A Simulation Framework for Evaluating Designs for Sponsored Search Markets

Soam Acharya, Prabhakar Krishnamurthy, Ketan Deshpande, Tak W. Yan, Chi-Chao Chang

Yahoo! Inc.
2821 Mission College Boulevard
Santa Clara, CA 95054
408-349-5000

{soam, pkmurthy, deshpa, tyan, chichao}@yahoo-inc.com

Abstract

Sponsored search is a rapidly growing business and there is tremendous industry and research interest in improving the designs and functioning of the sponsored search marketplace. Launching new designs and enhanced features for the sponsored search marketplace requires careful evaluation of their potential consequences to user experience and financial impact on the multiple parties (advertisers, publishers and marketplace operator) involved. The complexity of market dynamics makes it difficult to draw definite conclusions about the market without comprehensive testing. While limited field testing is often performed, it has several disadvantages: limited control over design parameters, limited sample sizes and scenarios that can be tested. Simulation testing is a viable option. Though some previous works have reported on the use of simulations, most of these are ad hoc and intended to test specific scenarios.

In this paper, we describe the design of a general purpose simulation framework that supports the evaluation of alternative designs and features. We initially discuss the functional and architectural requirements for implementation of this framework. From a methodological perspective, there is a need to simulate a "micro-market" – a small scale representation of a complete market – for effective evaluation. Hence, we next describe how micro-market data samples are generated and an approach to scaling the metrics produced from simulations, using such samples, to represent an entire market. Finally, we relate our experiences in applying this simulation framework to evaluating designs and features for sponsored search at Yahoo!

Categories and Subject Descriptors

J.4 [**Social and Behavioral Sciences**]: Economics.

General Terms

Algorithms, Measurement, Performance, Design, Economics, Experimentation, Standardization, Theory, Verification.

Keywords

Sponsored search, simulation, frameworks, sampling, Yahoo!, keyword marketplace, budgeting.

1. Introduction

Sponsored search has been growing rapidly as an online advertising industry over the last few years. Companies in this space are constantly innovating. At the same time, there exists much research interest in improving the designs and functioning of the sponsored search marketplace. Areas of exploration include: auction design, ad ranking algorithms, pricing algorithms, advertiser budget optimization, and ad matching techniques.

Launching these innovations require careful evaluation of different aspects of the marketplace. Changes in the marketplace have large impact on the following:

i) The experience of the millions of users who view and click on ads;

ii) Advertisers who depend on the leads generated from sponsored search – some advertisers exclusively rely on the Internet for generating leads and sales of their products and services;

iii) Publishers who display the ads on their web sites; some publishers are dependent on sponsored search as their main source of revenue;

iv) The marketplace operator generates substantial revenues from sponsored search.

The users, advertisers and publishers all react to each others actions, based on the information available to them. The scale of the marketplace is large – millions of users view millions of advertisements from hundreds of thousands of advertisers. The complexity and scale of marketplace dynamics make it difficult to draw definite conclusions about the market without comprehensive testing. While limited field testing is often performed, it has several disadvantages: limited control over design parameters, limited sample sizes and scenarios that can be tested.

In this paper, we present simulation testing as a viable option. Though some previous works have reported on the use of simulations, most of those are ad-hoc and intended to test specific scenarios. We present a simulation system called Cassini that supports a more general purpose evaluation of alternative designs and features. The system is in active use at Yahoo!

In the remainder of this paper, we first outline related work. Next, we discuss the functional and architectural requirements for implementation of this framework. From a methodological perspective, there is a need to simulate a "micro-market" – a small-scale representation of a complete market – for effective

evaluation. Consequently, we next describe how micro-market data samples are produced, and an approach to scaling the metrics produced from simulations using such samples to an entire market. In the results section, we relate our experiences in applying this simulation framework to evaluating designs and features for sponsored search. Finally, we present our conclusions and next steps.

2. Related Work

Simulation modeling of marketplaces has been used for either comparing marketplace designs or bidding strategies. With respect to auction marketplaces, Csirik et al [3] describe a simulator intended to be used by bidders participating in FCC spectrum auctions. While these auctions are not web-based and in other ways differ significantly from sponsored search auctions, their work is interesting for being one of the first simulations of real world auction based markets and sharing some of our design objectives. Their simulator implements the detailed rules of the auction and provides flexibility of representing alternative bidding strategies. Their framework supports customization and allows the framework to be extended to simulate other combinatorial auctions and bidding strategies. The FCC itself is reported to have used simulation as a method to understand the impact of alternative spectrum auction rules in the process of designing the auction. On a related note, Powell [8] describes a Java based toolkit for simulation of adaptive/non-adaptive agents for Market Based Control tasks using Double Auctions.

