Graphic Tool For Big Data - A Simulation System for Search Ads Auction

Jian Qiao
Ryan Casey
Tian Liu
Marc Capelo

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2016-64
http://www.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-64.html

May 12, 2016

Acknowledgement

I would like to express my thankfulness to Prof. John Canny and Phd advisor Biye Jiang for their guidance and help throughout the project.

University of California, Berkeley

Master of Engineering Final Report

Graphic Tools For Big Data

A Simulation System for Search Ads Auction

Jian Qiao

Supervised by

Professor John Canny

With team member

Ryan Casey, Marc Capelo, Tian Liu

May 9, 2016

Contents

Chapter 1

Technical Contributions

1.1 Introduction

With more consumers generating web traffics, many advertisers have chosen to move their budgets from traditional advertisement channels to the internet (Turk, 2015, p.4), and as a result, major search engines have been developing state-of-the-art algorithms to ensure the fairness of their advertisement platform and the maximization of revenue. However, such algorithms are often very difficult to tune due to the level of complexity, and with the huge amount of data involved, it often takes a very long time for ads operation personnel to see the effect after a tuning action is made. Our project, Graphic Tool for Big Data, addresses this problem by producing a visualization tool for machine learning algorithms in search advertisement bidding process. By simulating the bidding process with historical data and visualizing the metrics, our project enables the ads operation personnel to see real-time changes on performance metrics after a tuning attempt, therefore speeding up the model adjustment process.

As stated, our project stands at the intersection of machine learning algorithms, search advertisement auction process, and data visualization. To produce the final prototype, we developed a simulation pipeline for the prevalent search ads bidding process, set up a ma-

chine learning model used by that process to predict click-through rate, and built a web tool that visualizes various metrics from the process and enables interaction with the parameters. Following this work breakdown, we divided our project group into two subteams, in which Ryan and Marc worked on the web development part that includes the server and visualization, and Tian and I worked on the backend side that includes data processing, modeling and simulation pipeline (see Figure 1.1). During the one-year working period, the team has collaborated on most of the tasks, and I have also been working on various components of the project besides my major responsibility on the backend. In this paper, I will only focus on my major contribution towards the simulation pipeline.

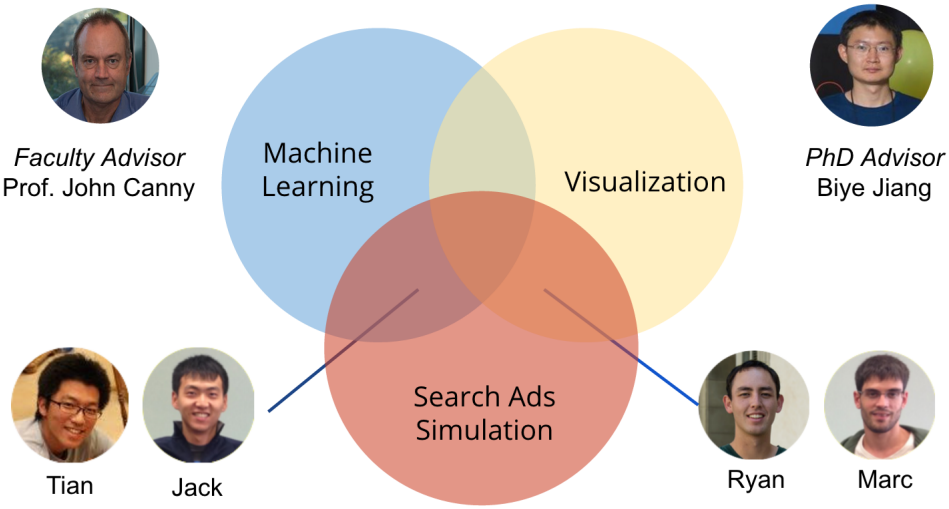

Figure 1.1: Team members and work breakdown.

1.2 The Search Ads Auction Process

The starting point of the interaction loop visioned in our project is to take an input stream of historical search ads auction data and simulate the auction process to produce various metrics that would be in the interest of our users. The search ads auction process could date back to 1998 when GoTo.com (later rebranded as Overture) created the business

model of auctioning advertisement slots on search result pages. In the process, a group of advertisers bid on the slots available on a result page, and the advertisers with the highest bids win the slots and pay what they have bid (Jansen & Mullen, 2008, p.119). Inefficient and easy to temper with, the process was greatly refined by Google in 2002 with the release of AdWords Platform, in which it established the new slot allocation method and pricing scheme that we have seen today (Jansen & Mullen, 2008, p.120). Although there are no detailed documentations or publicly accessible implementations of the process, there exist high-level summaries online from AdWords Platform that introduces the ads auction mechanism to advertisers. Thus, in order to complete our project's interaction loop, we need to implement a backend pipeline that simulates the whole process according to the public documents about search ads auction process.

