Online Experimentation: Principles and Practice

Roger Longbotham, Ji Chen, Justin Wang
Experimentation Platform http://exp-platform.com
Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

Abstract
Websites and Web services can benefit greatly from well designed and analyzed experiments. Companies such as Amazon, Google, Yahoo, Microsoft and others regularly test changes to their sites prior to release. They have found that a key to successful innovation is having a software platform for conducting experiments. Even though Design of Experiments theory is fully applicable, there are many ways in which online experiments differ from other experiments. This paper will discuss design principles, pitfalls and other lessons learned in this unique software-intensive environment. We will also cover challenges such as highly skewed distributions, effect of webcrawlers, data quality and large sample sizes. We will close with several research questions.

Key Words: design of experiments, online, software, web sites, DOE

1. Introduction

The main purpose of this paper is to highlight the ways in which online experimentation differs from most other statistically designed experiments. In addition we will make the case that experimentation is a natural fit in the online realm. Finally, since this is a fairly new area for design of experiments (DOE), we will give some open questions that could be areas of future research.
Outline of paper:

- The case for Online Experimentation
- Design Principles
- Lessons Learned
- Research Topics

1.1 The Case for Online Experimentation

Running randomized, controlled experiments is a natural fit for websites and web services. In contrast to the corner bookstore or grocery store, an online business does not know their customers personally and hence cannot make changes based on an intimate knowledge of their likes, dislikes and future needs. Indeed, most websites know little or nothing about the visitors to the site. A rare exception would be a site that requires users to log in, in which case they may have information from their sign-up form and subsequent behavior. Therefore, most sites depend on the behavior of the visitors to the site to see if their needs are being met. The positive side of this is that most sites get many visitors so we are dealing with large sample sizes. Another argument for online experimentation is that the field of web design is fairly young and evolving rapidly due to the rapid development of software capabilities so a structured approach to learning what works best is a big asset. Thirdly, computers are a necessary part of the online service and

can integrate data collection and analysis easily into the business process. Finally, changes to websites and services can be conceived and tested fairly quickly giving the decision-making process much needed information.

1.2 Opportunities for Experimentation

There are many areas in which experimentation can help an online business be more successful.

- User Interface (UI) – i.e. how the content of the site is arranged on a page or among different pages of the site. This is the most obvious use of experimentation and most online experiments deal with UI changes. This could be simple changes in color or font or could involve moving content on a page or to other pages as well as changes in navigation of the site.
- Performance improvements – i.e. page load time. This is an often-overlooked aspect of website improvement, but many experiments have shown that a fractional second improvement in how long it takes for a page to load can have a significant impact on user behavior. (Brutlag, 2009) (Schurman & Brutlag, 2009)
- Backend algorithms. Examples of algorithms that can be optimized through experimentation include decisions
    - To determine content (e.g. headline optimization – which headlines to show and ordering of headlines)
    - Based on user input (e.g. search relevance results)
    - Based on user behavior (e.g. customers who viewed this…)
    - Based on user characteristics (e.g. returning vs. new customer)

  Sites such as Amazon.com, Bing.com and Google.com (Tang, 2010) experiment on algorithms continually to test new algorithms and refine parameters of existing algorithms.
- Synergies with other sites or services. Examples include
    - Email or referring site(s). In other words, marketing efforts that send traffic to your site.
    - Cross-site experiments. A cross-site experiment is one in which the experiment takes place on two or more sites.

2. Design Principles for Online Experimentation

The ways DOE differs for the online space are due to the characteristics of the data and the environment.

2.1 Characteristics of Online Experimentation

Online experiments differ from the typical offline experiment in the following ways:

- Huge amounts of data (our platform averages 5-10 million visitors per experiment). Some experiments may have as few as 10,000 users and others close to 100 million.
- Know little or nothing about individual visitors.
- Can involve all users. Most online experiments can randomize all visitors to the site into the experiment which helps avoid selection biases. Offline experiments often need to be limited in scope to limit the expense of running the experiment, but, due to the automation of online experiments this is not necessary.

