t) \geq \cdots \geq b_K(t)$. Therefore, without a reserve price ($\alpha = 0$) the payoff could be denoted as $r(t) = b_2(t)$. Now suppose the publisher sets a non-zero reserve price at each step, denoted by $\alpha(t)$. The payoff

function becomes:

$$r'(t) = \begin{cases} \alpha(t), & b_1(t) \geq \alpha(t) > b_2(t) \\ b_2(t), & b_2(t) \geq \alpha(t) \\ 0, & \alpha(t) > b_1(t) \end{cases} \tag{5.1}$$

The overall income is $R(T) = \sum_t^T r'(t)$. It is assumed zero payoff when the reserve price is too high. In practice, publishers usually redirect these impressions to managed campaigns or other ad marketplaces for re-selling. This reduces the risk of over-optimisation. Comparing with the original payoff function, the case $r'(t) = \alpha(t)$ provides an extra gain, whereas the case $r'(t) = 0$ incurs a loss.

This section addresses the problem at the placement level; in other words, reserve prices are adjusted periodically (instead for every impression). From a control theory's point of view, throughout the planning horizon the publisher wants to explore (learn) the optimal $\alpha$ of the placement; the same time the publisher also exploits (predict) the known best $\alpha$ to get as much payoff as possible and to compensate the loss.

### 5.3.1 Optimal Auction Theory

Bidders are encouraged to bid their private values in the second price auctions [36, 172] when a reserve price is absent. Note that this dominant strategy does not hold in modern sponsored search where *quality scores* are used [8] in ad ranking. Without quality scores, the strategy of bidding at the private value forms part of the Nash equilibrium of the system, meaning as time elapses advertisers have no incentive to change their bids, given that all other factors remain the same. In this *non-cooperative game* [33], the winner could, but would not, lower his bid to let other competitors win because losing the auction is not beneficial in either short-term or long-term (lowering the bid while still winning has no effect since the winner always pays the second highest bid).

Suppose the publisher knows the bidders' private value distribution. The optimal auction theory mathematically defines the optimal reserve price [160, 173, 38, 36]. Here let us briefly review the theory to make the paper self-contained [33]. Again suppose there are $K$ bidders, and they are risk-neutral and symmetric, i.e. having identical value distributions. Each bidder $k \in K$ has private information on the value

of an impression, drawn from distribution $F_k(x)$, where $F_k(x)$ denotes the probability that the advertiser's private evaluation value is less than or equal to a certain number $x$. Usually it is assumed Log-normal [37] or Uniform distribution [36]. Assuming private values are independently distributed, the distribution over value vector is

$$F(\cdot) = F_1(\cdot) \times \cdots F_K(\cdot),$$

and then the optimal reserve price is given as:

$$\alpha = \frac{1 - F(\alpha)}{F'(\alpha)} + v_P, \tag{5.2}$$

where $F'(\alpha)$ is the density function, the first order derivative of $F(\alpha)$ and $v_P$ is the publisher's private value. In practice, $v_P$ could be obtained from a guaranteed contract with a flat CPM, or from another ad network where the average revenue is known.

In the experiments this theory was implemented as OPTAUC and the Log-normal distribution assumption of bidders' private values was followed. The symmetric assumption was also adopted, i.e., there was only one distribution for all bidders. Under these assumptions, the optimality of the auction under GSP is proved in [154]. The estimation of Log-normal's mean and standard deviation was obtained using the training dataset (impression-level logs from 14 Dec 2012 to 18 Jan 2013).

#### 5.3.1.1 Drawbacks in RTB Practice

In practice, there are drawbacks of the optimal auction theory mostly due to the difficulty of learning bidders' private values, i.e., $F(x)$. Firstly, a bidder could have a complex private value distribution for impressions. In RTB, an advertiser computes a bid for each impression based on the contextual [24] and behavioural [26] data. This makes the bidding algorithms more based on a regression model than a simple parametric distribution. Besides, the bidding algorithms or regression models are never disclosed to publishers or other advertisers. This is especially true and important in RTB. On the contrary, in sponsored search, search engines run bidding algorithms for advertisers and host auctions as a publisher at the same time. This allows full access of information. Also, in sponsored search the auctions are based on keywords, so the population of the bidders are relatively stable; whereas, in RTB, the auctions are at the impression level and the advertisers are more flexible in participation.

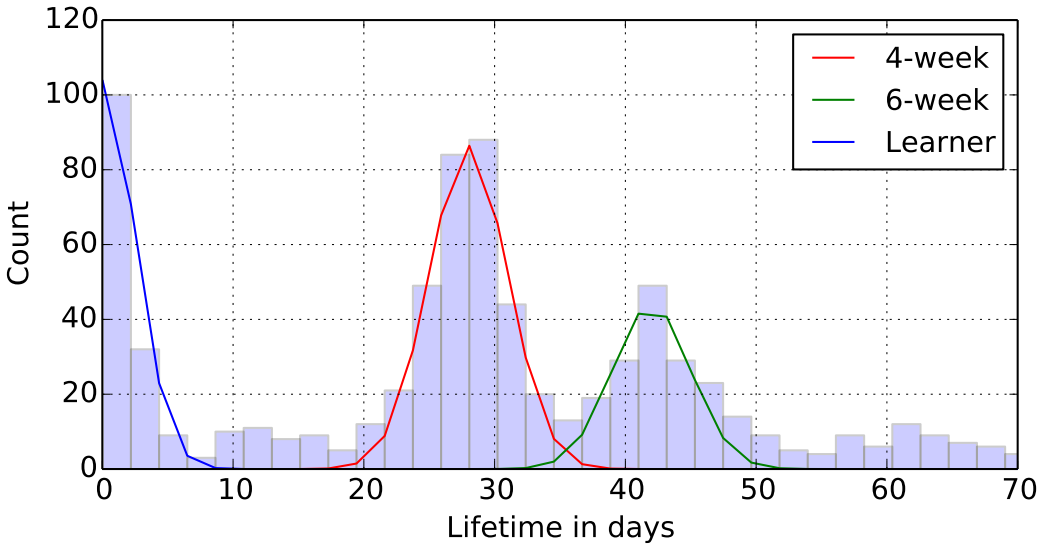


**Figure 5.15:** The histogram of 755 bidders' lifetime across all placements. Each campaign is considered a bidder in the study. The figure unveils three patterns: two created by common flight-time of campaigns (four weeks and six weeks) and one created by exploration of learning algorithms (0-2 days). Due to limited time of data collection there are some instances lying between 0-25 days because these bidders started their activities before the start date of experiments.

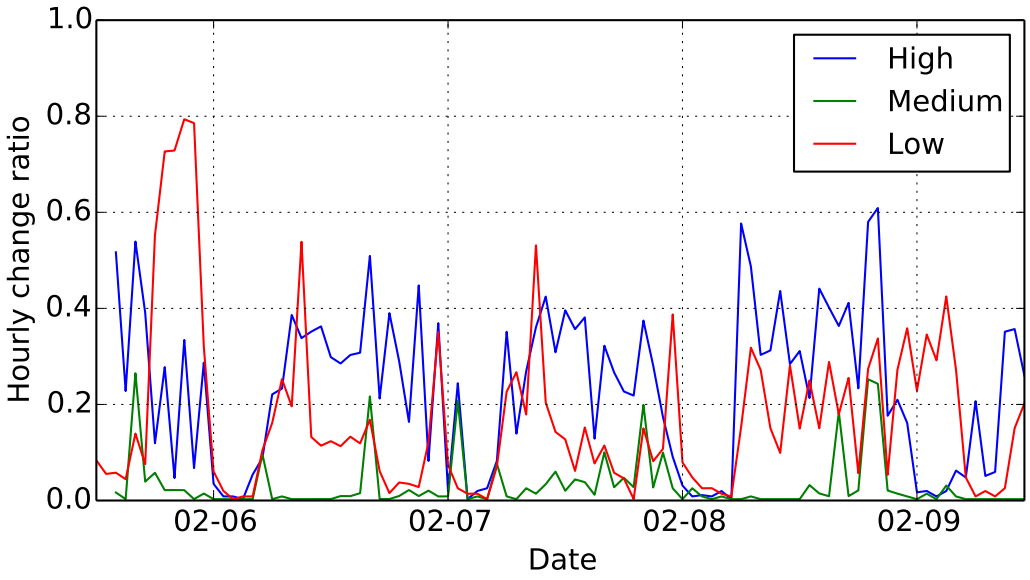


**Figure 5.16:** The change of winners for placements with different levels of competition in four days. The fact that a bidder does not always win could add difficulty to reserve price detection if undisclosed. The result also implies the change rate does not necessary relate to the competition level.

The Uniform distribution assumption at placement level and Log-normal distribution at both placement and impression level were tested. Although these distributions are widely adopted in research literature [36, 37], only a small portion of tests returned positive results as shown in Figures 5.7 and 5.8.

Secondly, it is assumed that advertisers bid at their private values in the second price auction [160, 37]. However, in practice, an advertiser may not know his private valuation of an impression. Instead, he often wants to achieve the best possible performance. Also in different stages (learning, prospecting, retargeting, etc.) of an advertising campaign, the bidding strategy may change. This makes the bidding activity vary considerably across the limited flight time of a campaign. See a plot from the experiments in Figure 5.15.

Thirdly, there are other practical constraints including accessibility of auction details, noise introduced by the frequent change of auction winners, c.f. Figure 5.16. I believe these drawbacks lead to undesirable performance of an optimal auction theory based algorithm in the real-world experiments.

### 5.3.2 A Simplified Dynamic Game

To address the above-mentioned issue, this thesis takes an alternative view and proposes a simple auction game between the publisher and auction winners and identify the dominant strategies. The game could be simplified by dropping the *repeated* nature of auctions. Thus, the publisher only considers the current auction and do not learn the private values from historical knowledge. In fact, the result of this simplification follows the instinct and is easy to implement as shown later; it also performs the best in most of the cases in the online large scale real-world experiments.

First let us write down the extensive form representation of this dynamic game:

- Player: the winner of auctions (advertisers) $w$ and the publisher $p$.
- The information set $\mathbb{I}$ before acting is the same for the winner and the publisher. It has two decision nodes:
  $I_1$, the winning bid $b$ is equal to or higher than the current reserve price $\alpha$;
  $I_2$, the winning bid is lower than the reserve price.
- The action set of the winner $A_w$:
  $a_{w1}$, to increase $b$ to higher than $\alpha$;
  $a_{w2}$, to increase $b$ to lower than $\alpha$;
  $a_{w3}$, to decrease or hold $b$ to higher than $\alpha$;
  $a_{w4}$, to decrease or hold $b$ to lower than $\alpha$.
- The action set of the publisher $A_p$:

$a_{p1}$, to increase or hold $\alpha$ to higher than $b$;

$a_{p2}$, to increase or hold $\alpha$ to lower than $b$;

$a_{p3}$, to decrease $\alpha$ to higher than $b$;

$a_{p4}$, to decrease $\alpha$ to lower than $b$.

- The sequence of move: first the publisher, then the winner.

The game tree representation, as well as the payoff function, is given in Figure 5.17. Note for some nodes the payoff of the winner consists two numbers, e.g., $30/-10$, when the winner chooses to increase the bid, i.e., the action $a_{w1}$. The positive value stands for the case where it is still profitable to raise the bid, while the negative value stands for the possible loss if raising the bid, since the advertiser has reached the maximum affordable price (i.e., the revenue of sales through the ad). In the latter case, an advertiser would choose other actions like $a_{w2}$ or $a_{w4}$. These values have been carefully selected to reflect the positions of bidder. For example, $I_1 \rightarrow I_1|a_{p2} \rightarrow I_1|a_{w1}$ would give less payoff to the advertiser than $I_1 \rightarrow I_2|a_{p1} \rightarrow I_1|a_{w1}$ because he has been increasing bids to win auctions.

When deducing dominant strategies it is assumed that both cases happen with an equal chance since both publishers and advertisers do not use historical knowledge but only the last state. Therefore, payoff of the publisher in these cases is discounted by $0.5$ since a rational advertiser would not choose a negative payoff.

#### 5.3.2.1 The Publisher's Dominant Strategy

This section analyses the case where the publisher and the winner play the game for only one round. The dominant strategy for the publisher is:

$$s_p^*(I) = \begin{cases} a_{p2}, & \text{if } I = I_1 \\ a_{p4}, & \text{if } I = I_2 \end{cases} \tag{5.3}$$

which gives the expected payoff:

$$R(s_p^*|I) = \begin{cases} 60 & \text{if } I = I_1 \\ 40 & \text{if } I = I_2 \end{cases} \tag{5.4}$$

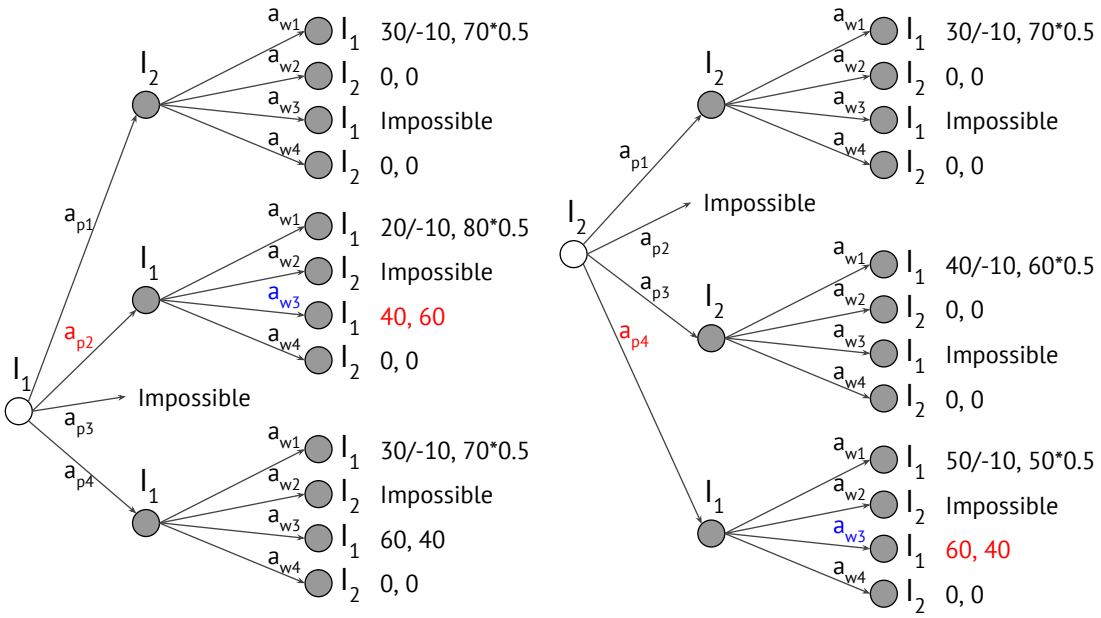


**Figure 5.17:** The game between the winner and the publisher in the reserve price problem. The publisher moves first, then the winner. Players are not drawn. Numbers are chosen carefully to reflect the positions of players rather than the absolute revenue. $\{I_1, I_2\}$ represent the information set. For the first round when the publisher has to start randomly, or set a reserve price to his private valuation of the impression. The leaf nodes give the result information set as well as the payoffs of (winner, publisher). Note for the action $a_{w1}$ the payoff of the winner could be negative if he has already been bidding the maximal affordable price. This model assumes these cases happen at a chance of $50\%$ due to no utilisation of historical data. Thus, the payoff of the publisher is discounted by $0.5$ in these cases.