This section will introduce the details of the search ads auction process that we have implemented. The whole process pipeline can be divided into two major sections: ranking and pricing. The ranking process solves the problem of allocating advertisement slots to advertisers based on a series of factors, while the pricing process solve the problem of setting prices for each slot so that the search engine company gains profit from the process. In the industrial-level search auction system, both of those processes have been optimized to a complex scale to ensure stability and prevent any attempts of gaming. Our version, however, focuses on simulating the core functionality of this process while putting efficiency and stability at a second level.

1.2.1 Ranking

Our simulation pipeline operates on the unit of auction, which corresponds to the event when a user enters a query q in the process. A group of advertisers will be matched to the query based on specific keywords in the query, and each of the advertisers would give a bid they would like to pay for the advertisement slot on the page. Given the bids placed by advertisers, the slots need to be allocated to advertisers using certain rules, often in the

order of some metrics. The order of the advertisers from the top slot to the last slot is called the ranking.

There are several metrics one could use to determine the ranking of the advertisers. The most straight-forward metric is to simply use the bidding prices alone, which is adopted in most of the physical auctions in the world as well as Overture when they invented search ads auction. However, an advertiser could place irrelevant information on the slot once it wins, which damages the user experience and diminish search engine's profitability.

To solve those problems, Google's AdWords altered the metric to take other factors into account besides the bid amount from the advertisers (Jansen & Mullen, 2008, p.123). Now, the metric for ranking the advertisers is a result returned from a **quality function**, which is usually a combination of the bid amount and quality measurements such as the expected click-through rate (CTR) or conversion rate (CVR), and other related constraints (Ye, Berkhin, Anderson, & Devanur, 2011, p.1). In the function, the expected click-through rate is usually estimated by machine learning algorithms on historical data for each combination and query, and it is used as an estimated description of the relevance of the ad to the query, based on the premise that users would be more likely to click it if the ad is more relevant. Also, the function should have the property of being always positive and monotonically increasing, so that it is always possible to determine an order given any two different advertisers. In our simulation pipeline, we retrieve the expected CTR value from an Variational Bayes Latent Dirichlet Allocation model built with Yahoo!'s WebScope dataset (Hoffman, Blei, & Bach, n.d.), and the quality score of each advertiser i is a combination of exponential functions

$$q_i = \mathrm{CTR}_i^{\alpha} \cdot \mathrm{bid}_i^{\beta}, 0 < \alpha < 1, 0 < \beta < 1$$

in which α and β are parameters that controls the relative effect of two components on the quality score. Increasing α would put emphasis on the advertisement's relevance and raise the rank of advertisements that are popular and with high quality, while increasing β

would put emphasis on the bidding value and promote advertisements who values the slots more. By adjusting these two parameters, ads operation personnel could control the effect of bid amount and expected CTR on the ranking result, which can influence advertisers' corresponding bidding decisions addressing the change. After calculating the quality score for each advertiser, we sort the advertisers based on the score and allocate them in the decreasing order to each slot.

1.2.2 Pricing

Second Price Auction

After the ad slots are allocated, the price of the slots are determined and the advertisers will be charged. Again, several different pricing schemes exists, in which the most fundamental ones are first-price auction and second-price auction. The first-price auction is a straightforward scheme in which the advertiser pays the bid amount they have placed. However, a strategic advertiser could trick the system by creating price cycles in which all advertisers have to bid and pay higher prices in order to win a higher slot, which imposes a high cost for the advertisers (Jansen & Mullen, 2008, p.123). Thus, Google's scheme adopts second-price auction, in which the bidder's payment is not determined by its own bid, but by its position in the rank and other bidder's bid (Jansen & Mullen, 2008, p.123). Auction theory shows that such auction scheme is incentive compatible under certain conditions, meaning that the optimal strategy for advertisers is to bid according to their true valuation, which would eliminate strategic playing in the auction and ensures that slots are sold to advertisers who value it the most (Jansen & Mullen, 2008, p.123).

Thus, we have also chosen to implement the second-price auction in our simulation. Specifically, we have implemented Generalized Second Price Auction (GSP), which is the dominant pricing mechanism in most search engine companies such as Google and Yahoo! (Edelman, Ostrovsky, & Schwarz, 2007, p.1). The simplest GSP states that for an advertiser at position r_i in the rank, it will be charged equal to the next highest bid, or the bid from

the advertiser at rank r_{i+1} (Edelman et al., 2007, p.2) . However, that scheme can be unfair for the advertiser at rank r_i when the advertiser at rank r_{i+1} has a very high bid (it is placed at rank r_{i+1} rather than higher ranks because of a very low CTR), which might cause the r_i advertiser to pay more than what it has bid. Thus, after consulting with our adviser, we implemented a modified version of GSP based on the thought that not only does an advertiser's payment for a slot is determined by the bid from the slot below it, but also the price should be the minimum value that could keep the advertiser to be ranked right above the next advertiser. Formally, assume the advertiser i and j are adjacent in the ranking with rank $r_i > r_j$, and quality score q_i and q_j. If advertiser i would like to obtain a score $q_i \geq q_j$, it needs to satisfy

$$q_i = \text{CTR}_i^{\alpha} \cdot \text{bid}_i^{\beta} \geq q_j$$

which gives

$$\text{bid}_i \geq (\frac{q_j}{\text{CTR}_i^{\alpha}})^{\frac{1}{\beta}}$$

We denote the value on the right hand side to be the **inverse quality function**, and the price each advertiser would pay is therefore the output of inverse quality function applied with the next advertiser's quality score and its own CTR.