Simulators have been designed and implemented for internet based auctions as well. The Michigan Internet AuctionBot [9] is an early example of a software test bed intended for classroom testing of various auction mechanisms. Bapna et al [2] describe a simulation developed to test the effects of various parameters in Yankee-type auctions (Yankee-type auctions are a popular auction format used by online merchandise auction sites such as eBay). They present their simulator as a tool for use by marketplace operators to investigate design options. Similarly, Anthony et al [1] detail an agent able to participate simultaneously in multiple Internet auctions of different types including English, Dutch and Vickrey.

Simulators for sponsored search auctions have been developed by Feng et al [5] and Kitts and LeBlanc [6]. While Feng et al [5] use simulation to compare the equilibrium performance of alternative methods of ranking online advertisements for placement alongside search results, Kitts and LeBlanc [6] describe a system to compare the performance of different trading strategies (i.e. advertiser bidding strategies) under a specific set of scenarios. Both these simulators are implemented specifically to study specific scenarios.

The design we present in this paper is intended to serve as a general purpose test bed for sponsored search marketplace design. By incorporating the advertiser, user and system aspects of the marketplace, our approach attempts to be as comprehensive as possible. In addition, we focus on methodological aspects of simulation as well. Certain design aspects that are important to us – budget management, for instance – pose problems of data sampling and scaling the results to represent real search traffic and are not addressed elsewhere. Here, we present our solutions to these issues as well.

3. Motivation for Offline Simulation

Given the complexity and scale of the marketplace, one option for testing a marketplace design scenario would be to utilize the production system itself: divert a portion of the overall user web search traffic to the experiment and evaluate the subsequent results. However, real-world testing of this type is limited by several factors:

- The fraction of traffic utilized for testing each new configuration is typically small. This is done to mitigate risk. However, there are cases when the traffic level might be insufficient for adequate evaluation. Depending upon the position of an ad and its quality, the probability of a click can be very small. Such ads need to be displayed large number of times to obtain a realistic distribution of clicks.
- Isolating all design factors in a production system is often not possible. Live testing does not guarantee that ads from an advertiser will only appear in the tested traffic. Consequently, it is difficult to draw conclusions about factors such as user experience, advertiser budgets and reactions, given that these are influenced by the entire marketplace.
- Given the relatively small sample sizes, a live test must run for several days in order to gather statistically significant data. This lengthens the turnaround time of each test as a wait is necessary before we can be sure of the results and limits the experiments that can be performed.
- Production testing also involves additional overhead in addition to the experiment itself. The deployed system must meet production Service Level Agreement (SLA) requirements – consequently, the experiment must be designed and coded to meet such requirements. These can limit scope of the designs being tested and cause preparation delays.
- A key consideration in evaluating a marketplace design is its long term effect on advertiser behavior (second order effects). However, live testing, due to its inability to isolate the various design factors, limited sample sizes, and, to a lesser extent, the limited duration of each test does not provide an effective way for determining second order behavior. An alternative involves staging mock auctions with real bidders but these are difficult to set-up, can be time-consuming and are inappropriate for large-scale marketplaces.

An offline simulation framework can go a long way towards addressing the problems inherent in live testing. We can devote the entire traffic to a single experiment; hence samples sizes are not such a big factor during evaluation. Similarly, it is easier to isolate design factors such as budgets given that we control the simulation environment. The process of implementing and evaluating experiments is also significantly faster, thus cutting down on the candidates for testing. Finally, while a simulation framework by itself is not necessarily an accurate predictor of second order effects, it can be used in conjunction with other work to create scenarios which model the consequences of advertiser behavior. However, the design and implementation of

an effective framework poses its own set of requirements and challenges. We detail those next.

4. Simulation System Requirements

Requirements for a general purpose simulation system can be derived from consideration of the sponsored search marketplace and range of design decisions that need to be made.