The proof is eliminated here since it could be easily acquired by following nodes iteratively in the game tree, c.f. Figure 5.17. This strategy will serve as the foundation of the ONESHOT algorithm proposed in the following section. On the other hand, the dominant strategy for the winner (advertiser) is

$$s_w^*(I) = \begin{cases} a_{w3}, & \text{if } I = I_1 \\ a_{w1}, & \text{if } I = I_2 \end{cases} \tag{5.5}$$

which indicates that the bid price should be gradually reduced but increased again when lost the auction.

In sum, these strategies indicate the publisher should keep the reserve price below the winning bid while trying to approach the winning bid gradually. If the reserve price is too high, it should be reduced drastically to make sure it will be below the winning

**Table 5.2:** The payoff matrix of the publisher if the new winner bid randomly. $I_1$ and $I_2$ denote the starting information set. $a_{p*}$ stands for publishers' actions. The publisher's dominant strategy remains the same and is illustrated in bold font.

| | $I_1$ | $I_2$ |
|---|---|---|
| $a_{p1}$ | 1/3 * 70 | 1/3 * 70 |
| $a_{p2}$ | **1/3 * 80 + 1/3 * 60** | - |
| $a_{p3}$ | - | 1/3 * 60 |
| $a_{p4}$ | 1/3 * 70 + 1/3 * 40 | **1/3 * 50 + 1/3 * 40** |

bid. Following these actions, the final state of the system comes to the second price auction without a reserve price, i.e., the reserve price is always equal to the second highest bid. In this state, the publisher does not need to lower $\alpha$ anymore. The winner will not lower the bid otherwise he would lose the auction that gives a zero payoff. However, this state has never been observed in experiments due to the frequent change of bidders, variation of bids, possibly different strategies adopted by advertisers (e.g. always bid at the private value).

#### 5.3.2.2 Introducing the Randomness

One may argue that since winners of auctions are constantly changing this is no longer a game between two players, but between a player and a group. The frequency of changing is indeed high, c.f. Figure 5.16. However, since the knowledge of auctions is not shared among the bidder group, each player of this group only possesses an imperfect information set. This could result in the randomness of winner's actions: the new winner does not have the outcome of the last auction, so has to bid randomly, which is usually based on private valuation of the impression or simply instinct.

In this case, the dominant strategy of the publisher is still the one defined in Equation 5.3. This case removes the negative payoffs of the winner from the game tree in Figure 5.17 and allows him to bid randomly regardless of what action has been chosen by the publisher. The negative payoffs are removed because the new winner should not bid above his private valuation of the impression in the first-time step. The single-step payoff matrix of the publisher is shown in Table 5.2.

By playing this auction game with advertisers' random actions the publisher has

the payoff:

$$R'(s_p^*|I) = \begin{cases} \frac{1}{3} * 80 + \frac{1}{3} * 60, & \text{if } I = I_1 \\ \frac{1}{3} * 50 + \frac{1}{3} * 40, & \text{if } I = I_2 \end{cases} \tag{5.6}$$

This case describes the real-world marketplace more closely. In fact, the convergence to the second price auction state hardly happens due to two reasons. Firstly, the winner of auctions changes constantly as illustrated in Figure 5.16, especially when there are many bidders in auctions. Secondly, the detection of the reserve price could be difficult (when it is not disclosed) and costly. It becomes even harder if certain randomisation is introduced to the final reserve price setting. The winner could suffer from distinguishing a reserve price from his competitors' bids. From the bidder's perspective, the dominant strategy of competing with other bidders (bidding the private value) and with the publisher (keep lowering the bid before losing) is clearly different.

#### 5.3.2.3 The OneShot Algorithm

This section designs an algorithm based on the dominant strategy analysis above. For the publisher, if the winning bid is higher than the reserve price, slowly increase the reserve price; otherwise, decrease drastically. The actual implementation is slightly different that it introduces parameters to control the magnitude of the change under different situations. The equations are

$$\begin{cases} \alpha(t+1) = (1 - \epsilon^t \lambda_h) a(t) & \text{if } \alpha(t) > b_1(t) \\ \alpha(t+1) = (1 + \epsilon^t \lambda_e) a(t) & \text{if } b_1(t) \geq \alpha(t) \geq b_2(t) \\ \alpha(t+1) = (1 + \epsilon^t \lambda_l) \alpha(t) & \text{if } b_2(t) > \alpha(t) \end{cases}$$

where $\epsilon \in (0, 1]$ and $\lambda_h, \lambda_e, \lambda_l \in [0, 1]$. $\epsilon$ is a decay factor w.r.t. time, allowing the reserve price to converge if needed. $\lambda_h$ controls the cooling speed when the reserve price is too high; $\lambda_e$ controls the continued exploration when the reserve price is successful; $\lambda_l$ controls the heating up speed when the reserve price is too low.

Note the values of these parameters depend on targeting combinations (place-

ments, date and time, geography, etc.). They need to be tuned to achieve the best result. For example, in the experiments an effective setting of these parameters was $\epsilon = 1.0$, $\lambda_h = 0.3$, $\lambda_e = 0.01$, and $\lambda_l = 0.02$ for placement 834119 and hour-of-day=8. These parameters were obtained using the training dataset (impression-level logs from 14 Dec 2012 to 18 Jan 2013). This algorithm is denoted as ONESHOT in the experiments.

### 5.3.3 Other Private Value Based Algorithms

Although not optimal in theory, there are other algorithms that make use of private value distributions. Here two simple methods based on Bayes' rules are described. An advantage of such algorithms is that they could be easily tuned to be more aggressive or less according to the greediness (or risk preference) of the publisher. A disadvantage is that it does not take the future into account, so only the current payoff is maximised.

#### 5.3.3.1 Bivariate Log-normal Distribution

To make the algorithm easier to understand let us keep the Log-normal distribution assumption of private values. A different distribution changes the formulae but the idea and derivation still hold. Here, the first and second highest bids are captured using a bivariate Log-normal distribution:

$$\begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \sim \text{lognorm}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{5.7}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ stand for mean and variance respectively. For simplicity let us consider $\boldsymbol{\Sigma}$ known and use a bivariate Gaussian distribution as the conjugate prior for $\boldsymbol{\mu}$:

$$\boldsymbol{\mu}(t) \sim \mathcal{N}(\boldsymbol{\theta}(t), \boldsymbol{\Delta}(t))$$

The priori could be learned using historical data. In the experiments, impression logs from 14 Dec 2012 to 18 Jan 2013 were used. During evaluation, each time the publisher observes the highest and 2nd highest bids in an auction, then updates the

belief using Bayesian inference:

$$\begin{bmatrix} B_1 \\ \\ B_2 \end{bmatrix} | \boldsymbol{b}(t) \sim \text{lognorm}(\boldsymbol{\theta}(t+1), \boldsymbol{\Delta}(t+1) + \boldsymbol{\Sigma})$$

where

$$\begin{aligned} \boldsymbol{\theta}(t+1) &= (\boldsymbol{\Delta}^{-1}(t) + \boldsymbol{\Sigma}^{-1})^{-1}(\boldsymbol{\Delta}^{-1}(t)\boldsymbol{\theta}(t) + \boldsymbol{\Sigma}\boldsymbol{b}(t)) \\ \boldsymbol{\Delta}(t+1) &= (\boldsymbol{\Delta}^{-1}(t) + \boldsymbol{\Sigma}^{-1})^{-1} \end{aligned}$$

Then with $p(B_1, B_2)$ estimated at current time $t$, a reserve price which has the maximum probability of sitting between the two bids can be solved:

$$\alpha(t)^* = \arg\max_{\alpha(t)} P(B_1 \geq \alpha(t) \geq \lambda B_2) \tag{5.8}$$

where $\alpha(t)^*$ is the current reserve price and $\frac{B_1}{B_2} \geq \lambda \geq 1$ is the risk preference parameter. It pushes the result closer to the winning bids. In the experiments it was $\lambda = 1$ to mimic a risk-averse choice.

The maximum could be achieved by letting the first derivative equal zero. A numerical solution was used and the result was approximated in the algorithm's implementation using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, the most popular one in Quasi-Newton methods class. The BFGS method uses the first derivative and has proven good performance for smooth optimizations [174] and. This algorithm is denoted as BAYESIANB in experiments.

#### 5.3.3.2 Univariate Log-normal Distribution

The most greedy choice from Equation 5.8 is to let $\lambda = \frac{B_1}{B_2}$: regardless of the second highest bidder, the publisher could always learn and approach the highest bid directly. It is equivalent to modelling the winning bids using a univariate Log-normal distribution, which has a simpler solution

$$\begin{aligned} B_1 &\sim \text{lognorm}(\mu, \sigma^2) \\ \mu(t) &\sim \mathcal{N}(\theta(t), \delta^2(t)) \end{aligned}$$

and still it is assumed that $\sigma^2$ is known. Every time an observation on the highest bid $b_1$ gives

$$\theta(t+1) = \frac{\theta(t)\delta^2(t) + \sigma^2 b_1}{\sigma^2 + \delta^2(t)}$$
$$\delta^2(t+1) = \frac{\delta^2(t)\sigma^2}{\sigma^2 + \delta^2(t)}$$

where the priori was obtained using the training dataset.

Then the publisher simply chooses

$$a(t+1) = \log(\theta(t+1)) + \beta(\sigma^2 + \delta^2(t+1)) \tag{5.9}$$

as the new reserve price where $\beta$ is the risk preference for this model which could be either positive or negative. In the experiments it was $\beta = 0$ for simplicity.

This algorithm is denoted as BAYESIANU. Again, the fitting of private value distribution is beyond the scope of this paper but the ideas and algorithms always hold, allowing different models to be implemented easily (it is highly likely that different models would be used for different placements, hour-of-day, and etc.).

### 5.3.4 Private Value Free Algorithms

There are algorithms based on conventional wisdom or business intuition. They do not assume that each advertiser has their own private valuation of the impression. In fact, they play a significant role in reserve price setup in the real-world. This section presents two simple methods as baselines among many choices.

The most essential one is the FIXED algorithm which sets a static reserve price for all time steps, regardless of the outcome of auctions. Formally, it is $\alpha(t) \equiv a$ where $a$ denotes the pre-chosen fixed reserve price.

When $\alpha = 0$ it becomes ZERO which totally relies on the auction mechanism, here second price auction, assuming that quality score (or similar factors like bid bias) is absent. It is apparently the most altruistic one in the strategy space which does not try to gain any extra benefit and could be used as a baseline to measure bidders' attrition.

The FIXED (including ZERO) is the most straightforward and easy to implement. Besides, it directly reflects the publisher's private valuation of the inventory. Before they adopt the automated selling, publishers are used to negotiating contracts with ad-

vertisers or their representatives. The contract price could be easily converted to a reserve price. Also, due to the simplicity this is probably the most widely adopted algorithm in today's ad marketplaces. In the experiments, FIXED and ZERO were used as the most naive baselines. However, although simple, under certain circumstances they performed surprisingly well.

Another family of private-value-free algorithms is based on historical payoff of auctions. For example, the AVERAGE sets the reserve price to the average of past payoff. Formally it is

$$\alpha(t) = \frac{1}{M} \sum_{i=t-M}^{t-1} r(i) \tag{5.10}$$

where $r(t)$ is the payoff at time step $t$ and $M$ is the averaging window.

A natural extension is the weighted average variation, which values more of recent payoff. Formally, it is

$$\alpha(t) = \frac{1}{M} \sum_{i=t-M}^{t-1} w(i,t) r(i) \tag{5.11}$$

where $w(\cdot)$ is the weighting function, which could take various forms. In the experiments, linear weights were used and the algorithms is denoted WEIGHTEDL.

AVERAGE and WEIGHTEDL share the similar insights with FIXED, except recognizing the fluctuation of demand and supply in the market. This fluctuation may affect the reserve price setting especially at hour-of-day level.

## 5.4 Online Test Results and Discussions

The online experiments were carried out in the same production platform as the empirical study. The experiments include 25 placements as the treatment group and additional eight as the control group. For each placement, RTB auction logs were collected which record bidders and their bids in every auction. The live experiments ran from 19th Jan to 21st Feb 2013. The parameters of algorithms were trained using logs from 14th Dec 2012 to 18th Jan 2013.