Reserve Pricing

The second-price auction scheme encourages advertisers to bid closer to their true valuation of the slots, but it is still possible for advertisers to simultaneously place their bids at a low level, which damages the overall profitability of search engine companies. For example, consider the case when there are fixed costs for publishing ads on the search result page due to system implementation and maintenance. The search engine company would like the revenue from each auction to be able to cover the fixed cost, while in reality advertisers might not value the slot as high and give bids that are lower than the fixed cost. As a result,

the search engine company would bear a loss for the auction if the slot is sold.

To make the auction generate highest payoff for the seller (search engine) and avoid the situation of bearing a loss when slots are sold, a reserve price is usually added to the auction, in which the search engine slot is only "sold" when there are bids that reach the preset value. Ostrovsky and Schwarz have shown that when correctly set to an optimal value, reserve price helps the revenue generated from sponsored search auction to increase substantially, and the effect is especially pronounced for queries with relatively high volumes and for queries with a relatively small number of bidders (2009, p.2).

To model the search ads auction process more closely and enable users to see the effect of reserve pricing on the auction metrics, we have also implemented reserve pricing in our simulation. Specifically, after we obtain the ranks of advertisers r_1 to r_n, we only accept the amount from the top advertisers which has a bid $b_i > p_r$, where p_r is the reserve price. Then we apply the second-price auction mentioned before, with the last advertiser remaining paying the amount equal to the reserve price p_i.

1.3 Implementation Details

The ranking and pricing are two major backbones in the simulation pipeline which was implemented following the algorithm introduced in the previous section. However, as one of the components in the back end of the graphical tool, there are more components to be implemented in order to link the process from input data to output metrics that will be consumed by the visualization. This section will give an overview of the system and introduce the implementation details of the simulation pipeline.

1.3.1 System Overview

Our simulation system is a pipeline that takes in records of previous search ads auction transactions and run the search ads auction process introduced in Section 1.2. As Figure

1.2 shows, the pipeline is divided into 5 stages in order: BIDMach DataSource, auction decomposition, ranking, pricing and metrics calculation. Each stage accepts the output of last stage directly and do more computations on top of it. On the scale of the whole visualization tool, the simulation utilizes values from the click-through rate model (which is introduced in detail in Ryan's paper), and receives values of parameters of α, β and reserve price from the front end (see Figure 1.3).

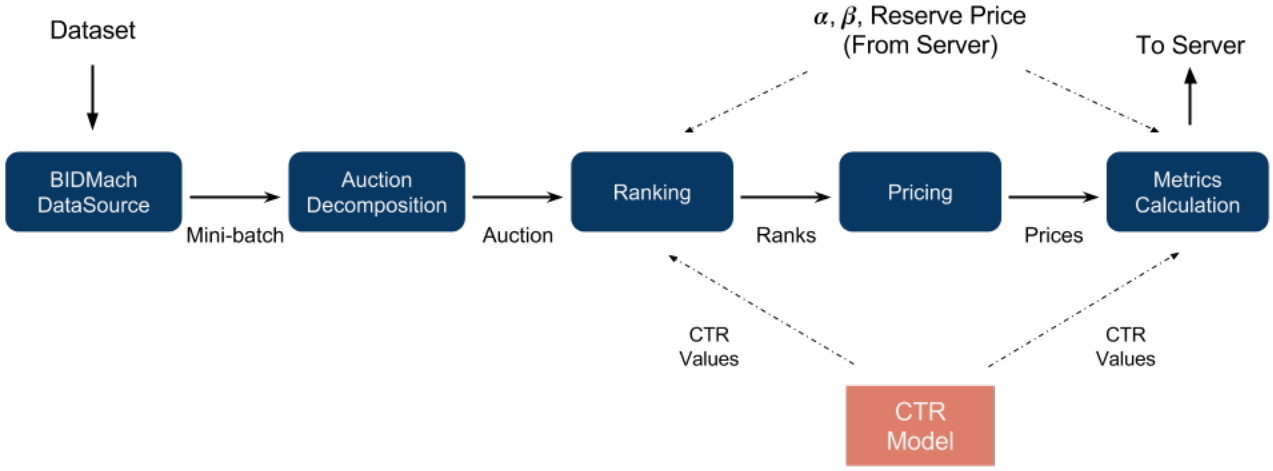

Figure 1.2: The simulation pipeline. The ranking and metrics calculation stage uses the CTR value from the CTR model, and the user-input parameters are passed in by the web server.