- Technical challenges caused by robots, software glitches and complexities. Problems we run into with online experiments that are not typical in offline experiments are often due to webcrawlers or (software) robots and idiosyncrasies of the complex software environment. (Crook, Frasca, Kohavi, & Longbotham, 2009)
- Highly skewed data. The typical metric for an online experiment is highly skewed and also includes outliers due to real users in addition to robots.
- Very dynamic (nonstationary) environment with pronounced cycles and trends. The online environment has pronounced daily, weekly and longer term cycles as well as long-term trends due to an increasing population of online users as well as changes in the ways people get online (e.g. mobile devices).

2.2 Design Principles for Online Experimentation

The design principles for online experiments differ from the typical offline experiment because of the characteristics mentioned in section 2.1 and the software enablers and limitations.

2.1.1 Run all treatment combinations concurrently

The first principle is that all treatment combinations (TCs) should be run concurrently. Many offline experiments will run one TC at a time until the all TCs are completed. However, due to the severe non-stationarity seen in online data, the change in the metric averages from one time period to another would overwhelm most treatment effects. Any online experiment where the TCs are not run at the same time is likely to be misleading. The software nature of online experiments makes this an easy thing to do. Simply randomly assign each user to a TC and show the variant of the site corresponding to that TC to the user.

2.1.2 Randomize by visitor

Generally, online experiments are run where a visitor is randomly given a treatment assignment and, as much as possible, that visitor is kept in the same treatment group throughout the experiment. For many experiments it would be a bad customer experience to be shown a different set of TCs each time the user returns to the site, especially if it is a large, noticeable change. Other ways to randomize could be by page view, by session, by product code or other, but these are almost always seen as inferior to randomization by visitor and used when it is infeasible to randomize by visitor.

2.1.3 Each visitor needs a user ID stored in a cookie

In order to randomize by user and have the user continue to see the same variant throughout the experiment an identifier for the user needs to persist across time. This is normally done by storing a user id (UID) in a cookie for the user. By using a hashing algorithm with the UID as the seed for randomization, the user will continue to get the same variant. Another method is to cache the treatment assignment or use a look-up table for the UID. One benefit of randomizing users based on a UID stored in a cookie is keeping users in the same TC for the duration of the experiment. Another benefit is the ability to calculate metrics on a per user basis. In other words, we could calculate clicks per user, page views per user, purchases per user, etc. for the duration of the experiment. One downside to this approach is it depends on cookies. Some users don't allow cookies to be set, others clear their cookies frequently and still others use multiple browsers or computers, each with a different cookie. We do not see this as a large negative, but one that the experimenter should be aware of. Some have expressed concern about the use of

cookies for privacy reason. Online experimentation can be very effectively performed without the use of any personally identifiable information (PII) and, by policy, our platform does not store or use PII data.

2.1.4 Each experiment has all combinations of factors

Generally, in traditional DOE, a significant amount of time and expertise is needed to determine the best statistical design, which is usually some subset of all combinations of the factor levels being tested. Some online experiments are run on a subset of all combination of factors but that is not necessary. It is no more difficult or expensive to set up a full factorial experiment online for any number of factors and levels.

2.1.5 Each factor is set up as an independently randomized experiment

This principle is perhaps not as clear-cut or universal as the first three principles, but we believe this is an advantageous way to run a multi-factor experiment. Our software platform for experimentation sets up each factor as an independently randomized experiment. Each of these experiments should be run on the same set of users for the same time period. Then, each user will be randomly assigned to a level for each factor and in the analysis any interaction among the factors can be calculated. Additionally, if one of the experiments is doing poorly due to a bug or poor user acceptance, it could be turned off easily without interfering with the other factors.

3. Lessons Learned from Online Experimentation

From past experience in online experimentation, we present several lessons learned, quite a number of which are unique to the online space.