### 5.4.1 Algorithms and Scheduling

The following algorithms and configurations were evaluated.

- FIXED, with publisher value set to \$1.0;
- ZERO, with a fixed reserve price set to \$0.0;
- AVERAGE;
- WEIGHTEDL, with a looking-back-window-size of five;
- ONESHOT, with $\epsilon = 1$ and $\lambda_h, \lambda_m$, and $\lambda_l$ learned individually for each placement;
- BAYESIANB, with $\boldsymbol{\theta}$, $\boldsymbol{\Sigma}$, and $\boldsymbol{\Delta}$ learned individually for each placement;
- BAYESIANU, with $\theta, \sigma$, and $\delta$ learned individually for each placement.

Note that the private value of \$1.0 was based on the publisher's suggestion.

Algorithms were evaluated as the following: impression-level logs were continuously sampled then fed to algorithms. The experiments used a Round Robin scheduler at the hour level. For example, at Hour=1 and Day=1 Algorithm=OPTAUC was used on Placement=1, Algorithm=ZERO was used on Placement=2, etc.; on Hour=2 and Day=1 Algorithm=BAYESIANU was used on Placement=1, Algorithm=OPTAUC was used on Placement=2, etc. This scheduling ensures that the day-of-week factor does not influence the evaluation of performance of algorithms. In this way every (hour-of-day, algorithm, level of competition) tuple generated ten sets of observations. The results were split into six non-overlapping chunks and the Wilcoxon signed-rank test [175] was used to check the significance.

### 5.4.2 Results

This section reports the performance of algorithms with respect to levels of competition in Figure 5.18. In sum, the private-value-free algorithms performed better than private-value-based ones. Considering the percentage of placements of high, medium, low level of competition, the ONESHOT performed significantly better in about 70% cases. This is very consistent with the findings made in [37].

Looking at the results and analysis made before, there is a good reason to believe that the undesirable fitting of Log-normal distribution led to the poor performance of private-value-based algorithms on placements with many bidders. Although there is a steady hourly average winning bids pattern on these placements, there are also lots

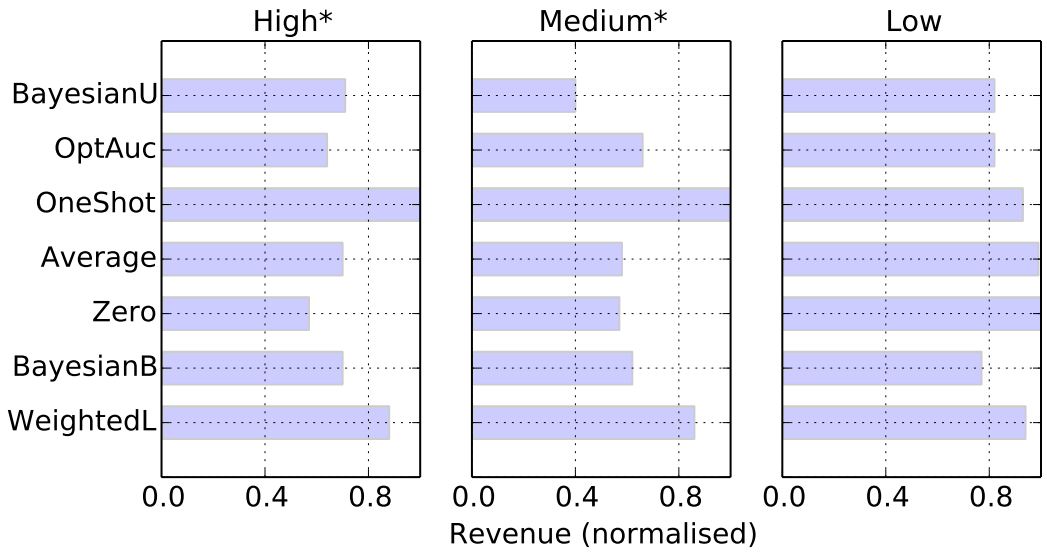


**Figure 5.18:** A snapshot of main results of the online experiments, on all placements, from 6am to 10am across all days. The $*$ indicates the OneShot performed significantly better than others (except the WeightedL) using Wilcoxon signed-rank test. Considering the percentage of placements of high, medium, low level of competition, the OneShot performed significantly better in about 70% cases. On placements having fewer bidders, it is hard to distinguish the results. Although not reported, at other hours the results are similar.

**Table 5.3:** The change of number of bidders after the experiments. On most of placements, especially ones with high level of competition, the increment is observed

| | $\leq$ -10% | -5% $\sim$ -10% | 5% $\sim$ 10% | $\geq$ 10% |
|---|---|---|---|---|
| High | 3 | 0 | 0 | 3 |
| Medium | 4 | 2 | 4 | 6 |
| Low | 3 | 0 | 0 | 8 |
| Control | 1 | 0 | 0 | 5 |

of short periods of burst (both upward and downward) around the curve as shown in Figure 5.9, which could have dragged the models away from the curve easily. On the contrary, the private-value-free algorithms had a better chance of capturing these bursts.

The ONESHOT shares the same intuition as the WEIGHTEDL: using the observations of the recent future to infer the reserve price. More specifically, the auction logs were replayed in training dataset against various parameter sets and adopted the best-performing one.

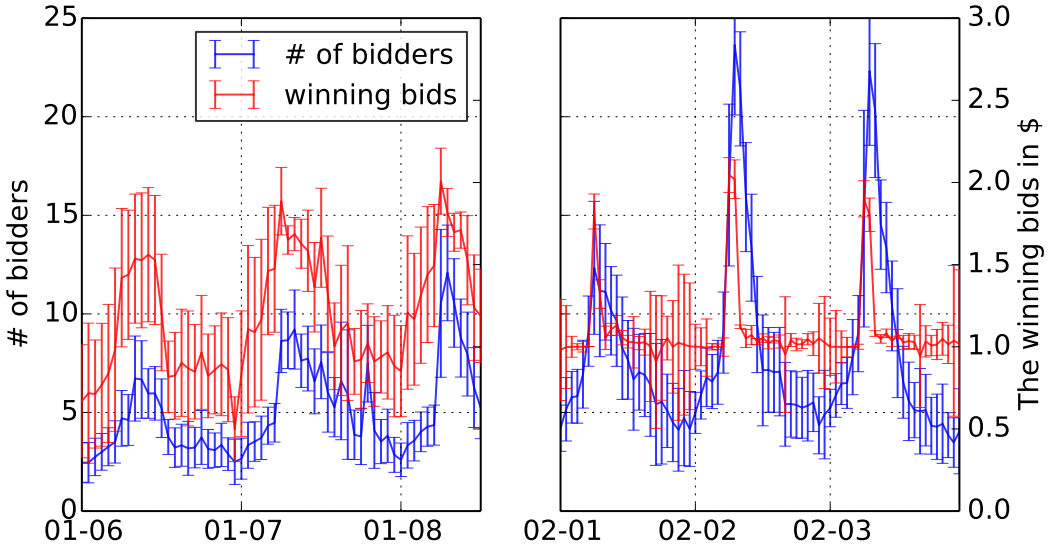


**Figure 5.19:** The comparison of winning bids and number of bidders across all placements before and after the experiments. Both factors increased after the experiment, which contradicts the attrition hypothesis that bidders would reduce their bid or volume after reserve price optimisation.

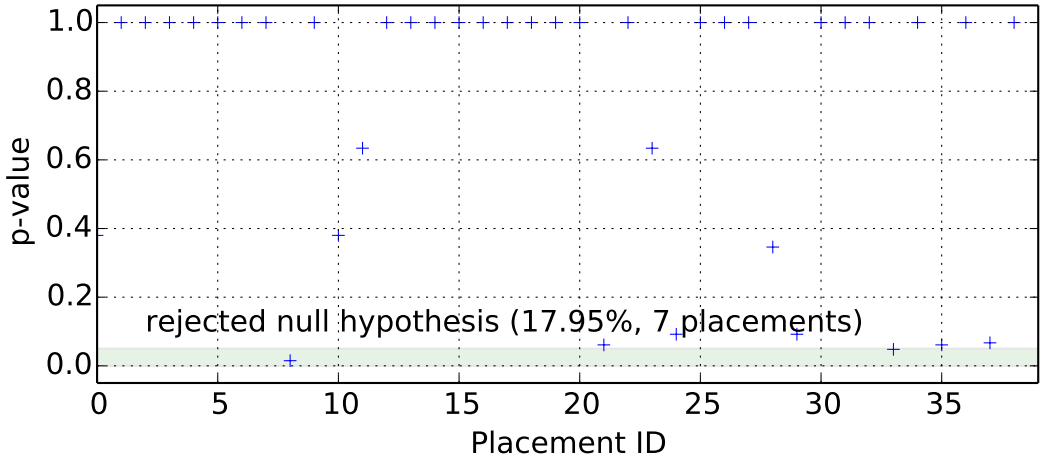


**Figure 5.20:** Bidders' attrition test after the optimisation, comparing with the number of bidders before the experiments. The t-test suggests the attrition was significant on seven placements. Three are placements of low level of competition and four of medium level. However, many more placements see an increment of the number of buyers after the experiments, c.f. Table 5.3.

### 5.4.3 Bidders' Attrition

As argued before, a key difference of reserve price optimisation in RTB is that the inventory is much less limited than ordinary auctions or sponsored search. Thus, it is important to study whether the reserve price optimisation could result in bidders' attrition [33] in a long term (e.g., reduce their bid values or volume). Figure 5.19 compares winning bids and number of bidders before and after the reserve price optimisation

experiments. The result suggests the insignificant attrition.

To test the statistical significance of the change, the mean and standard deviation of the number of bidders were taken for each placement before (12-18 Jan 2013) and after (15-21 Feb 2013) the optimisation experiments. The null hypothesis of the buyers' attrition could be denoted as $H_0$: $\mu_1 - \mu_2 > 0$ where $N_1(\mu_1, \sigma_1)$ and $N_2(\mu_2, \sigma_2)$ are the Normal distribution fitted from number of bidders before and after the experiments respectively. Its rejection implies insignificance of bidders' attrition.

The result of t-test with p-value 0.05 is summarised in Figure 5.20. There are only seven placements out of 39 showed a significant drop of the number of bidders after the experiments. 3 of them are placements of low level of competition and 4 of medium level, where a slight change of would be significant. Interestingly, Table 5.3 unveils a opposite pattern: the publisher was seeing more bidders on most of the placements. This could be due to three reasons: 1) during holidays (the beginning of the experiments), there were fewer bidders because of a reduced amount of Internet traffic; 2) bidders were not aware of the reserve price optimisation, so no reaction is taken; 3) the campaigns' lifetime introduces fluctuation.

## 5.5 Conclusions

This chapter discusses a specific problem for publisher revenue optimisation in Real-Time Bidding: to find the optimal reserve price for single-item display ad auctions. It is a problem of significant importance, which has been studied extensively in the game theory, but is relatively new in online advertising research literature, especially in RTB that has many unique features. This chapter analyses drawbacks of the optimal auction theory in the RTB practice, derives dominant strategies from the one-shot version of the auction game, and compares it with other commonly adopted algorithms in online experiments in a production platform. The challenges of optimising reserve prices in RTB are also discussed and bidders' attrition in the experiments was analysed. Due to the complexity of the problem and practical constraints there are unsolved issues, including the fitting of bids, parameters tuning, and more comprehensive attrition analysis. These are left to the feature works.

To carry out the online experiments a software system was developed to perform scheduled data feeds collection, parsing, cleaning, and saving to a database, computing

reserve prices using pre-defined algorithms, and applying the result via API services. To make this complex process easier to follow, with the support from UCL Advances, I describe the design and implementation of the system in Appendix B.

# Chapter 6

# Conclusions and Future Works

Overall, the thesis has proposed an optimisation framework for the supply side. The main objective of optimisations is to achieve the maximum revenue in the long term (usually constrained by a finite planning horizon). This thesis studies the online display advertising ecosystem including its history and structure (Figure 1.1), processes in Real-Time Bidding (RTB, Figure 1.2), and the cash flow (Figure 1.3). Following the life cycle of ad-supported websites managed by publishers (the supply side), several optimisation problems have been investigated including ad density optimisation, ad source selection, and reserve price optimisation. For each problem, proposed innovative models and algorithms have been proposed which then have been evaluated in large-scale experiments.

To solve the ad density optimisation problem, a game theoretic optimal control model has been built which considers the competition of similar publishers. Comparing with the case without competition (social welfare maximisation) shows the competition leads to global sub-optimal solution, which results in Internet users suffering from seeing more ads. This finding aligns well with the *tragedy of the commons* theory in economics. Besides, the empirical study has shown the general increasing trend of online display ads on the Internet, as well as the correlation of ad density with many other factors.

To solve the ad channel selection problem, a Partially Observable Markov Decision Process (POMDP) model has been proposed which uses correlation of candidates to accelerate exploration. This thesis has presented both the exact solution using value iteration and approximation using a multi-armed bandit model. Also, the thesis includes a toy example to showcase the calculation process and importance of explo-

ration. Comparing with other popular algorithms on a real-world dataset shows the advantage of proposed algorithms that are mainly contributed by the correlated belief update. These have been published as [114] and [176].

To solve the reserve price optimisation problem, heuristics derived from a simplified game model have been proposed. The model considers the competition between the buyer and the seller to be *one-shot* instead of *repeated*. In this way, the model avoids the private value distribution fitting hurdle which virtually results in the poor performance of optimal auction theory in RTB. The large scale online experiments have validated the better performance of the proposed model, too. These have been published as [177] and [47].

To make the thesis easier to understand, a list of terminologies has been included in Appendix A. Also, the implementation details of the reserve price optimisation prototype system have been included in Appendix B, hoping to bridge the gap between our research and the industry.