1.3.2 Accepting Input Data

Ideally, the input data of the simulation pipeline should be a continuous stream of search ads auction records. The stream of data could come from an actual sponsored search system, from which the records of bidding would flow into our system, and our system would simulate a search ads auction once we receive the records for a full auction. However, in reality we have no such source of data, as search records are usually considered proprietary information

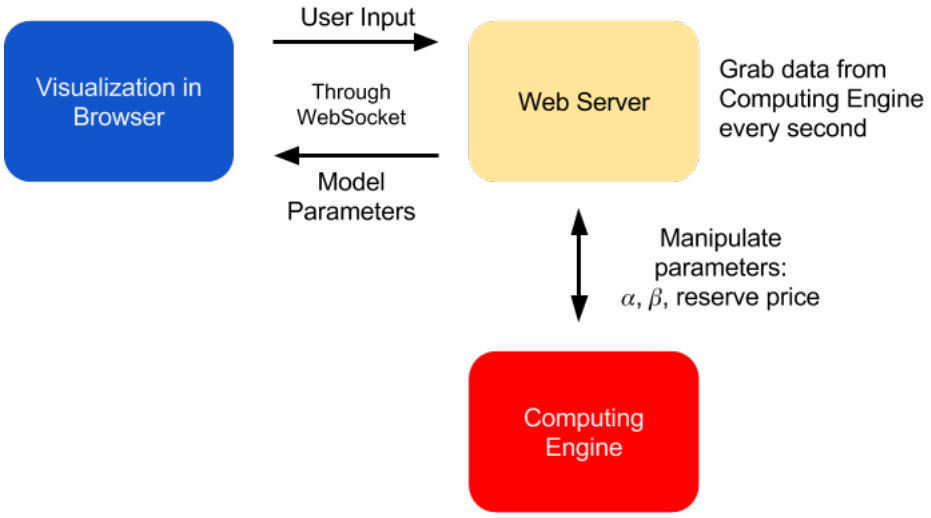

Figure 1.3: The visualization tool diagram. The Computing Engine consists of the simulation pipeline and the CTR model, which receives user-input parameters from the web server and sends the metrics to server for plotting. Figure modified from Biye Jiang and John Canny, *Interactive Machine Learning Using BIDMach*.

of search engine companies. Therefore, for the purpose of constructing a prototype, the Yahoo! WebScope Dataset (which we have used to model the expected click-through rate) are reused to be the input data of the simulation. The dataset consists of more than 70 millions of bidding records from search ads auctions sampled from the range of 127 days, and each record contains the hashcode of advertisers and queries, the amount of bids, rank assigned (in the real-life auction), and impressions and clicks received after the auction. To imitate a data stream, we have preprocessed the dataset by sorting on *day* column first and then randomizing the order of queries, so that the records in the dataset is in a random order of auctions happened in each day. The detail of the data wrangling process is introduced in Tian's paper.

Having data coming from a whole dataset means that the input data is not a stream but rather a batch. However, given the size of the dataset (about 8GB) it is difficult to load the whole dataset in memory at once and execute the simulation logic on top of it. To keep the stream processing nature of our pipeline while being compatible with the dataset size, we

have chosen to adopt the mini-batch data processing scheme in *BIDMach*, a large-scale machine learning library that has single-node performance exceeding most cluster-based system and used in our project for the CTR model. BIDMach's *Learner* framework is designed to process large dataset into small mini-batches to support batch operations in machine learning algorithms (Canny & Zhao, n.d., p.2), which is very similar with our simulation pipeline as the latter also takes batches of auctions and simulate the results for each auction. Thus, we have implemented the input data consumption following BIDMach's Learner framework by using BIDMach's *DataSource* abstraction. Specifically, the pre-processed dataset is coded with numbers and processed into BIDMach-format matrices, which is picked up by a *DataSource* instance. In runtime, the *DataSource* instance will iterate over the matrices and automatically slice them into mini-batches with a preset size, and the simulation will run auction logic (including ranking and pricing) on each mini-batch of data.

1.3.3 Auction Decomposition

As mentioned in previous section, the lack of a stream of search ads auction data have imposed a challenge on our simulation pipeline to accept a batch of records. Although that problem is solved by adopting BIDMach's *DataSource* abstraction, the pipeline still need to decompose each mini-batch into record groups for each auction. Ideally, records from the same search ads auction would contain the same query and same day timestamp, and each record from the same auction should have distinct yet continuous ranks. Thus, in theory we could decompose the mini-batch into auctions by grouping on the query column and day column of the data.