3.1 Identify robots/webcrawlers and remove them from analysis

A unique concern in conducting experiments online is the existence of robots. Robots (also called webcrawlers or spiders) are software programs that traverse many pages in the web automatically. There are many types of robots, some only generate pageviews, and some also generate clicks on links. Special types of robots are used by search engines as a means of providing up-to-date data (Kobayashi & Takeda, 2000). Robots can be benign or malicious, and multiple types of malware also technically can be referred to as robots. Internet sites can get a significant amount of robot traffic (search engine crawlers, email harvesters, botnets, etc.). Robots may provide half of the traffic for some websites.

Robots are not welcome in an experiment because they can generate tens of thousands of pageviews and/or clicks that far exceed human limits and thus can significantly skew the results. Figure 3.1 gives an example of an AA experiment, i.e. there is no difference between Treatment and Control (Peterson E. , 2004). We would expect them both to have the same daily pattern; however, the graph shows that there are many hours where Treatment and Control deviate by a large amount. Further investigation reveals that all of the large deviations were caused by robots. Although each hour has thousands of users, each of these deviations was caused by a single user (robot) with a very large number of clicks. The top “user” had about 7,000 clicks in an hour.

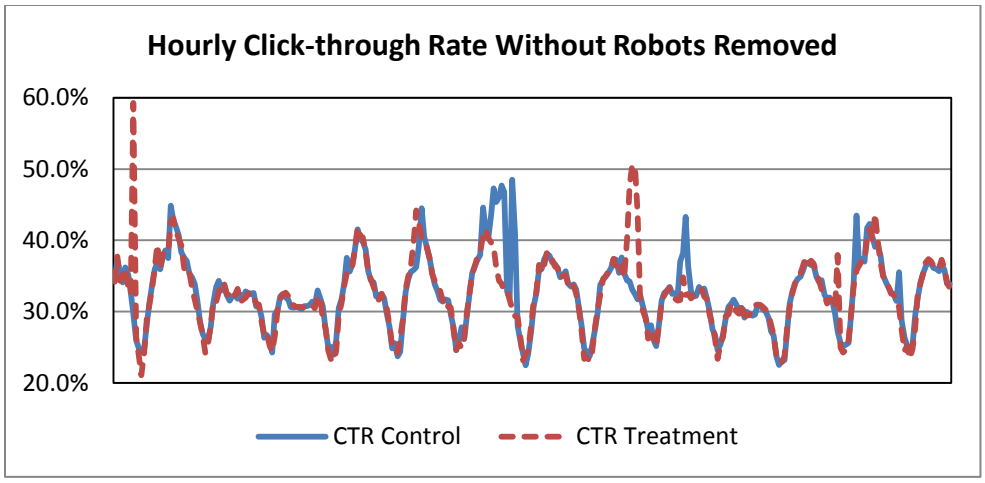

Figure 3.1: Hourly click-through rate plot without robot removal for an AA experiment, each cycle being one day

The best way to identify robots remains an open question. Benign or simple robots can often be filtered by basic characteristics such as IP address and user agent. Some robots can be identified using behavioral heuristics (Tan & Kumar, 2002). Examples of the heuristics include number of pageviews/clicks in a certain period of time (e.g., more than 100 clicks in an hour), regular pattern for the actions on site (e.g. click every 10 minutes), etc. More sophisticated robots even mimic human behavior and execute Javascript, and hence are difficult, if not impossible to identify except as outliers. The goal of our robot removal process for experimentation is to remove robots that might skew the experiment results, so the focus is on robots with many actions on the site.