During the development of the thesis, a lot of interesting ideas and issues have emerged. However, due to limited time and resources, only a few of them have been discussed. Here I summarise the possible directions of the future works:

- To combine the ad density control with the reserve price optimisation; to use the *users' satisfactory* as the lower bound of the reserve price. I.e., reject the bid if it cannot compensate for the loss of users' satisfactory in the long term;

- To investigate a wider range of probabilistic distributions for bids in RTB auctions, and their impact on the reserve price optimisation;

- To integrate the attrition metrics into the reserve price optimisation algorithms. I.e., do not set the reserve price too high (even if auctions still hold) if it drives away buyers in the long term;

- To study the statistical arbitrage among possible ad sources. I.e., bidding for an impression in one ad source and selling it immediately in another;

- To use users' segments as key features of selling impressions, especially for setting reserve prices;

- To develop algorithms of packaging impressions into guaranteed or non-guaranteed deals, and optimise the allocation of selling in advance and selling in RTB.

# Bibliography


[1] Interactive Advertising Bureau. IAB Internet Advertising Revenue Report 2013, 2014.

[2] Andrei Broder. Computational advertising and recommender systems. *Proceedings of the 2008 ACM conference on Recommender systems SE - RecSys '08*, pages 1–2, 2008.

[3] T Graepel, T Borchert, and R Herbrich. Probabilistic Machine Learning in Computational Advertising. In *Proceedings of PASCAL - Pattern Analysis, Statistical Modelling and Computational Learning*, 2009.

[4] Ao-Ying Zhou, Min-Qi Zhou, and Xue-Qing Gong. Computational Advertising: A Data-Centric Comprehensive Web Application. *Chinese Journal of Computers*, 34:1805–1819, 2011.

[5] Lizhen Xu, Jianqing Chen, and Andrew Whinston. Interplay Between Organic Listing and Sponsored Bidding in Search Advertising. *cCombs Research Paper Series No. IROM-13-09*, (403), 2009.

[6] Google. Choose an ad format - AdWords Help, 2014.

[7] David S Evans. The Online Advertising Industry: Economics, Evolution, and Privacy, 2009.

[8] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. Technical report, National Bureau of Economic Research, 2007.

[9] Interactive Advertising Bureau. IAB Display Advertising Guidelines, 2012.

[10] Google. The Arrival of Real-Time Bidding, 2011.

[11] Koenw De Bock and Dirk Van Den Poel. Predicting website audience demographics forweb advertising targeting using multi-website clickstream data. *Fundamenta Informaticae*, 98(1):49–70, 2010.

[12] Mikhail Bilenko and Matthew Richardson. Predictive client-side profiles for personalized advertising. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11*, page 413, 2011.

[13] Myriam Abramson. Toward the attribution of web behavior. In *2012 IEEE Symposium on Computational Intelligence for Security and Defence Applications, CISDA 2012*, 2012.

[14] Guillaume Roels and Kristin Fridgeirsdottir. Dynamic revenue management for online display advertising. *Journal of Revenue and Pricing Management*, 8(5):452–466, May 2009.

[15] Erik Vee, Sergei Vassilvitskii, and Jayavel Shanmugasundaram. Optimal online assignment with forecasts. In *Proceedings of the 11th ACM conference on Electronic commerce - EC '10*, pages 109–118, 2010.

[16] Vijay Bharadwaj, Wenjing Ma, Michael Schwarz, Jayavel Shanmugasundaram, Erik Vee, Jack Xie, and Jian Yang. Pricing Guaranteed Contracts in Online Display Advertising. *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 399–408, 2010.

[17] Vijay Krishna. *Auction Theory*. 2002.

[18] K. Zhang and Z. Katona. Contextual Advertising, 2012.

[19] Berthier Ribeiro-neto, Belo Horizonte, Marco Cristo, Paulo B Golgher, Caram Pampulha, and Edleno Silva De Moura. Impedance coupling in content-targeted advertising. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 496–503, 2005.

[20] Wen-tau Yih, Joshua Goodman, and Vitor R. Carvalho. Finding advertising keywords on web pages. In *Proceedings of the 15th international conference on World Wide Web - WWW '06*, page 213, 2006.

[21] Aris Anagnostopoulos, Andrei Z. Broder, Evgeniy Gabrilovich, Vanja Josifovski, and Lance Riedel. Just-in-time Contextual Advertising. In *Cikm*, page 331, New York, New York, USA, 2007. ACM Press.

[22] Xiaoyuan Wu and Alvaro Bolivar. Keyword extraction for contextual advertisement. In *Proceeding of the 17th international conference on World Wide Web - WWW '08*, page 1195, 2008.

[23] Anísio Lacerda, Marco Cristo, Marcos André Gonçalves, Weiguo Fan, Nivio Ziviani, and Berthier Ribeiro-Neto. Learning to advertise. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval SIGIR 06*, page 549, 2006.

[24] Andrei Broder, Marcus Fontoura, Vanja Josifovski, and Lance Riedel. A semantic approach to contextual advertising. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '07*, page 559, 2007.

[25] Deepayan Chakrabarti, Deepak Agarwal, and Vanja Josifovski. Contextual advertising by combining relevance with click feedback. In *Proceeding of the 17th international conference on World Wide Web - WWW '08*, page 417, 2008.

[26] Jun Yan, Ning Liu, Gang Wang, and Wen Zhang. How much can behavioral targeting help online advertising? *Proceeding WWW '09 Proceedings of the 18th international conference on World wide web*, pages 261–270, 2009.

[27] Foster Provost, Brian Dalessandro, and Rod Hook. Audience Selection for Online Brand Advertising : Privacy-friendly Social Network Targeting. In *Brand*, pages 707–715, 2009.

[28] Xiaohui Wu, Jun Yan, Ning Liu, Shuicheng Yan, Ying Chen, and Zheng Chen. Probabilistic Latent Semantic User Segmentation for Behavioral Targeted Ad-

vertising *. In *Proceedings of the Third International Workshop on Data Mining and Audience Intelligence for Advertising*, pages 10–17, 2009.

[29] Google. Check and understand Quality Score - AdWords Help, 2012.

[30] Jun Wang and Bowei Chen. Selling Futures Online Advertising Slots via Option Contracts. In *Www*, volume 1000, pages 627–628, 2012.

[31] As Dilling and Cb O'kelley. Advertising Bid Price Modifiers, 2011.

[32] G Hardin. The tragedy of the commons. *Science (New York, N.Y.)*, 162:1243–1248, 1968.

[33] Martin Osborne. *A course in game theory*, volume 29. 1995.

[34] Lc Evans. *An introduction to mathematical optimal control theory*. 2005.

[35] Chelsea C. White. A survey of solution techniques for the partially observed Markov decision process. *Annals of Operations Research*, 32:215–230, 1991.

[36] R. B. Myerson. Optimal Auction Design. *Mathematics of Operations Research*, 6(1):58–73, 1981.

[37] Michael Ostrovsky and Michael Schwarz. Reserve prices in internet advertising auctions: A field experiment. *Search*, pages 1–18, 2009.

[38] Baichun Xiao, W Yang, and J Li. Optimal reserve price for the generalized second-price auction in sponsored search advertising. *Journal of Electronic Commerce Research*, 10(3):114–129, 2009.

[39] Subodha Kumar, Varghese S. Jacob, and Chelliah Sriskandarajah. Scheduling advertisements on a web page to maximize revenue. *European Journal of Operational Research*, 173(3):1067–1089, 2006.

[40] Kristin Fridgeirsdottir and S Asadolahi. Revenue management for online advertising: Impatient advertisers. Technical Report 2007, Working paper, London Business School, 2007.

[41] Jon Feldman, Nitish Korula, Vahab Mirrokni, S. Muthukrishnan, and Martin Pál. Online Ad assignment with free disposal. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5929 LNCS:374–385, 2009.

[42] Jon Feldman, Monika Henzinger, Nitish Korula, Vahab S. Mirrokni, and Cliff Stein. Online stochastic packing applied to display ad allocation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6346 LNCS, pages 182–194. Springer, 2010.

[43] K Fjell. Online advertising: Pay-per-view versus pay-per-click A comment. *Journal of Revenue & Pricing Management*, 2(3):200–206, 2009.

[44] A Mangani. Online advertising: Pay-per-view versus pay-per-click. *Journal of Revenue & Pricing Management*, 2(4):295–302, 2004.

[45] Changhyun Kwon. Single-period balancing of pay-per-click and pay-per-view online display advertisements. *Journal of Revenue and Pricing Management*, 10(3):261–270, 2011.

[46] Wei Yang, Jun Qiao, Youyi Feng, and Baichun Xiao. Optimal reserve price in static and dynamic sponsored search auctions. *Journal of Systems Science and Systems Engineering*, 22(4):440–456, December 2013.

[47] Shuai Yuan, Bowei Chen, Jun Wang, Peter Mason, and Sam Seljan. An Empirical Study of Reserve Price Optimisation in Real-Time Bidding. In *Proceeding of the 20th ACM SIGKDD conference*, 2014.

[48] Haiying Ou. Maximization of online display advertising slots. *Proceedings of the 2012 5th International Conference on Business Intelligence and Financial Engineering, BIFE 2012*, pages 65–68, August 2012.

[49] Florin Constantin and J Feldman. An online mechanism for ad slot reservations with cancellations. In *Proceedings of the . . .* , pages 1265–1274, 2009.

[50] B. Chen, J. Wang, I. J. Cox, and M. S. Kankanhalli. Multi-Keyword Multi-Click Option Contracts for Sponsored Search Advertising. *arXiv preprint arXiv:1307.4980*, 2013.

[51] Hemant K. Bhargava and Juan Feng. Pay for Play: Sponsored Recommendations in Information Gatekeepers. *SSRN Electronic Journal*, February 2006.

[52] Chang-Hoan Cho, University of Texas At Austin) Is a As, and Hongsik John Cheon. Why Do People Avoid Advertising on the Internet? *Journal of Advertising*, 33(4):89–97, December 2004.

[53] a. Goldfarb and C. Tucker. Rejoinder–Implications of "Online Display Advertising: Targeting and Obtrusiveness". *Marketing Science*, 30:413–415, 2011.

[54] Cost-per-click-through Cost-per action and Yu Jeffrey Hu. Pricing of Online Advertising :. In *Sciences-New York*, pages 1–9, 2010.

[55] J Turner, A Scheller-Wolf, and S Tayur. Scheduling of dynamic in-game advertising. *Operations research*, 59(1):1–16, 2011.

[56] Chingning Wang, Ping Zhang, Risook Choi, and Michael D Eredita. Understanding Consumers Attitude toward Advertising. In *Eighth Americas Conference on Information Systems*, pages 1143–1148, 2002.

[57] Bowei Chen, Shuai Yuan, and Jun Wang. A Dynamic Pricing Model for Unifying Programmatic Guarantee and Real-Time Bidding in Display Advertising. In *Proceeding of the 8th International Workshop on Data Mining for Online Advertising (ADKDD 2014)*, page 9, 2014.

[58] Arpita Ghosh and Bip Rubinstein. Adaptive bidding for display advertising. In *Proceedings of the 18th international conference on World wide web*, pages 251–260, 2009.

[59] Santiago Balseiro, Omar Besbes, and Gabriel Y. Weintraub. Auctions for Online Display Advertising Exchanges: Approximations and Design. *SSRN Electronic Journal*, March 2012.

[60] Moshe Babaioff, Yogeshwer Sharma, and Aleksandrs Slivkins. Characterizing Truthful Multi-Armed Bandit Mechanisms. In *In ACM-EC*, pages 79–88, 2008.

[61] Kalyan Talluri and Garrett van Ryzin. Revenue Management Under a General Discrete Choice Model of Consumer Behavior. *Management Science*, 50(1):15–33, 2004.

[62] Yan Chen, Hadi Amiri, Zhoujun Li, and Tat-Seng Chua. Emerging topic detection for organizations from microblogs. *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '13*, page 43, 2013.

[63] Yongma Moon and Changhyun Kwon. Online advertisement service pricing and an option contract. *Electronic Commerce Research and Applications*, 10(1):38–48, 2011.

[64] Anselm Levskaya, Orion D Weiner, Wendell a Lim, and Christopher a Voigt. Spatiotemporal control of cell signalling using a light-switchable protein interaction. In *Nature*, volume 461, pages 997–1001. ACM, 2009.

[65] Kartik Hosanagar, Vadim Cherepanov, and Acm. *Optimal Bidding in Stochastic Budget Constrained Slot Auctions*. 2008.

[66] Yunhong Zhou, Deeparnab Chakrabarty, and Rajan Lukose. Budget constrained bidding in keyword auctions and online knapsack problems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5385 LNCS:566–576, 2008.

[67] Claudia Perlich, Brian Dalessandro, Rod Hook, Ori Stitelman, Troy Raeder, and Foster Provost. Bid optimizing and inventory scoring in targeted online advertising. In *Proceedings of the ACM SIGKDD 2012*, page 804, 2012.

[68] Vibhanshu Abhishek and Kartik Hosanagar. Optimal Bidding in Multi-Item Multislot Sponsored Search Auctions. *Operations Research*, 61:855–873, 2013.

[69] Animesh Animesh, Vandana Ramachandran, and Siva Viswanathan. Online Advertisers Bidding Strategies for Search, Experience, and Credence Goods: An

Empirical Investigation. In *Second Workshop on Sponsored Search Auctions. ACM Electronic Commerce*, 2005.

[70] Aranyak Mehta, Amin Saberi, Umesh V Vazirani, and Vijay V Vazirani. AdWords and generalized online matching. In *Journal of the ACM*, pages 264–273, 2007.

[71] Kursad Asdemir. Dynamics of bidding in search engine auctions: An analytical investigation. In *Proceedings of the 2nd Workshop on Sponsored Search Auctions*, 2006.