However, during implementation we found out that the grouping operation did not give us a reliable decomposition of auctions. Specifically, as the dataset is filtered and sampled from real sponsored search logs, the records under the same query for the same day contains duplicate and non-continuous ranks, which means those records are from different auctions happened on the same day under the same query and could be further decomposed. However,

it is challenging to carry out the further decomposition as there are no further information from the dataset for us to identify the auction each record is from. Thus, we resolved this problem by giving a heuristic of the number of auctions in a record group of day-query and evenly allocating records into each auction randomly. We used the count of the mode of the rank column in a day-query group, with the reasoning that the number of auctions in such a day-query group must be larger than the count of every rank in the rank column. For example, if rank r_x appeared x times and rank r_y appeared $y < x$ times among the records in the group of day d and query q, the number of auctions in this group must be no less than x because each rank only appear in each auction once. After decomposing the mini-batch into auctions, records from each auction are sent into the ranking and pricing stage for simulation.

1.3.4 Profit Metrics Calculation

In industry-level systems, a search ads auction ends at the point when prices have been set for slots on the page. After that, the advertisers pay the search engine company based on the amount of some user-related metrics. Early in the history of search ads auction, the search engine used "pay-by-impression" model, which directly comes from the traditional print media where the most available metric is how many copies of ads have been sold with the print media (Jansen & Mullen, 2008, p.125). However, as Jansen et al. mentioned, counting views (or called impressions) is hard for online media as each company has its own methodology to determine what qualifies as a view (2008, p.125). which makes it hard to justify the amount charged from advertisers. Later, the payment mechanism was refined again by Google's AdWords platform with "pay-by-click" model, in which advertisers only pay when user clicks on the ad (Khan, 2014, p.15).

As a result, the price and the profit metrics are separated after an auction has been run. To finish the simulation with the profit metrics that ads system operators care about, we estimate the number of clicks per thousand-impression using the estimated click-through

rate for each advertiser with the current query it bid for and rank it obtained. Then, each advertiser i would generate a profit

$$P_i = p_i \cdot \text{CTR}(i, q, r_i) \cdot 1000$$

where $\text{CTR}(i, q, r_i)$ is the expected click-through rate of advertiser i for its rank r_i and current query q, which comes from the CTR model we built using WebScope Dataset as well. The output of the simulation for each auction is an array of BIDMach matrices, where each matrix is a tuple containing auction ID, advertiser ID, profit-per-thousand-impression from that advertiser, and other potential metrics to be added later. Such design leaves the task of aggregation to the visualization, allowing it to be able to display both overlooking metrics (like total profit) or granular metrics (such as profit per advertiser). Also, it makes it easy for us to expand the type of metrics we can output beyond profit, as other metrics such as expected ranks for advertiser can also follow this format.

1.3.5 Parameter Feedback

One important feature of our visualization tool is that the user of the tool (usually ads operation personnel) can change the parameters of the auction in real-time once they see the metric plots. Thus, the simulation pipeline need to be able to record parameter changes when user requests to do so. Currently, the simulation parameters exposed to users are α, β, and the reserve price p_r. However, there is the possibility that more parameters could be exposed to the user and it would be a hassle to modify the source code to add getter/setter method for each new parameter every time. In order to make the pipeline extendable, we decided to use Java reflection to implement a unified update function, which essentially enables the pipeline to dispatch a string into a field name in the simulation Scala class to allow operations such as *simulation.setParam("paramX", valueX)*. Using reflection saves time for the frontend development from using multiple getter and setter methods for different parameters, and it

makes it easier later to add feedbacks for other parameters between frontend and backend.

1.4 Result

For each mini-batch of search ads auction records, the simulation pipeline outputs a set of metric records in BIDMach matrix format to the frontend. Each metric record is a row containing an numerical ID for the auction assigned by the simulation pipeline in the order of encounter, the identifier for the advertiser who participated in the auction, and a list of metrics related to the advertiser, such as the profit the advertiser contributed in the auction and the estimated clicks for the advertiser, etc. As mentioned in Section 1.3.4, the frontend server takes the metric outputs for each mini-batch from the simulation pipeline and conducts further aggregations on the metrics to produce 1 value point per mini-batch shown on the real-time graph (see Figure 1.4). The details about the frontend is introduced in Ryan's paper.

Later in the testing phase with the frontend, we conducted an empirical test on the capability of the simulation system by varying the size of the input mini-batch size. The testing shows that the simulation pipeline is capable of producing metric outputs for a mini-batch of up to 50000 records per second, which roughly contains about 14000 search ads auctions. With such capacity, the effect of tuning the parameter value could be shown immediately in the next several mini-batches of data. The computation time on each mini-batch started to be noticeable when we raised the mini-batch size to 100000, causing each update on the front real-time graph to lag for 2-3 seconds. We have not found any existing benchmarks on systems similar to our implementations, but in the future we could increase the capacity by parallelizing the simulation pipeline to be comparable with real-life search ads systems.