4.1 Marketplace Technologies

The simulation system must be able to mimic marketplace technologies such as:

- Ranking and pricing
- Advertiser budget management
- Advertiser participation constraints
- Query to ad matching
- Auction formats

Since these are the technologies for which alternative designs are being tested, the simulation system must allow new designs to be "plugged in" easily. Hence, it is important to be able to interface with external modules that implement specific marketplace policies and features that are included in the comparison set. Additionally, many different teams are involved in the development of these designs and they are likely to prefer their own programming paradigms for developing modules. In some cases the modules may be part of production system. The simulation framework should thus be able to execute in tandem with these external modules by exchanging data and parameters via well defined interfaces.

4.2 Advertiser Behavior

In the repeated auction setting of sponsored search, advertisers are adaptive. They respond to user behavior, marketplace changes, and to each other's actions. In fact, many advertisers are known to use automatic bid management tools which actively change bids and budgets in response to marketplace conditions.

For most scenarios we can simulate equilibrium conditions where advertiser behavior is non-adaptive. However, we are often interested in how the equilibrium would shift under a different policy, say for ranking ads. In such cases it is important to model the adaptive behavior of advertisers. In the present version of our simulation system, we have assumed non-adaptive bidding agents. With this assumption, we try to gauge the reactions of advertisers to alternative policies by interviewing a sample set of advertisers and estimating the conditions at equilibrium. The new equilibrium states are provided as inputs to the simulation.

4.3 User Behavior

While user behavior changes over time – in terms of the keywords that are searched, click behavior, etc., we assume their behavior is static in the short run. Clicks are generated by users and sponsored search revenues are based on user click behavior. Therefore a robust and accurate click model is very important. There is another aspect to this: marketplace designs for ranking and pricing ads in an auction rely on prediction of users clicks in their decisions. Designs employing the same click model as the simulator will tend to perform better in the simulation for this reason alone. It is important to keep this in mind while performing simulation experiments that compare the performance of different prediction models. Consequently, the simulator should support a family of plausible alternative user click models.

4.4 System Performance

From a performance standpoint, the primary requirement is the ability to process large volumes of data quickly. This is necessary for both simulation preparation and execution phases.

In the preparation phase, actual web traffic logs constitute a primary source of information for both phases. First, the logs are processed to extract information that will be used to train various statistical models. Next, the logs are also gleaned to obtain user access data that is used to drive the actual simulation. Due to the heavy volume of Yahoo! search traffic; this necessitates iterating through billions of rows of log information.

In the execution phase, the system must be able to complete each session in a timely fashion, faster than the searches in the corresponding trace. In addition, the system should lend itself to automation – be able to repeat each simulation session multiple times without user intervention. Doing so allows us to determine the statistical significance of metrics generated by various scenarios.

While some scenarios can be tested on relatively small samples of data, simulations involving advertiser budget management, query matching techniques etc. require large samples – running multiple trials and scenarios against large samples requires high degree of simulator performance – ideally each scenario completing in under an hour.

5. Cassini System Design

In this section, we present the design of our simulation system Cassini, and explain how we address the system requirements above.

5.1 Cassini Core Modules

Cassini is implemented in C++ and Perl (about 20K lines). Cassini modules can be divided into four different types: those mimicking marketplace technologies, approximating user behavior, emulating advertiser behavior and keeping track of simulation state.

5.1.1 Marketplace Technologies

Queries are the primary simulation drivers in Cassini. A query sequence is usually derived from Yahoo search traffic. Once a query is injected into the Cassini pipeline, the first module it encounters is usually the Query/Ad Matching module. Here, the query is paired with a list of potential ad candidates. These candidates are usually actual ads from live data.

The candidate ads pass through the Budget Filtering module. This module eliminates ads if, for example, the advertiser is over budget. The remaining ads are then ranked and priced by the corresponding Ranking and Pricing modules.

Next, the Click Generator, utilizing pre-calculated models, determines whether the current ranking of ads attracted any clicks. The Budget and Advertiser module is responsible for performing bookkeeping on advertiser accounts that have undergone recent spending activity.

There exist multiple versions of certain marketplace modules to better simulate the various technologies on trial. Cassini allows mixing and matching of modules depending on the scenario being

modeled. For greater flexibility, modules can also expose input parameters that are set at runtime. For example, the module implementing pricing has an option to change the minimum bid of the auctions. In addition, Cassini also allows modular functionality to be bypassed. A common Cassini usage involves outsourcing the ad matching, ranking and pricing to a different system (illustrated by the dotted lines in Figure 1). This greatly eases the process of collaborating with other groups at Yahoo!