[72] Moira Regelson and D Fain. Predicting click-through rate using keyword clusters. In *Proceedings of the Second Workshop on Sponsored Search Auctions*, pages 1–6, 2006.

[73] Azin Ashkan, Charles L a Clarke, Eugene Agichtein, and Qi Guo. Estimating ad clickthrough rate through query intent analysis. *Proceedings - 2009 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2009*, 1:222–229, 2009.

[74] Ilya Gluhovsky. Forecasting Click-Through Rates Based on Sponsored Search Advertiser Bids and Intermediate Variable Regression. *ACM Transactions on Internet Technology*, 10(3):1–28, 2010.

[75] H Brendan McMahan, Gary Holt, D Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, Sharat Chikkerur, Dan Liu, Martin Wattenberg, Arnar Mar Hrafnkelsson, Tom Boulos, and Jeremy Kubica. Ad click prediction: a view from the trenches. *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1222–1230, 2013.

[76] Matthew Richardson, Ewa Dominowska, and Robert Ragno. Predicting clicks. *Proceedings of the 16th international conference on World Wide Web - WWW '07*, page 521, 2007.

[77] Krzysztof Dembczynski, Wojciech Kotlowski, and Dawid Weiss. Predicting

Ads' Click-Through Rate with Decision Rules. *Workshop on Targeting and Ranking in Online Advertising, WWW '08*, 2008.

[78] Haibin Cheng and Erick Cantú-Paz. Personalized click prediction in sponsored search. *Web Search and Web Data Mining*, pages 351–360, 2010.

[79] Weinan Zhang, Shuai Yuan, and Jun Wang. Optimal Real-Time Bidding in Display Advertising. In *Proceeding of the 20th ACM SIGKDD conference*, 2014.

[80] Yury Lifshits and Dirk Nowotka. Estimation of the click volume by large scale regression analysis. *Computer ScienceTheory and Applications*, 2007.

[81] Ying Cui, Ruofei Zhang, Wei Li, and Jianchang Mao. Bid landscape forecasting in online ad exchange marketplace. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11*, page 265, New York, New York, USA, August 2011. ACM Press.

[82] Brendan Kitts and Benjamin Leblanc. Optimal Bidding on Keyword Auctions. *Electronic Markets*, 14(3):186–201, September 2004.

[83] B. Kitts and B.J. LeBlanc. A trading agent and simulator for keyword auctions. *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, 2004. AAMAS 2004.*, pages 228–235, 2004.

[84] Christian Borgs, Jennifer Chayes, Nicole Immorlica, Mohammad Mahdian, and Amin Saberi. Multi-unit auctions with budget-constrained bidders. In *Proceedings of the 6th ACM conference on Electronic commerce - EC '05*, pages 44–51, New York, New York, USA, June 2005. ACM Press.

[85] Zoë Abrams. Revenue maximization when bidders have budgets. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm - SODA '06*, pages 1074–1082, New York, New York, USA, January 2006. ACM Press.

[86] Filip Radlinski, Andrei Broder, Peter Ciccolo, Evgeniy Gabrilovich, Vanja Josifovski, and Lance Riedel. Optimizing relevance and revenue in ad search. In

*Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '08*, pages 403–410, 2008.

[87] Nikhil R. Devanur and Thomas P. Hayes. The Adwords Problem: Online Keyword Matching with Budgeted Bidders under Random Permutations. *Science*, pages 71–78, 2009.

[88] Yunzhang Zhu, Gang Wang, Junli Yang, Dakan Wang, Jun Yan, Jian Hu, and Zheng Chen. *Optimizing search engine revenue in sponsored search*. 2009.

[89] Christian Rohrer and John Boyd. The rise of intrusive online advertising and the response of user experience research at Yahoo! In *Extended abstracts of the 2004 conference on Human factors and computing systems - CHI '04*, page 1085, New York, New York, USA, April 2004. ACM Press.

[90] Joanna Jaworska and Marcin Sydow. Behavioural targeting in on-line advertising: An empirical study. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5175 LNCS, pages 62–76. 2008.

[91] Rajiv M Dewan, Marshall L Freimer, and J I E Zhang. Management and valuation of advertisement-supported web sites. *Journal of Management Information Systems*, 19(3):87–98, 2003.

[92] Vf Araman and K Fridgeirsdottir. Cost-per-impression pricing and campaign delivery for online display advertising. pages 1–49, 2011.

[93] a Ranganathan and R H Campbell. Advertising in a pervasive computing environment. In *Proceedings of the 2nd international workshop on Mobile commerce WMC 02*, page 10, 2002.

[94] Paul Barford, Igor Canadi, Darja Krushevskaja, Qiang Ma, and S. Muthukrishnan. Adscape. *Proceedings of the 23rd international conference on World wide web - WWW '14*, pages 597–608, 2014.

[95] Nikhil R Devanur, Zhiyi Huang, Nitish Korula, Vahab S Mirrokni, and Qiqi Yan. Whole-page Optimization and Submodular Welfare Maximization with Online Bidders. *Ec*, 1(212), 2013.

[96] Rajiv M Dewan, Marshall L Freimer, Abraham Seidmann, and Jie Zhang. Web portals: Evidence and analysis of media concentration. *Journal of Management Information Systems*, 21(2):181–199, 2004.

[97] Subodha Kumar, Ym Li, and S Sethi. Optimal Pricing and Advertising Policies for Web Services. *Proceedings of the 14th Annual Workshop On Information Technologies and Systems (WITS)*, 2004.

[98] Subodha Kumar and Suresh P. Sethi. Dynamic pricing and advertising for web content providers. *European Journal of Operational Research*, 197(3):924–944, 2009.

[99] Yung Ming Li and Jhih Hua Jhang-Li. Pricing display ads and contextual ads: Competition, acquisition, and investment. *Electronic Commerce Research and Applications*, 8(1):16–27, January 2009.

[100] David J Walker. an Economic Analysis of. (October), 1989.

[101] Hemant K. Bhargava and Juan Feng. Paid placement strategies for internet search engines. *Proceedings of the eleventh international conference on World Wide Web - WWW '02*, page 117, 2002.

[102] G Hotchkiss, Steve Alston, and G Edwards. Eye tracking study. *Research white paper, Enquiro*, (June), 2005.

[103] Bernardo a. Huberman and Fang Wu. the Economics of Attention: Maximizing User Value in Information-Rich Environments. In *Advances in Complex Systems*, volume 11, pages 487–496, 2008.

[104] Rebecca Tushnet. Attention Must Be Paid: Commercial Speech, User-Generated Ads, and the Challenge of Regulation. *Buffalo Law Review*, 58:721–793, 2010.

[105] Georg Buscher and St Dumais. The good, the bad, and the random: an eye-tracking study of ad quality in web search. *Proceedings of the 33rd international ACM SIGIR*, pages 1–8, 2010.

[106] Jamie Murphy, Noor Hazarina Hashim, and Peter O'Connor. Take me back: Validating the wayback machine. *Journal of Computer-Mediated Communication*, 13(1):60–75, October 2007.

[107] Peter Lowe. Ad blocking with ad server hostnames and IP addresses, 2014.

[108] Eyeo GmbH. Known Adblock Plus subscriptions, 2014.

[109] AT Internet. Apps traffic grows throughout 2013, website traffic continues to decrease. Technical Report January 2014, 2014.

[110] Mary Meeker. Internet Trends 2014 Code Conference. pages 1–164, 2014.

[111] Frederick William Lanchester. Mathematics in Warfare. *The World of Mathematics*, 4:2138–2157, 1956.

[112] ComScore. comScore Releases February 2014 U.S. Search Engine Rankings - comScore, Inc, 2014.

[113] B Y Maeve Duggan and Aaron Smith. Social Media Update 2013, 2013.

[114] Shuai Yuan and Jun Wang. Sequential selection of correlated ads by POMDPs. In *Proceedings of the 21st ACM international conference on Information and knowledge management - CIKM '12*, page 515, 2012.

[115] Weinan Zhang, Jun Wang, Bowei Chen, and Xiaoxue Zhao. To personalize or not: a risk management perspective. In *Proceedings of the 7th ACM conference on Recommender systems - RecSys '13*, pages 229–236, New York, New York, USA, October 2013. ACM Press.

[116] Shu Wu and Shengrui Wang. Rating-based collaborative filtering combined with additional regularization. *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information - SIGIR '11*, page 1195, 2011.

[117] Sarah K. Tyler, Sandeep Pandey, Evgeniy Gabrilovich, and Vanja Josifovski. Retrieval models for audience selection in display advertising. In *Proceedings of the 20th ACM international conference on Information and knowledge management - CIKM '11*, page 593, 2011.

[118] Jie Tang, Ryen W White, and Peter Bailey. Recommending interesting activity-related local entities. *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information*, pages 1161–1162, 2011.

[119] Google. Profiting from Non-Guaranteed Advertising: The Value of Dynamic Allocation and Auction Pricing for Online Publishers. Technical report, 2012.

[120] AppNexus. AppNexus. www.appnexus.com (last visited 24/8/2013), 2014.

[121] David Maxwell Chickering and David Heckerman. Targeted Advertising on the Web with Inventory Management. *Interfaces*, 33:71–77, 2003.

[122] Sandeep Pandey, Deepayan Chakrabarti, and Deepak Agarwal. Multi-armed bandit problems with dependent arms. In *Proceedings of the 24th international conference on Machine learning - ICML '07*, pages 721–728, 2007.

[123] A R Cassandra. A survey of {POMDP} applications. In *Working Notes of AAAI 1998 Fall Symposium on Planning with Partially Observable Markov Decision Processes*, 1998.

[124] Chinese Physical Society. Monte Carlo Ł *. *Advances in neural information processing systems*, 62:1–8, 2013.

[125] Matthijs T J Spaan and Nikos Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24(1):195–220, 2005.

[126] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.

[127] Peter Turney. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2000.

[128] Anette Hulth. Improved Automatic Keyword Extraction Given More Linguistic Knowledge. In *EMNLP'03: Proceedings of the 2003 conference on Empirical Methods in Natural Language Processing*, pages 216–223, 2003.

[129] Hao Wu, Guang Qiu, Xiaofei He, Yuan Shi, Mingcheng Qu, Jing Shen, Jiajun Bu, and Chun Chen. Advertising keyword generation using active learning. In *Proceedings of the 18th international conference on World wide web*, pages 1095–1096, 2009.

[130] Sujith Ravi, Andrei Broder, Evgeniy Gabrilovich, Vanja Josifovski, Sandeep Pandey, and Bo Pang. Automatic generation of bid phrases for online advertising. In *Proceedings of the third ACM international conference on Web search and data mining - WSDM '10*, page 341, 2010.

[131] Ariel Fuxman, P Tsaparas, and K Achan. Using the wisdom of the crowds for keyword generation. In *Proceeding of the 17th*, pages 61–70, 2008.

[132] Yifan Chen, Gui-Rong Xue, and Yong Yu. Advertising keyword suggestion based on concept hierarchy. In *Proceedings of the international conference on Web search and web data mining - WSDM '08*, page 251, 2008.

[133] Amruta Joshi Amruta Joshi and Rajeev Motwani Rajeev Motwani. Keyword Generation for Search Engine Advertising. In *Sixth IEEE International Conference on Data Mining - Workshops (ICDMW'06)*, 2006.

[134] Vibhanshu Abhishek. Keyword Generation for Search Engine Advertising using Semantic Similarity between Terms. In *Proceedings of the ninth international conference on Electronic commerce (2007)*, pages 89–94, 2007.

[135] Paat Rusmevichientong and David P Williamson. An adaptive algorithm for selecting profitable keywords for search-based advertising services. In *7th ACM Conference on Electronic Commerce*, pages 260–269, 2006.

[136] Wei Li, Xuerui Wang, Ruofei Zhang, Ying Cui, Jianchang Mao, and Rong Jin. Exploitation and exploration in a performance based contextual advertising system. *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 27–35, 2010.

[137] John S Breese, David Heckerman, and Carl Myers Kadie. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In *UAI '98: 14th conference on Uncertainty in Artificial Intelligence*, UAI'98, pages 43–52, 1998.

[138] Jl Herlocker and Ja Konstan. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, page 8, 1999.

[139] Dimitri P Bertsekas. Dynamic Programming and Optimal Control, 3rd Edition , Volume II. *Control*, II:1–200, 2010.

[140] B.F. Torrence and E.a. Torrence. *The student's introduction to Mathematica: a handbook for precalculus, calculus, and linear algebra*. 2009.

[141] Peter Auer. Models for Trading Exploration and Exploitation using Upper Confidence Bounds. 3:397–422, 2005.

[142] Google. Google AdWords Keyword Tool. http://goo.gl/TYdOQ, 2009.

[143] Google. AdSense revenue share. http://goo.gl/yP4Ln, 2013.

[144] F Wilcoxon. Individual comparisons of grouped data by ranking methods. *Journal of economic entomology*, 39:269, 1946.

[145] Varsha Dani, T P Hayes, and Sham M Kakade. Stochastic Linear Optimization under Bandit Feedback. In *Computer*, pages 0–20, 2008.

[146] Paat Rusmevichientong and John N. Tsitsiklis. Linearly Parameterized Bandits. *Mathematics of Operations Research*, (1985):40, December 2008.

[147] Y Abbasi-Yadkori and a Antos. Forced-exploration based algorithms for playing in stochastic linear bandits. In *COLT Workshop on*, pages 1–6, 2009.

[148] Peter Auer. Using Confidence Bounds for Exploitation-Exploration Trade-offs. *Journal of Machine Learning Research*, 3:397–422, 2003.

[149] Kazuoki Azuma. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal*, 19:357–367, 1967.

[150] H. Zaragoza, N. Craswell, M. Taylor, S. Saria, and S. Robertson. Microsoft Cambridge at TREC-13: Web and HARD tracks. In *Proceedings of TREC 2004*. Citeseer, 2004.