3.2 Run for integer number of weeks; Estimate novelty/primacy effects

In the planning process of online experimentation, one necessary step is to determine how long the experiment should run. A power calculation can tell you roughly the minimum sample size (e.g. number of users, page views) needed which can be translated into how long to run the experiment (Kohavi, Longbothm, Sommerfield, & Henne, 2009). Even when the traffic volume exceeds the minimum sample size, typically the shortest time period for running any experiment is at least one full day. The reason for this is that users behave differently at different times of the day, so the effect of Treatment over Control might vary in a 24 hour period, and thus the results from a partial day might not be representative. In fact, user behavior might also vary between weekdays and weekends and having a full week cycle would allow us to estimate the overall effect as well as gain some insights into the day-of-week effect. Figure 3.2 is a typical graph of traffic for a site in a period of 9 days. Control and Treatment clicks were pretty close to each other, so they overlapped at most times. Both of them had lower traffic on weekends (Saturday and Sunday) and higher traffic during weekdays. Therefore, for most experiments we recommend running for multiples of a week. Actually the best practice is to run the experiment for at least two weeks in order to check the weekday /weekend effect and allow minimal day-exclusions if there are data anomalies.

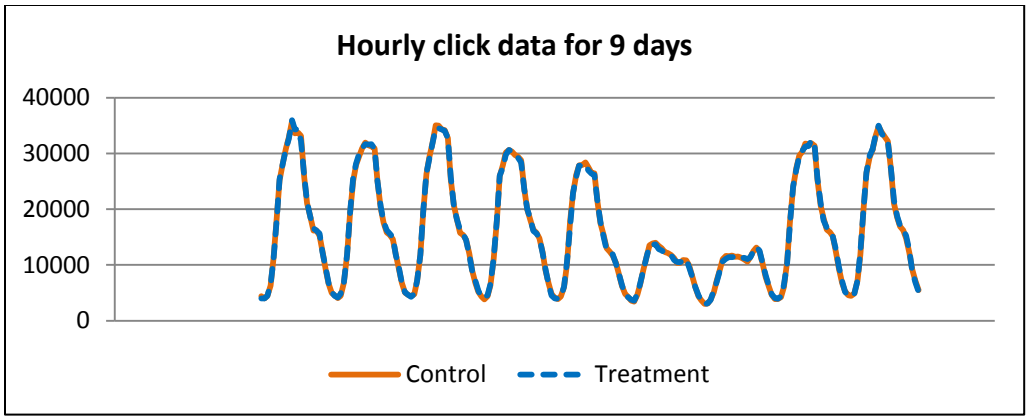

Figure 3.2: Hourly Traffic for a site for 9 days

For some experiments, there could be concerns about possible primacy or novelty effects. A novelty effect occurs when the initial effect of the Treatment is not the same as the long term effect. This could be an initial negative effect that becomes more positive as users get used to it or it could be a Treatment that causes much interest at first that falls off. In order to estimate a novelty effect we recommend you run your experiment for at least 4 weeks. Figure 3.3 is an example of novelty effect for an experiment. The treatment effect started at 5% and decreased over the first two weeks. The asymptotic effect was about 2.6%.