5.1.2 Keeping Track of Simulation State

Cassini contains several modules maintaining simulation state. In addition to the budget related modules tracking account and campaign spend, the Metric module keeps track of various useful summary metrics such as those described in Section 6.1. The module outputs the metrics at the end of each simulation.

The Simulation Log output module dumps event information in log file format during the course of each simulation. This can be loaded into a database for subsequent detailed post processing analysis. Cassini also possesses the ability to iterate multiple simulation sessions with the same set of inputs, particularly useful when determining confidence intervals for output metrics.

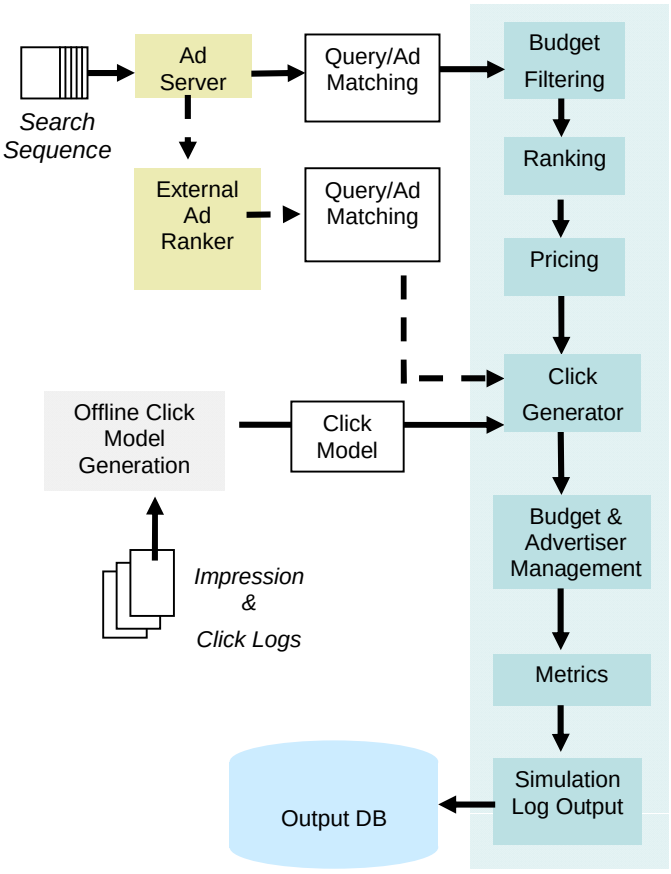


Figure 1: Components of Cassini Simulation Framework

5.1.3 User Behavior

As Cassini is often run with traces directly derived from actual search logs, it is possible to preserve the temporal properties of user accesses for testing those time dependent marketplace components such as budgets. The other aspect of user behavior incorporated into Cassini is that of user clicks. These are based on models that predict click probability given a query, a set of ads and the type of page containing the search results. Training these models from historical log data forms a considerable part of the simulation preparatory work. During run time, factors such as the prior success rate of the ad as well as the advertiser, the position each ad appears in, etc. are combined to provide a click probability estimate for each "page." A random number generator then generates synthetic clicks based on these values.

5.1.4 Advertiser Behavior

Because fine grained continuous emulation of advertiser behavior is difficult, we opted for a static approach. Currently, Cassini allows a number of advertiser attributes to be modified prior to each simulation session. In addition to specifying their budgets, advertisers can also be mapped into various categories. A proportion of advertisers within each category can have their budgets and bids on their ads perturbed. The proportion of advertisers so affected and the change in their budgets and bids are specified by the user via parameters to a normal distribution. Consequently, the number of advertisers chosen and the degree of actual change in bids and budgets vary from one simulation session to another. We plan to support dynamic agent-like behavior for advertisers in a future version.

5.2 Cassini Performance

Currently, Cassini runs on a single machine instance. A simulation consisting of approximately 80K unique queries occurring over an entire day takes a couple of hours to complete in Cassini. As Cassini loads most advertiser and query related information into memory, it remains memory-bound in terms of capacity. Disk I/O is our current bottleneck as the log information generated by Cassini for each query event can be quite large. For our next step, we are investigating parallelizing Cassini over a machine cluster.