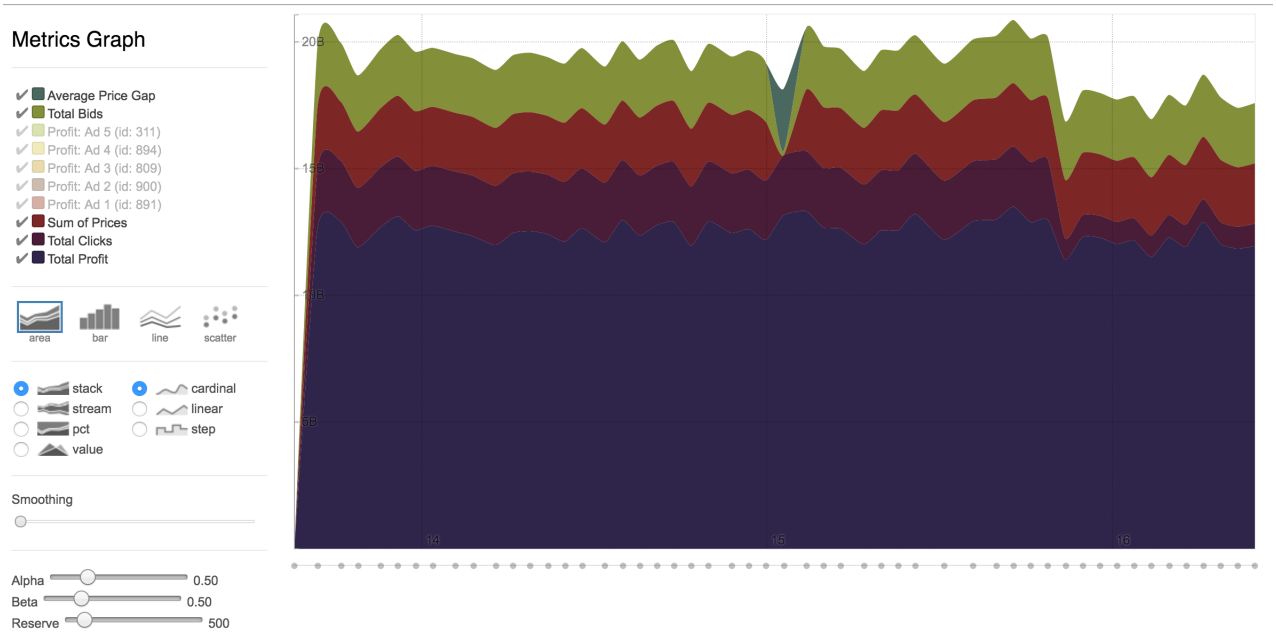

Figure 1.4: The visualization interface. Each colored area in the real-time stacked graph represents a metric. For a metric, the value of each point on the graph is aggregated from the metric values outputted from the simulation pipeline. By toggling the bar in the left-bottom corner, parameter values will be updated and the simulation pipeline would execute the next mini-batch using the new parameter values.

1.5 Future Works

As a proof-of-concept prototype, in the future there are a lot of extensions that we can add to the simulation pipeline. The most immediate and easiest one would be to expand the type of metrics outputted. Currently, our simulation pipeline can capture revenue metrics such as total profit from the auction and the price each advertiser paid for clicks, which are both focused on maximizing search engine's profitability. We can also use our CTR model to estimate the amount of clicks each advertiser gets after the auction. However, a well-tuned search ads auction should balance between 3 shareholders in total, namely the search engine users and the advertisers besides the search engine itself, which means ads operation personnel will also be interested in visualizing metrics related to user satisfaction and advertiser interests. In the future iterations of the tool, we could derive metrics such as the conversion rate of the user and number of ads shown/clicked per advertiser, to describe the trend of user satisfaction and advertiser interest along with the fluctuation of profits.

Another possible extension is to parallelize the simulation pipeline with a distributed computing system. Currently, the logic of the simulation pipeline is implemented in native Scala code sequentially. It is easy to implement the algorithm sequentially as it follows exactly what the algorithm specifies and debugging on the sequential code requires little effort. However, our simulation pipeline runs on the unit of auction, which mean the simulation logic could be run simultaneously on multiple groups of auction data. Also, parts of the simulation logic such as computing quality score for each advertisers requires no information from other components and could utilize matrix operations to be run in parallel. The parallelization of simulation pipeline would increase the speed of processing and enables our tool to consume much large dataset later, and it would be helpful for our tool to transfer into a product that is able to cope with the industry-level rate of data generation.

Finally, the simulation pipeline could be extended to take on other analytic tasks related to search ads auction. For example, one of the proposals from our contacts in Yahoo! is to explore interactive clustering on auction data, in which auction data is first clustered

by queries or advertisers using machine learning algorithms, and then piped into auction simulation in order to compare the difference of metrics between each cluster and obtain optimal parameter settings for each cluster respectively. In this case, the main simulation logic we have implemented could be reserved while the data source can be switched to take in clusters of data, and additional parameters can be added to enable users to choose to run simulation on selected clusters from the front end.