[151] Eibe Frank, Gordon W Paynter, Ian H Witten, Carl Gutwin, and Craig G Nevill-manning. Domain-Specific Keyphrase Extraction. In *Sixteenth International Joint Conference on Artificial Intelligence*, pages 668—-673. Morgan Kaufmann Publishers Inc., 1999.

[152] Google. Search Engine Optimization Starter Guide Welcome to Google ' s Search Engine Optimization. Technical report, 2010.

[153] S Muthukrishnan. AdX: a model for ad exchanges. *ACM SIGecom Exchanges*, 8(2):1–12, 2009.

[154] Benjamin Edelman and Michael Schwarz. Optimal auction design in a multi-unit environment: The case of sponsored search auctions. *Unpublished manuscript, Harvard Business School*, 2006.

[155] Eyal Even-Dar, Jon Feldman, Yishay Mansour, and S. Muthukrishnan. Position auctions with bidder-specific minimum prices. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5385 LNCS:577–584, 2008.

[156] Tanmoy Chakraborty. Selective call out and Real-time Bidding. *Proceedings of 6th Ad Auction Workshop*, pages 1–22, 2010.

[157] Arpita Ghosh, Preston McAfee, Kishore Papineni, and Sergei Vassilvitskii. Bidding for representative allocations for display advertising. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5929 LNCS:208–219, 2009.

[158] R. Cavallo, R. P. McAfee, and S. Vassilvitskii. Display advertising auctions with arbitrage. *Workshop on Ad Auctions, EC*, pages 1–21, 2012.

[159] Jon Feldman. Auctions with Intermediaries [ Extended Abstract ] Categories and Subject Descriptors. In *New York*, 2010.

[160] Vijay Krishna. *Auction Theory*. Academic press, 2002.

[161] John G Riley, William F Samuelson, and William F. Riley, John G. and Samuelson. Optimal Auctions. *The American Economic Review*, 71:pp. 381–392, 1981.

[162] R Preston McAfee and Daniel Vincent. Sequentially Optimal Auctions. *Games and Economic Behavior*, 18(2):246–276, 1994.

[163] Ravi Bapna, Paulo Goes, and Alok Gupta. Insights and analyses of online auctions. *Communications of the ACM*, 44(11):42–50, 2001.

[164] H Xu, C.K. Bates, and S.M. Shatz. Real-Time Model Checking for Shill Detection in Live Online Auctions. In *Proc. of the International Conference on Software Engineering Research and Practice (SERP'09)*, volume pp, pages 13–16, 2009.

[165] Rj Kauffman and Ca Wood. Running up the Bid: Detecting, Predicting, and Preventing Reserve Price Shilling in Online Auctions. In *Proceedings of the 5th international conference on Electronic commerce*, pages 259–265, 2003.

[166] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American Economic Review*, 97:242–259, March 2007.

[167] Jana Jurečková, Jan Picek, and Pranab Kumar Sen. Goodness-of-fit test with nuisance regression and scale. *Metrika*, 58:235–258, 2003.

[168] Priscilla E Greenwood and Michael S Nikulin. *A guide to chi-squared testing*, volume 280. Wiley-Interscience, 1996.

[169] Omid Madani, Dj Lizotte, and R Greiner. The budgeted multi-armed bandit problem. In *Learning Theory*, pages 1–2. Springer, 2004.

[170] Kun Deng, Yaling Zheng, Chris Bourke, Stephen Scott, and Julie Masciale. New algorithms for budgeted learning. In *Machine Learning*, volume 90, pages 59–90, 2013.

[171] Long Tran-Thanh, Archie Chapman, Alex Rogers, and Nicholas R. Jennings. Knapsack based optimal policies for budget-limited multi-armed bandits. 2012.

[172] Steven Matthews. *A Technical Primer on Auction Theory I: Independent Private Values*. Number 1096. 1995.

[173] R. Engelbrecht-Wiggans. On Optimal Reservation Prices in Auctions. *Management Science*, 33(438):763–770, 1987.

[174] J F Bonnans, J C Gilbert, C Lemaréchal, and C a Sagastizábal. *Numerical Optimization. Theoretical and Practical Aspects*. Springer, 2003.

[175] C Shaw. Wilcoxon Signed Rank Test. *Science*, page 2000, 2000.

[176] Shuai Yuan, Jun Wang, and Maurice van der Meer. Adaptive Keywords Extraction with Contextual Bandits for Advertising on Parked Domains. July 2013.

[177] Shuai Yuan, Jun Wang, and Xiaoxue Zhao. Real-time bidding for online advertising: measurement and analysis. In *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising*, 2013.

[178] S Muthukrishnan. AdX: a model for ad exchanges. *ACM SIGecom Exchanges*, 8(2):2–3, 2009.

[179] Dominique Xardel. *The Direct Selling Revolution. Understanding the Growth of the Amway Corporation*. Blackwell Publishing, 1993.

[180] DC Fain and JO Pedersen. Sponsored search: A brief history. 2006.

[181] Hairen Liao, Lingxiao Peng, Zhenchuan Liu, and Xuehua Shen. iPinYou Global RTB Bidding Algorithm Competition Dataset. In *Proceeding of the 8th International Workshop on Data Mining for Online Advertising (ADKDD 2014)*, 2014.

[182] I N Partnership With, With The, and Support Of. The Data Management Platform : Foundation for Right-Time Customer Engagement. (November 2012), 2012.

[183] E McCallister, T Grance, and K Kent. *Guide to protecting the confidentiality of personally identifiable information (PII)*. DIANE Publishing, 2010.

[184] John Downes and Jordan Elliot Goodman. *Dictionary of finance and investment terms*. Barron's New York, 1995.

[185] David Lucking-Reiley. Vickrey Auctions in Practice: From Nineteenth-Century Philately to Twenty-First-Century E-Commerce. *Journal of Economic Perspectives*, 14:183–192, 2000.

[186] ComScore. comScore MMX Ranks Top 50 U.S. Web Properties for May 2008, 2008.

[187] Adomik. adomik case study on Orange Ad Market, 2014.

# Appendix A

# Terminology and Explanation

This appendix summarises the terms and concepts used frequently in online advertising industry.

Ad Exchange: A marketplace where demand and supply from various sources are aggregated and traded [178], like a multi-level marketing structure [179] (i.e., they aggregate demand and supply of ad networks, and also connect with advertisers and publishers directly).

Ad Network: A marketplace for advertisers to buy impressions and for publishers to sell them. Usually ad networks connects with advertisers and publishers directly. Since they are usually limited by certain rules (geographical location, content vertical, etc.) on the commercial side, the demand and supply in a single ad network are usually unbalanced.

Bid Phrase: A keyword or key-phrase that advertisers bid on to represent their anticipation of the user intention, mostly used in sponsored search [8].

Click-Through Rate (CTR): A click-through occurs when a user clicks on an ad. The CTR is the number of clicks divided by the number of impressions [180].

Conversion Rate (CVR): A conversion occurs when a user completes some action (e.g., downloading, registering, or purchasing) after reading an ad. The conversion rate is usually calculated as the number of conversions divided by the number of impressions [180]. There are also cases where it is defined as the number of conversions divided by the number of clicks [181].

Cookie: A small file placed on the users computer by a website that is usually for saving personal information and could be used by advertisers for targeting [90].

Cost per Action (CPA): The price that an advertiser pays if a user converts (e.g., down-

loading, registering, or purchasing).

Cost per Click (CPC): The price that an advertiser pays if the ad is clicked.

Cost per Mille-impressions (CPM): The price that the advertiser would pay if his ad is displayed one thousand times. Since a single impression is usually very cheap (about \$0.001 to \$0.01), one thousand is used for the ease of presentation.

Creative: A brief description, in the form of text, picture, animation, or video, about a service or product that the advertiser wishes to promote. It is the actual content of an ad.

Data Management Platform (DMP): A system that collects, builds, and sells users' interest segment information. With the help of such information better matching could be achieved especially in Real-Time Bidding [182].

Demand Side Platform (DSP): An automated bidding platform for advertisers to buy impressions across ad exchanges.

Impression: An opportunity of displaying an ad to a user reading the webpage.

Inventory: Virtual assets owned by a publisher. It usually refers to a group of impressions from certain placements.

Landing Page: A webpage associated with an ad. The user will be redirected to this webpage after clicking the ad.

Organic Search Result: Search results that are returned by normal search algorithms with no advertising involved.

Personally Identifiable Information (PII): Information about a user that can be determined from their cookies, browser information, and other sources [183].

Placement (Ad slot): A section on a webpage (or multiple webpages, or even across websites) that displays ads.

Return-On-Investment (ROI): Usually expressed as a percentage, ROI is the ratio of money gained or lost (whether realized or unrealized) on an investment relative to the amount of money invested [184].

Real-Time Bidding (RTB): An ad trading mechanism in which an advertiser will be asked to submit a response for every bid request. RTB was introduced by ad exchanges where it is impossible to know all advertisers or publishers in advance.

Second Price Auction (Vickery Auction): The winner pays the next highest bid instead of his own bid [185].

Segment (Tag): A category that a user belongs to. In online advertising segments could be very flexible, for example by income range, ages of children, hobbies, psychometry, and short-term purchase intent. Usually segments are not explicitly stated but inferred by users' Internet usage.

Supply Side Platform (SSP): An automated platform for publishers to manage inventories and sell impressions in many ad exchanges. An essential advantage of using a SSP is to reach more potential buyers. Besides, such a platform is expected to optimise the revenue by applying various yield management tools.

User: A person who browsers the web and reads ads.

# Appendix B

# Implementing a Reserve Price Optimisation System

The chapter describes in detail the implementation of reserve price optimisation prototype system. The source code could be obtained from `https://github.com/shuaiyuancn/rtb-reserve-price-opt`. The system has been designed and implemented mainly in the AppNexus' advertising ecosystem. Thus, it follows certain standard and data formats, and requires account and privileges before use. However, the system can be easily modified to work with other Supply Side Platforms (SSP) as long as the general input (auction data feed or performance report) and output (API to manipulate the reserve prices) functions are available.

In a production environment, the cost of implementing such a system depends on several factors:

**The number of optimisation tasks.** Apparently, each task takes certain amount of time to complete (loading configuration, pulling data, running algorithms, applying result, logging, etc.). The number of tasks has a linear relationship to the cost or system's capacity.

**The granularity of optimisation.** If an ad placement has huge number of impressions, it is preferable to cluster impressions according to targeting rules that matter the most (e.g., hours of day, geographical locations, ad sizes, user segments). Every dimension adds extra sophistication to the optimisation process. The number of clustering dimensions has a polynomial relationship to the cost or the system's capacity.

**The optimisation algorithms.** In the prototype system there is a *brute force* algorithm that does the exhaustive search of the optimal reserve price value over the past

data feed. Although the algorithm itself has been optimised when implemented (by using `brute` function in `scipy.optimize` package[1] with reasonable number of iterations), it could take a long time to finish if the range of exploration is wide, e.g., from \$0.0 to \$10.0 with a step of \$0.01. Similarly, other algorithms with high computational complexity could easily add running time of each optimisation iteration, thus adding the cost.

In the experiments of Chapter 5, the prototype system has been set up in Amazon EC2[2] using an `m3.large` instance. Such an instance has has 2 virtual CPUs (each of that is equivalent to an Intel Xeon E5-2670 v2 (Ivy Bridge) processor), 7.5 Gigabyte memory, and 32 Gigabyte solid state drive (SSD) instance storage. To save the large scale of auction data feeds, two 1.0 Terabyte Amazon Elastic Block Stores (EBS)[3] have been used to form a RAID0 structure[4].

Such an instance has been created in the Europe region in Amazon EC2 with a 3-year reserved instance contract. The instance serves the hourly optimisation of more than 30 tasks. The effective monthly cost is less than \$240.

## B.1 Market Research

Before presenting the prototype system, this section surveys the current online advertising ecosystem for similar products or services, although details of such products or services are usually undisclosed to non-clients. The intention is to review the feasibility (most SSPs provide the reserve price setting) and relatively low competition (few companies offer the dynamic reserve price optimisation).

More commonly, SSPs provide the capability of setting up a reserve price (sometimes also a soft floor price) in their platform. Such function could usually be access via API provided by the vendor. For example,

- Doubleclick For Publisher (DFP)
- AppNexus Console
- OpenX SSP
- MoPub

[1] `http://goo.gl/f8VZ6A` (Last accessed: 4/8/2014)
[2] `http://aws.amazon.com/ec2` (Last accessed: 4/8/2014)
[3] `http://aws.amazon.com/ebs` (Last accessed: 4/8/2014)
[4] `http://goo.gl/I8ZJ4Q` (Last accessed: 4/8/2014)

The existence of rich APIs and few products or services suggest good opportunity of dynamic reserve price optimisation in online advertising marketplace. However, few of them provide such a function in their systems.

The only exception is Rubicon Project (NYSE: RUBI), founded in 2007, an online advertising technology firm based in Los Angeles, California. In February 2014, the company filed for an IPO and went public in April of 2014 opening at over $20 per share. The stock fell back to about $16 per share later in the month. In March of 2014, the company was named number two on the top Ad Exchange Entities by comScore [186].

The company optimises advertising revenue for web sites with the product *Seller Cloud*, by using thousands of proprietary and proven data-driven, machine-learning algorithms that replace guesswork. For the reserve price optimisation, the company offers to clients *Dynamic Price Floors* which is essentially a brute force approach. The algorithm is described as[5]:

> *One form of that analysis is auction simulation. Using a Hadoop cluster and proprietary algorithms the REVV platform simulates the auction at different price floors, usually at penny increments between five cents and 20 dollars. During the simulation the system records the revenue gained for each impression whether it goes to the highest bidder at the second price, at the floor price or even if the floor prices out all bids and the impression goes to a non-RTB demand source. After the simulation another algorithm runs through the revenue at each floor and determines at which floor price the revenue is maximized. These simulations are run at different levels of audience granularity and the resulting, boiled data is stored.*

Note that *REVV* is the former name of the *Seller Cloud*.