6. Simulation Methodology

In this section we discuss a methodology for evaluating the performance of a set of design choices. Specifically, the core inputs required by the simulator are:

- Query sequence – This is a sequence of sets of keywords that users type into a search box, each with a time stamp. We generate a sample search sequence from historical search traffic.
- Bid landscape – This is a mapping that associates search keywords with advertisements which have a bid on the keyword. The maximum bid for each advertisement is included in the data. The bid landscape comes from actual ads and bids submitted by advertisers.

Additional inputs may be required for certain scenarios. For instance, an advertiser budget file is required for scenarios involving budget management; parameter sets describing the designs under evaluation are required; advertiser reactions to the design choices are accepted as inputs.

In order to evaluate the design choices effectively, the following structure for simulation runs is typically executed:

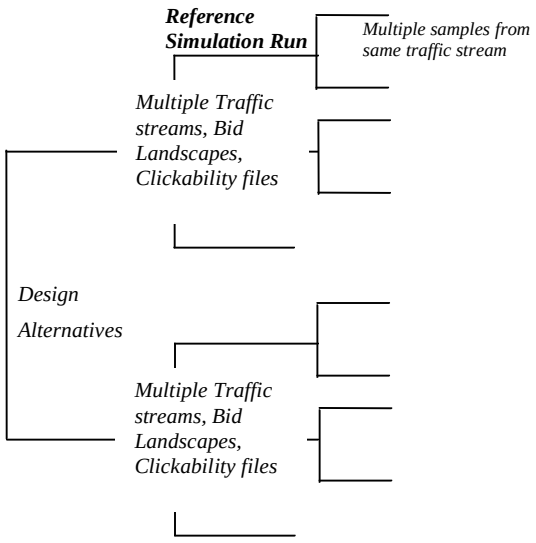


Figure 2: Structure of Simulation Experiment

The purpose of the reference simulation run is to validate the simulation set-up, and to serve as a baseline against which the performance of other design options can be compared. The reference run involves replicating the existing marketplace design parameters in the simulator for which we have historical data. This allows us to compare the outputs of the simulation against the actual historical output. The reference simulation run helps us validate:

- The representativeness of our data inputs in terms of coverage of queries and advertisers.
- Calibration of the user click model. The click model is calibrated by estimating the click probabilities associated with keyword – advertisement pairs from historical data.
- Calibration of the budget smoothing parameters. The budget smoothing parameters determine how forecasts of budget utilization in future time periods are generated. These forecasts are used by the budget smoothing algorithms.

We compare the results of the reference run to measurements from actual historical traffic for several days. Input data samples for simulation are drawn from the same historical traffic. We apply the t-test to compare simulation results to the actual historical results. Results are typically measured in terms of revenue per search, cost per click and overall click through rates (see Section 6.1). Other metrics are used depending on the design variables under evaluation. We adjust the simulation set-up until we have a validated simulation setup. Validation of the reference simulation run is important to ensure that the differences in the simulation output are due to the design parameters that are varied in the simulation experiment.

The reference simulation run is repeated for different samples of search traffic (over different days), bid landscapes, and click model. Test runs for specific design options and scenarios are run for the same combination of search traffic samples, bid landscapes and click models as the reference run.

6.1 Simulation Metrics

The simulation metrics of interest depend on the scenario, but the standard metrics we track include:

- Average revenue per search (RPS)
- Average cost per click (CPC)
- Average click through rate (CTR)
- Coverage – percentage of queries where ads are shown

Other metrics we have tracked include: percent of budget unspent, number of ads shutout (complete and partial).

6.2 Sampling Search Traffic

We have used two different sampling strategies in our experiments. The choice of sampling strategy is driven by the scenario under evaluation. The two sampling strategies are:

i) Stratified random sampling

ii) Micro-market sampling.

Stratified random sampling is used when the *independence of auctions* assumption is valid. This assumption states that each auction is independent – its outcomes are not influenced by the outcomes of other auctions. Thus, changing the sequence of searches will not affect the outcome. Of course, in a repeated auction setting this assumption can hold only in equilibrium. As stated before, we have assumed all agents in our simulation model are non-adaptive.