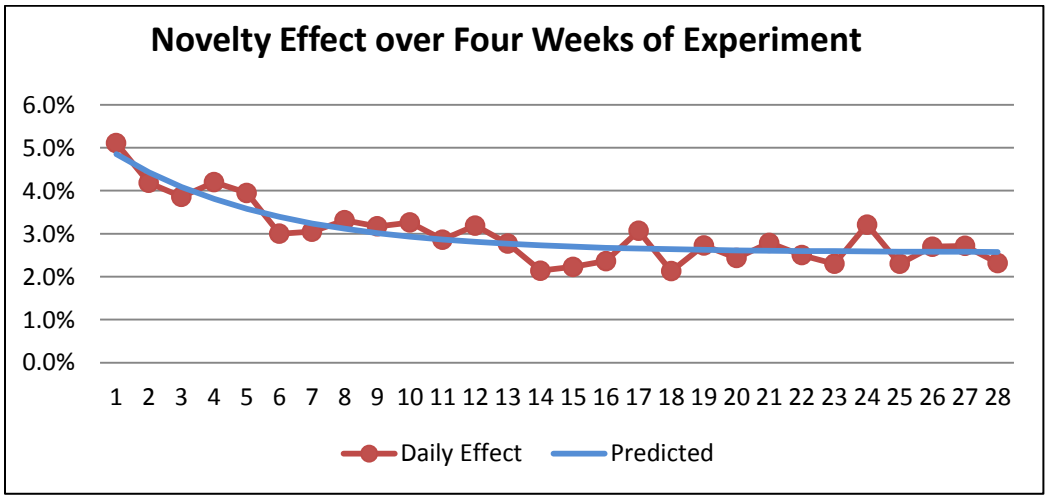

Figure 3.3: Treatment effect by day for an experiment. The predicted line is fitted as: $Effect = .026 + .028 * \exp(-.2 * t), t = 1, 2, \ldots 28$

Another scenario where we recommend running an experiment for 4 weeks or longer is when you need to estimate a long term effect such as user loyalty or return rate (Peterson E. , 2005). If there is a novelty effect you should make sure you have enough data for the level of sensitivity required after the time period with the novelty effect is removed from the analysis, so the real long term effects will be estimated.

On the other hand, very long experiments have many problems. It is all too easy for teams to introduce changes only to Control or Treatment during the long period, and thus

long experiments are hard to keep control. In addition, cookie churn becomes more and more of an issue as time goes, and users that erase cookies nightly start to appear as a more significant percent of users and skew metrics. Therefore, we would recommend most experiments to run for less than six weeks.

3.3 Standard deviation calculations

One of the common KPI metrics for a web page is clicks per session (or visit), i.e. the ratio of total number of clicks and total number of sessions. One user can visit the website multiple times and have more than one session. The number of clicks and page views of the sessions generated by the same user are not independent. For the same reason, this issue occurs in by user-day metrics and click-through rate (CTR), defined as the total number of clicks divided by the total number of page views. Ignoring the independence would underestimate the standard deviation of the metric, tend to reject the null hypothesis and, therefore, incorrectly conclude too many experiments being statistically significantly different. We suggest using a bootstrapping algorithm (Rao, 1988) or delta method (Oehlert, 1992) to estimate the standard deviation of the metric. Bootstrapping algorithms simulate the distribution of the metric by re-sampling. It is computationally intensive for large datasets. On the other hand, the delta method uses Taylor series approximation to estimate the variance and works well for large datasets since the higher-order terms converge to zero quickly.

3.4 Data/experiment integrity

The online environment is a complex integration of many different configuration settings distributed across multiple systems. It is important to verify the experiment is implemented as designed. We recommend the following data validation checks before data analysis.

- Percent of users in each variant. For a simple A/B test, we usually assign 50% of users to the Control and 50% to the Treatment. On many occasions we have seen the percent of users in each variant is not as expected. We have found several reasons to cause the imbalance: 1) certain set of users always assigned to Control. For example, all MSN cobranded users have to see Control by contract. In this case, the cobranded users should be excluded from the analysis of the experiment. 2) cookie-related bugs. User IDs are typically stored in browser cookies. If one of the variants updates the cookie more frequently (either by the users or by the system), there will be a consistent bias. 3) hardware or software implementation. More details of this issue will be discussed in the next section.
- Large change in treatment effect. An unexpectedly large change in treatment effect might suggest an implementation issue. In one experiment we found users in one Treatment saw another Treatment for a short period of time due to an error. In another experiment, the purpose was to test the module layout. However, during the experiment, not only the module layouts were different (controlled difference) the module contents were also different for some period of time (uncontrolled difference). For this period of time, the treatment effect was much larger than the rest of the time due to this uncontrolled difference. Also, large change of treatment effects due to special events might not be representative to the website under usual conditions. Those non-representative periods of time should be removed from the data analysis.

3.5 Differences due to hardware or software implementation

Several implementation issues can make the experiment not a true controlled experiment. For example, different loading speed, capacity, logging/tracking mechanism and caching of the page might cause a noticeable bias. Redirection of one variant but not another will cause the redirected variant to lose more events and users due to the loading delay. Tracking the page using JavaScript and image beacons will have different data loss rate and internet robots might respond to those two tracking mechanisms differently. Caching the page differently (higher cache-hit-rate usually improves user experience by loading the page faster) introduces another difference between Control and Treatment besides the intended test factor.