As argued in Chapter 5, buyers' attrition analysis could be crucial because any reserve price takes the risk of driving advertisers away. This is especially important with quick advancement of bidding algorithms in recent years, since more and more of them are taking into account the cost when making bidding decisions. In *REVV* or *Seller Cloud* such analysis module cannot be identified.

[5] `http://goo.gl/tMrb92` (Last accessed: 7/8/2014)

There are other companies providing reserve price optimisation products or services, For example,

- MediaFuse, details are undisclosed;
- PubSquared, details are undisclosed;
- AlephD, details are undisclosed;
- adomik, the dynamic floor prices are set at placement or publisher level. A case study on Orange Ad Market [187] showed 22% monthly revenue increase on average;

Comparing with many large media groups relying more and more heavily on online advertising revenue, the relatively few optimisation providers suggest a good opportunity in the market.

## B.2 Prerequisites

This section lists the prerequisites. The prototype system is implemented in Python 2.7.8 that could be obtained from `https://www.python.org`. However, it is strongly recommended using **Anaconda** distribution which could be obtained from `https://store.continuum.io/cshop/anaconda`. The major advantage of it, especially on a Windows/PC platform, is to save a lot of effort setting up the scientific computing environment, because the distribution comes with many useful libraries.

If one prefers to set up the environment with greater control, note that the following Python libraries are used by the prototype:

**Numpy**, obtained from `http://numpy.scipy.org` with version 1.8.1. NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more. In the prototype system, Numpy is widely used for random choices, series generation, applying computation to an array, etc.

**pandas**, obtained from `http://pandas.pydata.org` with version 0.14.1. pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with *relational* or *labelled* data both easy and intuitive. It aims

to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis/manipulation tool available in any language. In the prototype system, pandas is widely used to manage data efficiently, e.g., indexing and re-indexing, filtering, transforming, and some quick plotting.

**SciPy**, obtained from `http://www.scipy.org` with version 0.14.0. SciPy is a collection of mathematical algorithms and convenience functions built on the Numpy extension of Python. It adds significant power to the interactive Python session by providing the user with high-level commands and classes for manipulating and visualizing data. With SciPy an interactive Python session becomes a data-processing and system-prototyping environment. In the prototype system, `optimize` package in SciPy is used to solve convex function optimisation and exhaustive search problems.

**scikit-learn**, obtained from `http://scikit-learn.org` with version 0.15.1. scikit-learn is a simple and efficient tools for data mining and data analysis. It is made accessible to everybody, and reusable in various contexts. It has been built on NumPy, SciPy, and matplotlib. In the prototype system, generalised linear model packages from scikit-learn are used to solve regression tasks.

**Requests**, obtained from `http://www.python-requests.org` with version 2.3.0. Requests takes all of the work out of Python HTTP/1.1 making your integration with web services seamless. Theres no need to manually add query strings to your URLs, or to form-encode your POST data. Keep-alive and HTTP connection pooling are 100% automatic, powered by urllib3, which is embedded within Requests. In the prototype system, Requests is used to communicate with remote API services to manipulate reserve prices.

**matplotlib**, obtained from `http://www.matplotlib.org` with version 1.3.1. matplotlib is a library for making 2D plots of arrays in Python. Although it has its origins in emulating the MATLAB[6] graphics commands, it is independent of MATLAB, and can be used in a Pythonic, object oriented way. Although matplotlib is written primarily in pure Python, it makes heavy use of NumPy and other extension code to provide good performance even for large arrays. In the prototype system, matplotlib is used to generate analytical charts to monitor performance and attrition.

[6]MATLAB is a registered trademark of The MathWorks, Inc.

**web.py**, obtained from `http://webpy.org` with version 0.3. web.py is a very clean and simple web framework/library written in Python to assist in the development of Python web applications. Where other frameworks may offer more features and complexity, web.py excels in keeping things simple and efficient. In the prototype system, web.py is used to power a simple web UI for optimisation task management. We also use its `db` package to access the MySQL database.

**MySQL-Python**, or MySQLdb, obtained from `http://mysql-python.sourceforge.net` with version 1.2.2. MySQLdb is an thread-compatible interface to the popular MySQL database server that provides the Python database API. In the prototype system it is used by web.py to access the MySQL database.

Additionally, the prototype system uses **crontab** in Linux systems to schedule job running. The back-end database system is **MySQL**, which could be obtained from `http://www.mysql.com`.

## B.3 Overview of the prototype

This section describes modules of the prototype system and their functionalities; then the flow of data and their structure (fields) to provide an overview of the system.

### B.3.1 Modules

The prototype system consists of the following modules:

**Task controller**, also the entry point of the system, is responsible for running loading configuration from database for each task, pulling data feeds (mainly auction data), running assigned algorithm, accessing remote API services to apply reserve prices, and logging major activities for future analysis.

**API services controller** provides necessary functions for accessing remote API services, including authentication, pulling data feeds, downloading reports, and applying reserve prices and targeting rules. The API services controller is created by following AppNexus' format and standard thus it requires obtaining account and privileges from AppNexus before use.

**Data feed controller** pulls impression level data feed from remote API services, parse, clean, and join data to construct auctions, and save the result to database for future access.

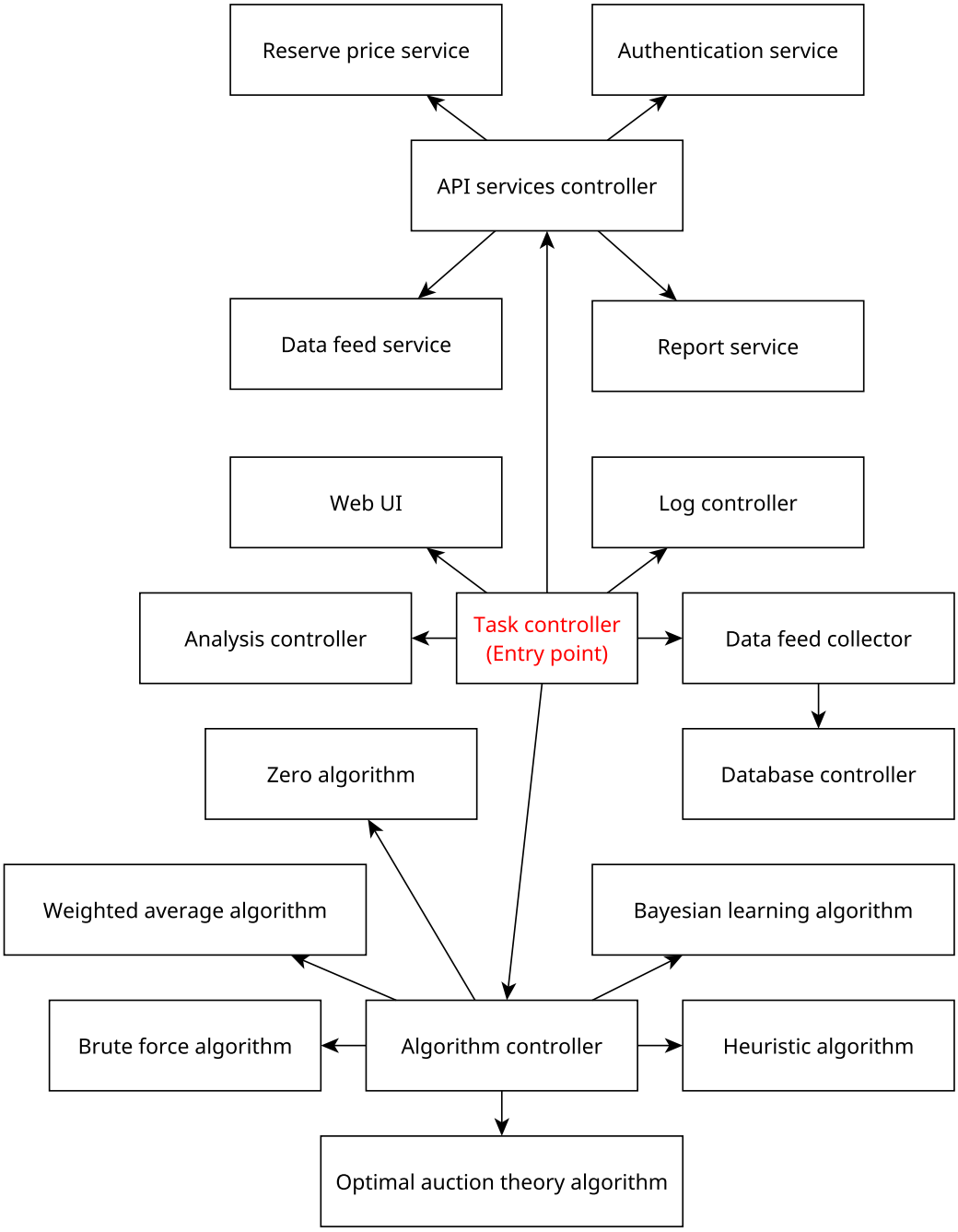


**Figure B.1:** The modules of the prototype system. The system starts by loading optimisation tasks. For each task, it loads data feed from remote API services and run assigned algorithms. The result reserve price is applied by accessing remote API services again. Additionally, the system has a simple web user interface (UI) for optimisation task management and an analysis module to monitor performance and possible attrition.

**Database controller** provides standard create, read, update, and delete (CRUD) functions of task configurations, data feeds, and logs.

**Algorithm controller** is responsible for training models for various optimisation

algorithms, selecting one for the current task(randomly or based on pre-assignment), running the algorithm with auction data and parameters, and reporting the result reserve prices. In this module we have implemented the brute force, heuristics, Bayesian learning, weighted average, and optimal auction theory algorithms. The mathematical detail of each algorithm has been presented in Chapter 5. The code is listed in Section B.4.

**Analysis controller** is used for performance and possible attrition analysis. It compares the predicted and actual uplift of revenue, revenue from the test and control group, and reports reserve prices of each task over time. It also monitors for each task the number of bidders, bid distribution and budget allocation of each bidder to detect possible attrition.

**Log controller** saves the result reserve price for each run of optimisation, and the remote reserve price setting after accessing API services. These logs are then used by analysis controller for further inspection and monitoring.

**Web user interface (UI)** provides an intuitive interface for task management, including creating a task, updating tasks' configurations, activating/de-activating a task, displaying raw logs, and displaying performance and attrition plots and reports.

### B.3.2 Data flow and structure

This section briefly describes the data flow of a single run of optimisation.

For each task, the task controller loads its configuration and invokes the data feed service API to get the auction data. Then the algorithm controller takes over, selects an optimisation algorithm by random or pre-assignment, runs the algorithm and collects the result, and invokes the reserve price service API to apply the result. Logs will be generated along with a predicted optimal reserve price and saved into database.

The task controller is also responsible for querying performance reports from the report service API (usually on a daily basis). These reports contain all placements belonging to the publisher including ones not in the prototype system. Those placements not having the reserve price optimisation are control groups in evaluation. The reports are saved into database. The analysis controller runs periodically to generate performance and attrition report for each task by taking logs generated from the optimisation process, as well as reports pulled from the SSP system.

The object models are summarised in Table B.1.

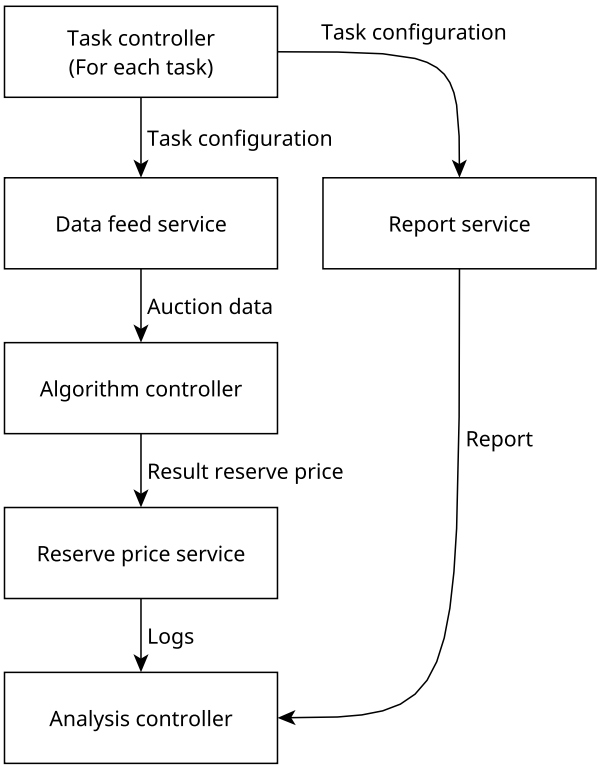


**Figure B.2:** The data flowchart of the prototype system. The data structure of each step is detailed in Table B.1.

**Table B.1:** The structures (fields) of major data objects in the prototype system.

| **Data object** | **Fields** |
|---|---|
| Task configuration | task_id, publisher_id, placement_id, reserve_price_range, algorithm, is_active, hour_targeting, geo_targeting, user_targeting |
| Auction data | auction_id, timestamp, placement_id, price_paid, geo, user_segment, a list of <bidder_id, bid_price> |
| Result reserve price | task_id, timestamp, reserve_price, hour_targeting, geo_targeting, user_targeting |
| Log | task_id, timestamp, reserve_price_rule_id, reserve_price, hour_targeting, geo_targeting, user_targeting |
| Report | hour, publisher_id, placement_id, total_imps, unsold_imps, clicks, conversions, revenue |

## B.4 Algorithm Implementation

This section discusses the implementation of each algorithm in detail by giving code samples and listing challenges and solutions.