One scenario that does not meet the "independence of auctions" assumption involves budget management. Here it is assumed that advertisers have budgets, and the amount spent should be managed such that their budgets are not exceeded. Advertiser budgets introduce interactions between queries and between advertisers. The budget constraint introduces path dependence – the outcome of a series of auctions depends upon the sequence of keywords in which that advertiser participates. In such scenarios we use the "micro-market sampling" strategy.

6.2.1 Stratified Sampling

Stratified sampling – independent sampling from multiple tiers or strata is important because search queries and advertisers are heterogeneous. When there is a constraint on the sample size, stratified sampling reduces sample variance.

While queries match to advertisements and obtaining a representative mix of queries and advertisers is important, for simplicity, we either sample queries or advertisers. By properly constructing advertiser tiers we can be confident of getting a representative sample of the marketplace. For instance, create tiers of advertisers on the following dimensions:

- Number of bidded keywords. Though not all keywords are equal, we distinguish keywords based on how frequently they appear in historical search traffic. We classify keywords based on their location in the head, middle or tail portions of the frequency distribution. We characterize advertisers in terms the number of bidded keywords that fall into each portion. Thus an advertiser may be described in this dimension as large head, large middle, and small tail.
- Ad Quality – The average ad quality of all the advertisements for an advertiser is classified as high, medium or low.
- Bids – Each advertiser is classified as a high, medium or low bidder.

Other attributes may be used when relevant – for instance, the mix of budgeted versus unbudgeted advertisers competing on the query. This would be employed if budget management is activated in the simulation. .

The tiering approach can also be extended to other facets of the marketplace. In particular, we use this technique to scale the simulation results – as discussed in section 6.2.3. For this, it is important that the assumption of fungibility – that all advertisers within a tier are equivalent – hold. Our strategy for determining tiers generally allows us to make this assumption – this is discussed in 6.2.1.1.

After we have determined the tiers, we usually obtain a proportional number of samples from each tier. However, independent sampling from each tier to achieve a certain minimum sample size within each tier (to limit the expected variation within specified bounds) generally yields tighter range of results, although the sample sizes tend to be larger and the simulation takes longer to run.

To determine the minimum sample size within each tier, the calculation is based on the click probability distribution. We can use the mean click probability of each tier to estimate a sample size to achieve a desired bound on standard error.

6.2.1.1 Tier determination

Our procedure for determining the bins (tiers) into which we place queries or advertisers is based on simultaneously minimizing within-tier variance and maximizing cross-tier variance. This procedure is applied when we want to transform a continuous variable such as ad quality into a categorical one with values high and low. We use the Fisher ratio as our objective function along with constraints for minimum number of queries and revenue in each tier. The Fisher ratio for a set of tiers is the cross-tier variance of means divided by mean of within-tier variance. The higher the Fisher ratio, the more similar are the elements within a tier and dissimilar across tiers. In the case of advertiser tiers, we have a 3-dimensional vector of (depth, average ad quality and average bid) characterizing each query. We take the dot product of two vectors as the distance between them and compute means and variances on the dot product to determine the optimal tiers.

6.2.2 Micro-market Sampling

What do we mean by a micro-market? A micro-market is a collection of associated advertisers and keywords (advertisers bidding on the queries) such that:

$$\sum A_i = \sum Q_j$$

Where A_i is the total amount spent by advertiser i, and Q_j is the total revenue from query j. The total market is all traffic in a period P, such that P is the period over which budgets are replenished.

Markets that are large enough and yet complete are typically hard to find – some advertisers bid on a large number of keywords. To generate a micro-market we used an algorithm developed by Kevin Lang et al [7]. This algorithm solves the so called "small boundary dense subgraph" problem. The input to the algorithm is a bi-partite graph, with one set of nodes representing advertisers (or more accurately ads) and another set of nodes representing queries. Links between nodes in the two sets represent a "match" relationship, i.e. a search involving a query *q* would display all ads to which it is linked. However, to derive a complete market of the requisite size requires us to estimate the probability distribution of clicks on each link. This problem is inefficient to solve considering the large numbers of queries and advertisers. Using a graph where the link represent clicks aggregated over a period of time converts the problem to a deterministic one.