Chapter 2

Engineering Leadership

This chapter is composed collaboratively and shared in reports of all team members.

2.1 Introduction

The digital advertising industry is booming - revenue from online advertisements was $13.6 billion in 2015, and will only continue to explode as internet traffic grows exponentially (Turk, 2015, p.3-5). Thus, even a small improvement in the way that advertisements are displayed would have an enormous impact on bottom lines for both search engines and advertisers. Our project seeks to develop an interactive, web-based graphical tool in order to optimize the bidding process for search engine advertising. Visualizations will allow search engines to better analyze performance metrics in real time and help deliver the best possible ad experience for both advertisers and users. This paper will conduct an industry analysis of the search advertising industry, survey the current landscape for both direct and indirect stakeholders as part of a market analysis, and explore potential intellectual property issues.

2.2 Industry Analysis

The search advertisement industry is relatively new, beginning in 1998 when GoTo.com (later rebranded as Overture) created the first search advertisement auction, in which advertisers placed bids for slots on the resulting search page (Jansen & Mullen, 2008, p.119). The Overture model was successful but inefficient, and was succeeded by the Google AdWords platform in 2002, which set the industry standard with a more profitable pricing/allocation method (Jansen & Mullen, 2008, p.120). In the Google model, after the bids are placed, the ranking of advertisements is determined by many factors including the bid price and the estimated click-through rate (CTR) (Ye et al., 2011, p.1). The search engine charges advertisers using a pay-by-click model, in which advertisers only pay when users click the ads (Khan, 2014, p.15).

As the internet has become more prominent in peoples lives, the search advertisement industry has grown tremendously, which has generated a huge demand for the software technology that powers the industry. According to an IBISWorld report, industry revenue is expected to grow 7.3% annually, and has a strong base for future growth as it survived the recent global recession unscathed (Khan, 2014, p.5). Advertisers who had previously spent conservatively during the recession have begun to scale upwards on their advertising budgets, which helps the overall growth of the industry (Khan, 2014, p.5). The increased advertiser demand has been encouraging technology upgrades - if a search engine company falls behind on adopting new technologies, the quality of search traffic decreases, so advertisers relocate their budgets to other competitors (Khan, 2014, p.21). Thus, companies are always looking for software that helps fine-tune their search advertisement operation strategy, which is where our project comes into play.

In a highly competitive environment for technological development, large companies with massive resources have dominated the market. The biggest company is Google, which controls 75.2% of the market with its AdWords platform (Khan, 2014, p.26). Other major players include Yahoo! Inc. and Microsoft, who formed a duo in 2010 by sharing core

algorithms and revenues (Khan, 2014, p.28). Facing fierce competition, we focus on differentiating ourselves by developing tools that can be used for daily monitoring of auction processes by Ad Operations teams, instead of by advertisers.

2.3 Marketing and Stakeholders

Now that we have an overview of the industry, we can analyze the stakeholders and market opportunities. The direct stakeholders of the project are search engine companies, and in particular the Yahoo Ad Operations team, who will use the real-time, interactive visualizations to guide critical business decisions and tune parameters to build the optimal model for ad auctions. Although advertisers and the users of the search engine are obscured from the inner workings of the advertisement auction process, they are also indirectly affected by the choices made by search engine companies.

2.3.1 Direct Stakeholders - Search Engine Companies

Search engine companies face the daunting task of optimizing the advertisement auction process in order to maximize revenue. Current machine learning solutions can deliver an answer to minimize a given cost function, but the issue is that many of these solutions resemble a black box, as there is no way to see how the different variables interact or how changing a parameter can affect different parts of the model. Our tool helps expose the internals of the algorithm, providing knobs to turn that enable the user to instantly compare performance tradeoffs, such as sacrificing profit to keep advertiser and/or user satisfaction high.

2.3.2 Indirect Stakeholders - Advertisers and Users

An advertisers primary goal is to display its advertisement to the correct target audience. If advertisers are unable to consistently win auctions at reasonable prices, they will be unable

to spend their advertising budget, and be forced to move on to other methods of advertising. A survey of 41,548 advertisers demonstrated that 42% of them had no exposure on the first page of search results on Yahoo Bing Network (Hamilton, 2013). To mitigate this issue, we include advertiser satisfaction performance metrics such as ads shown per advertiser, and average cost per ad. Additionally, we include user satisfaction metrics like click rate and conversion rate for Yahoos 800 million monthly users, who would like to be shown relevant ads that can enhance their online experience by directing them towards new products they might not have otherwise discovered (Gallagher, 2013). Our visualizations help search engine companies track user satisfaction via metrics such as click-through rates.