### B.4.1 Algorithm Base

Algorithm base, or the `BaseAlgorithm`, is the parent class of all algorithms in the system. Such design follows the Object Oriented discipline and makes the sharing basic and common functions among algorithms easy.

A key function in `BaseAlgorithm` is `get_bids`, which extracts into `Numpy` arrays the first and second highest bids and the managed bids from a list of auctions. The managed bids are bids from the owner of the placement. If an auction fails due to high reserve price and a managed bid is present, the impression will be won by the owner so that it could be redirected to other ad channels for re-sell.

```
def get_bid(publisher_id, grouped_auction):
    bid1 = []
    bid2 = []
    managed_bid = []
    for auction_id, all_bid in grouped_auction:
        if len(lst_all_bid) > 1:
            bid1.append(float(all_bid.iloc[0]['bid_price']))
        else:
            bid1.append(0.)

        if len(lst_all_bid) > 2:
            bid2.append(float(all_bid.iloc[1]['bid_price']))
        else:
            bid2.append(0.)

        df = all_bid[all_bid['bidder_id'] == publisher_id]
        if len(df) != 0:
            managed_bid.append(float(df.iloc[0]['bid_price']))
        else:
            managed_bid.append(0.)

    return np.array(bid1), np.array(bid2), np.array(managed_bid)
```

### B.4.2 Brute Force Algorithm

The brute force algorithm tries to find the optimal reserve price by exhaustive search within the pre-defined value range. The full definition of this algorithm is:

```
class BruteForceAlgorithm(BaseAlgorithm):
  def __init__(self, minimum, maximum, Ns):
    BaseAlgorithm.__init__(self, minimum, maximum)

    self.Ns = Ns

  def revenue_func(reserve_price, bid1, bid2, managed_bid):
    return - np.sum(np.where(reserve_price > bid1,
                  # auction failed
                  np.where(managed_bid != 0.,
                        np.amax(managed_bid, self.minimum),
                        0.),
                  # auction succeeded
                  np.where(reserve_price > bid2,
                        reserve_price,
                        bid2)))

  def compute(bid1, bid2, managed_bid):
    bid1, bid2, managed_bid = get_bid(seller_id, grouped_auction)
    ret = optimize.brute(func=revenue_func,
                ranges=((self.minimum, self.maximum), ),
                args=(bid1, bid2, managed_bid, self.minimum, ),
                Ns=self.Ns, # number of iterations
                full_output=True,
                finish=None,
                disp=False)

    reserve_price = float(ret[0])

    # revenue curve for further analysis
    x = ret[2]
    y = ret[3]
```

```
34      return reserve_price, x, y
```

Note that `np` is an alias for the `Numpy` library.

The `revenue_func` function returns a negative value for the ease of using optimisation libraries, since most of them try to find the minimum instead of the maximum.

As shown in the function body, the `compute` function also returns the revenue curve for the current auction data. A revenue curve is a series of reserve_price with corresponding revenue (negative value in our case). Such curves are useful in performance analysis, e.g., sharp-shaped revenue curves require conservative prediction since failed auctions incur great loss to the publisher.

### B.4.3 Heuristic Algorithm

The heuristic algorithm tries to find the optimal reserve price by iterating over auction data once, and adjust the reserve price after each auction according to parameters' set up. The full definition of this algorithms is:

```
class HeuristicAlgorithm(BaseAlgorithm):
2   def __init__(self, minimum, maximum, scale_low, scale_match,
      scale_high, decay):
      BaseAlgorithm.__init__(self, minimum, maximum)
4
      self.scale_low = scale_low
6     self.scale_match = scale_match
      self.scale_high = scale_high
8     self.decay = decay

10  def compute(self, bid1, bid2, managed_bid):
      reserve_price = minimum
12
      for b1, b2, managed_b in zip(bid1, bid2, managed_bid):
14      reserve_price = step(reserve_price, b1, b2, count)
        count += 1
16
      return max(min(reserve_price, self.maximum), self.minimum)
18
    def step(self, reserve_price, b1, b2):
```

```
    if reserve_price < b2:
      # too low
      reserve_price *= (1 + self.scale_low * self.decay ** count)
    elif b2 <= reserve_price <= b1:
      # desirable
      reserve_price *= (1 + self.scale_match * self.decay ** count)
    elif reserve_price > b1:
      # too high
      reserve_price *= (1 - self.scale_high * self.decay ** count)

    return reserve_price
```

### B.4.4 Optimal Auction Theory Algorithm

The optimal auction theory algorithm tries to find the optimal reserve price by fitting bids from auction data into a Log-normal distribution, and then solving Equation 5.2. The full definition of the algorithm is:

```
class OptimalAuctionTheoryAlgorithm(BaseAlgorithm):
  def __init__(self, minimum, maximum):
    BaseAlgorithm.__init__(self, minimum, maximum)

    def value_func(self, x):
        pdf_val = self.log_norm_pdf(x)
        pdf_d_val = self.log_norm_pdf_d(x)

        # we want this to be 0
        return x - (1 - pdf_val) / pdf_d_val - self.minimum

    def log_norm_pdf(self, x):
        return (math.exp(-(self.log_mean - x) ** 2 / (2 * self.sd **
    2)) /
                (self.sd * x * math.sqrt(2 * math.pi)))

    def log_norm_pdf_d(self, x):
        return (-self.log_norm_pdf(x) / x -
                self.log_norm_pdf(x) *
                (math.log(x) - math.exp(self.log_mean)) /
```

```
                (x * self.sd ** 2))

    def compute(self, grouped_auction):
        bid = []
        for auction in grouped_auction:
            bid.extend([i["bid_price"] for i in auction])

        sd, loc, log_mean = stats.lognorm.fit(data=bid, floc=0)

        res = minimize_scalar(fun=self.value_func,
                bounds=(self.minimum, self.maximum),
                method="bounded")

        return math.log(res.x)
```

### B.4.5 Weighted Average Algorithm

The weighted average algorithm looks back at revenue reports and tries to find a reserve price by averaging historical payoffs. The historical payoffs are discounted by weights since the algorithm believes a more recent report is more important. Weights follow either a linear or an exponential scale. The full definition of the algorithm is:

```
class WeightedAverageAlgorithm(BaseAlgorithm):
  # weight type
  LINEAR = 1
  EXPONENTIAL = 2  # base e

    def __init__(self, minimum, maximum, window_length, weight_type =
    LINEAR):
        BaseAlgorithm.__init__(self, minimum, maximum)

        self.window_length = window_length
        self.weight_type = weight_type

    def compute(self, recent_payoff):
        # make sure we use the correct window size
        actual_size = min(len(recent_payoff), self.window_length)

```

```
    weight_base = 0.
17          cumulative = 0.

19          for idx in range(len(recent_payoff),
                             len(recent_payoff) - actual_size,
21                           - 1):
                if self.weight_type == WeightedAverageAlgorithm.LINEAR:
23                  weight_base += idx
                    cumulative += idx * recent_payoff[idx - 1]
25              else:
                    weight_base += math.exp(idx + actual_size - len(
    recent_payoff))
27                  cumulative += math.exp(idx + actual_size - len(
    recent_payoff)) * recent_payoff[idx - 1]

29          if weight_base == 0.:
                reserve_price = 0.
31          else:
                reserve_price = cumulative / weight_base
33
    return reserve_price
```

### B.4.6 Bayesian Learning Algorithm

The Bayesian learning algorithm has two variations. They try to find a reserve price by fitting bids from auction data to a univariant or a bivariant Log-normal distribution. For the univariant case, the reserve price is taken as the mean of the distribution; for the bivariant case, the reserve price is set to the value that maximise the probability of being within two means. The full definition of the algorithm is:

```
class BayesianLearningUnivariantAlgorithm(BaseAlgorithm):
2   def __init__(self, minimum, maximum, theta, delta, sigma):
        BaseAlgorithm.__init__(self, minimum, maximum)
4
        self.theta = theta
6       self.delta = delta
        self.sigma = sigma
8
```

```
    def compute(self, bid):
        self.theta = (self.theta * self.delta +
            bid * self.sigma) / (self.sigma + self.delta)
        self.delta = (self.delta * self.sigma) / \
           (self.sigma + self.delta)

        reserve_price = self.theta

    return max(min(reserve_price, self.maximum), self.minimum)

class BayesLearningBivariantAlgorithm(BaseAlgorithm):
    def __init__(self, minimum, maximum,
                 m1, v_log1, v_normal1,
                 m2, v_log2, v_normal2):

        BaseAlgorithm.__init__(self, minimum, maximum)

    self.m1 = m1
        self.v_normal1 = v_normal1
        self.v_log1 = v_log1

        self.m2 = m2
        self.v_normal2 = v_normal2
        self.v_log2 = v_log2


    def bayesian_inference(self, m, v_normal, v_log, x):
        new_v_normal = 1 / v_normal + 1 / v_log
        new_m = (m / v_normal + x / v_log) / new_v_normal

        return new_m, new_v_normal

    def lognormal_pdf(self, x, m, var):
        sd = math.sqrt(var)
        return (math.exp(-(m - math.log(x)) ** 2 /
        (2 * sd ** 2)) /
                (sd * x * math.sqrt(2 * math.pi)))

```

```
    def lognormal_cdf(self, x, m, var):
48      return 0.5 + 0.5 * erf((math.log(x) - m) /
                math.sqrt(2 * var))
50
    def object_function(self, x):
52      return -(self.lognormal_cdf(x, self.m1,
                   self.v_normal1) *
54              (1 -
        self.lognormal_cdf(x, self.m2,
56                  self.v_normal2)))

58  def compute(self, bid1, bid2):
        (self.m1,
60       self.v_normal1) = \
      self.bayesian_inference(self.m1,
62                  self.v_normal1,
                    self.v_log1,
64                  bid1)
        (self.m2,
66       self.v_normal2) = \
      self.bayesian_inference(self.m2,
68                  self.v_normal2,
                    self.v_log2,
70                  bid2)

72      op_result = \
      minimize_scalar(fun=self.object_function,
74            bounds=(self.minimum, self.maximum),
              method="bounded")
76
    reserve_price = math.log(op_result.x)
78
    return reserve_price
```

### B.4.7 Zero Algorithm

The zero algorithm always returns the minimum possible value within the pre-configured range. The full definition of the algorithms is:

```
class ZeroAlgorithm(BaseAlgorithm):
    def __init__(self, minimum, maximum):
        BaseAlgorithm.__init__(self, minimum, maximum)

    def compute(self):
        return self.minimum
```

## B.5 Quick User Manual

This section lists critical steps of setting up the environment and the prototype system. More detailed information about each step could be found above in this Appendix.

1. To create a Linux server instance with good capacity. For 30 optimisation tasks with a random mixture of all available algorithms, we recommend an Amazon EC2 `m3.large` instance or its equivalent;
2. To install `Python` 2 and required libraries. `Python` is usually supplied with the Linux OS. However, if you do not have it yet you can install it by running

   ```
   sudo apt-get install python2
   ```

   Note that `apt-get` or `yum` could be the package manager depending on your Linux OS distribution.

   To install the required libraries, you need to run

   ```
   pip install numpy scipy matplotlib pandas scikit-learn requests
       web.py MySQL-Python
   ```

   Depending on the Linux OS distribution, you may be able to install some of the libraries supplied with the OS, for example,

   ```
   sudo apt-get install python-numpy python-scipy
   ```

Alternatively, you can choose to install Anaconda distribution which contains `Python` and most of required libraries, for example

```
bash Anaconda-2.x.x-Linux-x86[_64].sh
```

3. To install MySQL DB, then create a database, create a user with password, and grant the privileges, for example in MySQL's admin console,

```
CREATE DATABASE `optimisation`;
CREATE USER 'jeffrey'@'localhost' IDENTIFIED BY 'mypass';
GRANT ALL ON `optimisation` TO 'jeffrey'@'localhost';
```

4. To clone the source code of the system from `https://github.com/shuaiyuancn/rtb-reserve-price-opt` into a convenient location, for example,

```
cd /workspace
git clone git@github.com:shuaiyuancn/rtb-reserve-price-opt.git
```

5. To edit the `mysql_wrapper.py` for the correct DB details as created above;
6. To edit the API credentials in `api_func.py`;
7. To start the Web UI by running

```
cd /workspace/rtb-reserve-price-opt
python www/server.py 8080
```

Then you can access the Web UI via `http://SERVER-IP:8080`

8. To create an optimisation task, for example,

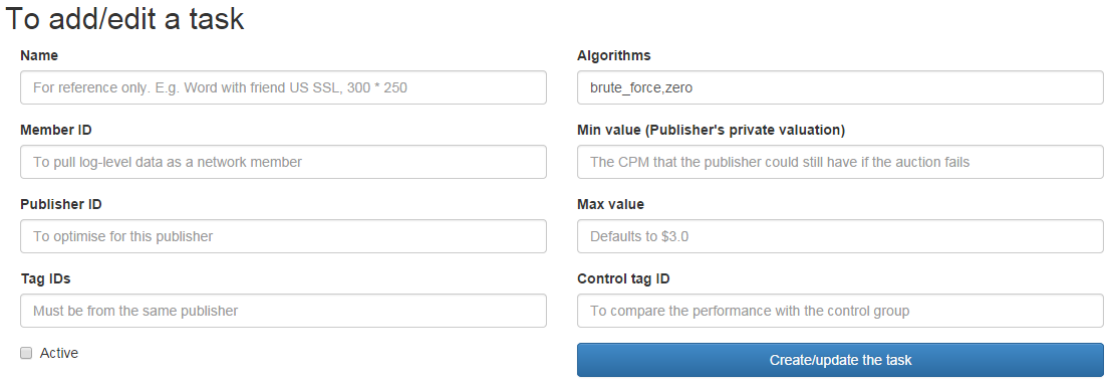


**Figure B.3:** The interface for adding an optimisation task in the Web UI.

9. Then the task will run automatically every hour.