Essentially we are looking for a sub-graph such that:

- It contains an adequate number of nodes (queries + advertisers). A small number of nodes can lead to poor coverage in terms of the different tiers of advertisers and queries, too large a number leads to huge simulation run times. Typically, we have used between 100,000 to 150,000 nodes in our simulations.
- There is a large amount of spend on the edges within the sub-graph.
- There is a small amount of spend on the edges connecting nodes within the sub-graph to the rest of the nodes (relative to the amount of spend on the edges within the sub-graph).

There are parameters that control the relative importance of these different concerns. The algorithm produces solutions for each set of parameter settings.

A micro-market as we have defined it is typically not representative; i.e., the distribution of tiers of advertisers or queries in the sample may not be proportional to the distributions in historical traffic. We will have to scale the results to obtain results that pertain to an actual day's traffic.

6.2.3 Scaling the results

For scaling we use the stratification scheme we described in section 6.2.1. We can use the stratification scheme for advertisers or for queries depending upon the metrics we are interested in evaluating.

Scaling is based on the idea that the queries or advertisers within each tier are fungible. In scaling, the objective is to extrapolate results obtained for the sample to an actual day's traffic.

The scaling procedure we employ is simple:

We scale revenue and clicks in each tier as follows:

$r^{scaled}(i) = r^{sim}(i)*n(i)/n^{samp}(i)$

where $r^{sim}(i)$ is the spend in tier i from the simulation, n(i) is the number of queries (or advertisers) in the traffic to which we are trying to scale, and $n^{samp}(i)$ is the number of queries (or advertisers) in the sample used as input for the simulation.

Similarly, for clicks:

$c^{scaled}(i) = c^{sim}(i)*n(i)/n^{samp}(i)$

We then add up the revenue and number of clicks across the tiers and obtain the total number of searches and the coverage from the actual traffic.

7. Experiences

We have used the Cassini simulation system and applied the simulation methodology described to explore ideas in diverse aspects of marketplace design at Yahoo. Overall, our experience has been very positive. While we do not rely on the simulation system to produce absolute results, we have found it to be very useful in *comparing* one design against another, and understanding the *relative* performance of the designs.

Some of our experiments include:

- Ranking and Pricing – We tested different ranking methods, including ranking by bid only, ranking by bid and ad quality, and interesting hybrid combinations where the two types of ranking are combined. We also compared the output of the simulation system against live testing, and found that the simulation results tracked the live results well directionally. We also tested different pricing schemes, such as first price auction, second price auction, and others.
- Budgeting – We experimented with different budget management schemes, including one where an advertiser's ads are shown for every matching query, until the budget is consumed, vs. another one where the frequency of the ads being displayed is gradually lowered until the budget is consumed.
- Advertiser Participation Scenarios – We wanted to quantify the value of adding one new advertisement into a sponsored search marketplace. Different scenarios assuming different number of advertisement of different quality (how relevant it is to the user query) were run. These scenarios allow us to quantitatively understand the tradeoff between increased CPC because of the increased number of advertisers, vs. the potential drop in CTR because of the addition of less relevant advertisements. This kind of understanding would be impossible or very difficult to obtain through live testing.

Below, we describe one set of experiments – Query to Ad Matching – in more details.

7.1 Query to Ad Matching Simulations

When advertisers bids for placement in the sponsored search marketplace, they specify the exact keywords (e.g., "nike shoes") to which the ads should match. If a user query matches the keywords exactly, then the advertiser's ads are eligible to be displayed to the user. There are many cases where the user query may not exactly match the specified keywords, but the ads may still be relevant to that user query (e.g., "air jordan sneakers"). Query to Ad Matching, or simply Matching, techniques allow the marketplace operator to "broaden" the user query, matching relevant ads that are not exact matches to the original user query.

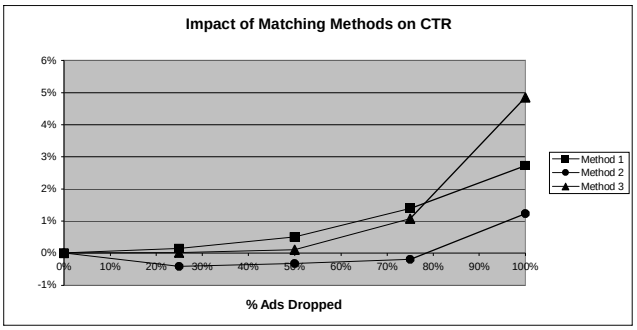
In this simulation, we compare three Matching methods. Details of these methods are outside the scope of this paper. The methods differ in how "aggressively" they match a user query to ads. A more aggressive method may be less precise, but may recall more ads that are potentially relevant. In the results below, we denote Method 1 as the least aggressive expansion method, and Method 3 as the most aggressive expansion method.