Varian (2014), Googles Chief Economist, argues that market forces at play give search engine companies an incentive to increase advertiser and user satisfaction. If user satisfaction is high, they are more likely to click the ad and go through with the purchase. This makes advertisers happier, as they make more profit per click, and can now afford to bid more per click, since their conversion rates are higher. Search engine companies are also happy, as higher bid prices and higher click rates both equate to more revenue. Our tool helps search engine companies balance the advertiser and user satisfaction metrics with other performance goals, and run simulations to weigh potential consequences of different choices.

2.4 Intellectual Property

As we move forward with the project, there are important intellectual property issues that we need to consider. The general purpose of the project is to build a tool that enables a user to have real-time visual feedback when the parameters of a machine learning model are tuned. This idea is very close to a patent filed by Microsoft Corporation in November 2014, titled Interactive Optimization of the Behavior of a System, which would make it very difficult to patent the general idea of our project. (Kapoor, Lee, Tan, & Horvitz, 2014). However, the Microsoft patent is not very specific with regard to potential applications, so

perhaps we could file a more detailed patent addressing the specific problem of ad ranking optimization.

Although we might have a chance in the future to patent our projects idea, we currently do not see a clear opportunity to pursue a patent. First, our project is closely related to the thesis work of a UC Berkeley EECS PhD student, in that both projects leverage the computing power of a GPU through a dedicated library (BIDMach) to build an interactive visualization tool, and we have received direct help from that student. Thus, it may not even be possible to patent this work, as it is too close to works belonging to the university. Furthermore, in the short term we do not see a need to try to patent our work, as we are not oriented towards a full product, but rather a research question about the value of interactive visualizations. However, if the project prototype turns out to be extremely valuable or we find an interesting use case eventually, we could possibly utilize the intellectual property to start a company when the use case is sufficiently different from the PhD students work. Finally, open-sourcing is not as viable an option as it once was, since it makes it difficult to find VC funding and generate privately held intellectual property.

Building a full product out of this project would require us to further study market opportunities, and most likely change the use case. Ad Operations is a relatively small market in which there are only a few main players, so it is not likely to have a large customer base to penetrate. We could apply the same tools and techniques to other problems, such as eSports and online betting markets where there might be more market opportunity, but this would still be at least 1 to 2 years on the horizon.

References

Canny, J., & Zhao, H. (n.d.). BIDMach: Large-scale learning with zero memory allocation. Retrieved March 15, 2016, from http://www.eecs.berkeley.edu/ñzhao/papers/BIDMach.pdf

Edelman, B., Ostrovsky, M., & Schwarz, M. (2007). Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. Retrieved March 10, 2016, from http://www.benedelman.org/publications/gsp-060801.pdf

Gallagher, B. (2013, September 11). Yahoo monthly active users are up 20to 800m, including 350m on mobile, says marissa mayer. *TechCrunch*. Web. Retrieved October 20, 2015, from http://techcrunch.com/2013/09/11/marissa-mayer-yahoo-monthly-active-users-are-up-20-to-800m-including-350m-on-mobile/

Hamilton, G. (2013, April 4). Google adwords vs yahoo bing network - a ppc performance comparison. *Search Engine Watch*. Web. Retrieved October 20, 2015, from http://searchenginewatch.com/sew/study/2259377/google-adwords-vs-yahoo-bing-network-a-ppc-performance-comparison

Hoffman, M. D., Blei, D. M., & Bach, F. (n.d.). Online learning for latent dirichlet allocation. Retrieved May 2, 2016, from https://www.cs.princeton.edu/~blei/papers/HoffmanBleiBach2010b.pdf

Jansen, B. J., & Mullen, T. (2008). Sponsored search: An overview of the concept, history, and technology. *Int. J. of Electronic Business*, *6*(2), 114–131.

Kapoor, A., Lee, B., Tan, D. S., & Horvitz, E. J. (2014). *Interactive optimization of the*

behavior of a system (Nos. 8,898,090). (Microsoft Corporation, assignee.)

Khan, S. (2014). IBISWorld industry report 51913a: Search engines in the us. Retrieved October 15, 2015, from http://clients1.ibisworld.com/reports/us/industry/default.aspx?entid=1982

Ostrovsky, M., & Schwarz, M. (2009). Reserve prices in internet advertising auctions: A field experiment. *Int. J. of Electronic Business*. Retrieved November 27, 2015, from https://faculty-gsb.stanford.edu/ostrovsky/papers/rp.pdf

Turk, S. (2015). IBISWorld industry report OD5889: Digital advertising agencies in the us. Retrieved October 15, 2015, from http://clients1.ibisworld.com/reports/us/industry/default.aspx?entid=5889

Varian, H. (2014, June 18). *Insights on the adwords auction.* Online video clip. Youtube. Retrieved October 18, 2015, from https://www.youtube.com/watch?v=PjOHTFRaBWA

Ye, C., Berkhin, P., Anderson, B., & Devanur, N. R. (2011). Real-time bidding algorithms for performance-based display ad allocation. Paper presented at 2011 ACM SIGKDD Conference on Knowledge Discovery and Data Mining, San Diego, August 21-24.