In the simulation system, in the Matching step, we tested each expansion method, one at a time. All other parts of the system are held constant. For a fixed set of searches randomly sampled from real search traffic, we simulated each search against the simulation system, ads that are matched by all three methods, together with those that exactly match the user query, are ranked and priced, and user clicks are simulated against the final ranked list of advertisements.

We measured the performance of the methods using RPS, CTR, CPC and Coverage metrics. The results from the simulations are consistent with the expected performance of the different query rewrite methods, in terms of both the directional impact on the metrics, and the relative performance of the methods.

As an example, in one simulation we decrease the percentage of results included from the methods being tested. For example, suppose Method 1 returned a total of 10 advertisements for a given user query. At 10%, we would drop 1 ad (selected at random) matched by Method 1 from the final set of advertisements for that query.

The chart below shows the results. As we increase the percentage of results dropped from 0% to 100%, we notice that there is a generally positive effect on CTR. This is expected because when the user query is being broadened, we may introduce ads that are less relevant to the user's query. The more aggressive methods have larger impact on CTR; the larger magnitude of the increase CTR as more ads are dropped for Method 3 is also consistent with our expectation.



8. Conclusion

Cassini is a simulation framework intended for fast, iterative evaluation. We present our approach to best approximate the marketplace technologies, user and advertiser behavior. In addition, we describe how to effectively sample the marketplace and re-scale the simulation results back to the overall marketplace.

Our experience using Cassini to evaluate marketplace designs has been very positive. Future work includes moving to a parallel processing architecture that would allow us to simulate the entire marketplace instead of a portion thereof. Given growing interest in Cassini from groups at Yahoo outside sponsored search, we are exploring whether the Cassini methodology can be extended to other products. Finally, we plan to leverage existing research [4, 10] on modeling dynamic agent-like behavior for advertisers by incorporating support for this functionality in a future version.

9. REFERENCES

[1] Anthony, P., Jennings N., "Developing A Bidding Agent for Multiple Heterogeneous Auctions," *ACM Transactions on Internet Technology*, Vol 3, Issue 3, August 2003, pp 185-217.

[2] Bapna, R., Goes, P., and Gupta, A., "Replicating Online Yahoo Auctions to Analyze Auctioneers' and Bidders Strategies," *Information Systems Research*, Vol. 14, No. 3, September 2003, pp. 244-268.

[3] Csirik, J., Littman, M., Singh, S., and Stone, P., "FAucS: An FCC Spectrum Auction Simulator for Autonomous Bidding Agents," *Second International Workshop on Electronic Commerce*, November 2001.

[4] Edelman, B., Ostrovsky, M., and Schwarz, M., "Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords." *American Economic Review*, forthcoming.

[5] Feng, J., Bhargava, H. K., and Pennock, D. M., "Implementing Sponsored Search in Web Search Engines: Computational Evaluation of Alternative Mechanisms," *Informs Journal on Computing*, Forthcoming.

[6] Kitts, B, LeBlanc, B. J., "A Trading Agent and Simulator for Keyword Auctions," *Proceedings of the Third International Joint Conference on Autonomous Agents & Multi Agent Systems*, New York City, July, 2004.

[7] Lang, Kevin J., and Andersen, Reid, "Solving the Small Boundary Dense Subgraph problem," Yahoo! Research working paper, 9/2006.

[8] Powell, Michael, "An Experimental Study of Adaptive and Non Adaptive Agents in a Simulated Double Auction Market," Technical Report, University of East Anglia, 2003/2004.

[9] Wurman, P., Wellman, M., and Walsh, W., "The Michigan Internet AuctionBot: a Configurable Auction Server for Human and Software Agents," *Proceedings of the Second International Conference on Autonomous Agents*, 1998, pp 301-308.

[10] Zhang, X., and Feng, J., "Price Cycles in Online Advertising Auctions," *Proceedings of the 26th International Conference on Information Systems (ICIS)*, Dec. 2005, Las Vegas, NV.