3.6 Distributional Challenges

Internet data are usually highly skewed. Figure 3.4 shows the distribution of clicks per user for the users who clicked at least once. The distribution is highly skewed with a long tail. Some distributions have the same shape plus 95% zeros. The skewness of the distributions leads to lower power for the usual two sample t-tests compared to alternatives. Transformation (or truncation) and using nonparametric approach might improve the power of the tests. Distribution of ratios is also challenging. For example, the distribution of CTR is very hard to formulate even though the standard deviation can be estimated by using bootstrapping algorithm or delta method as suggested earlier.

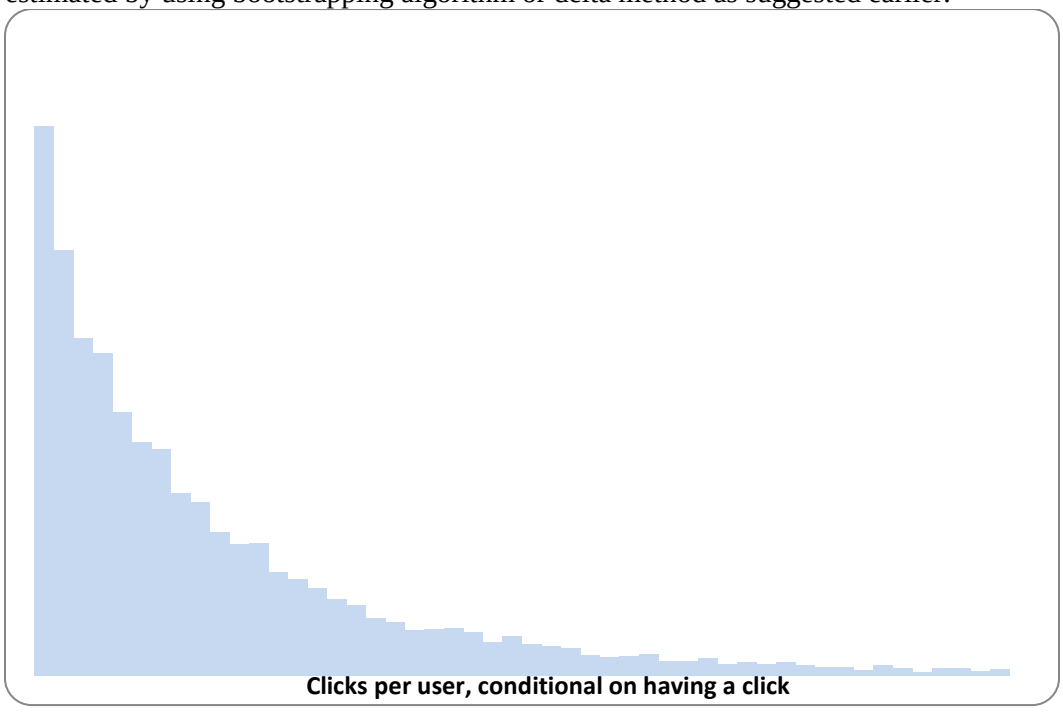

Figure 3.4: Clicks per user, conditional on having a click

4. Research Topics

Some questions in the online experimentation area still remain open and these serve as interesting research topics:

4.1 Automated tests for robots

Section 3.1 introduces some heuristics to identify robots including unreasonable high traffic volume (page views or clicks) and regular pattern for the actions on site. Much more research needs to be done to define additional heuristics and refine the parameters of the existing heuristics. For example, the parameters of the heuristics should depend on the type and the size of the website but no research has been done to automatically adjust the parameters based on site characteristics. Machine learning algorithms can be applied to the empirical data to determine those parameters.

4.2 Data validation check

The online environment is a complex integration of many configuration settings distributed across multiple systems. It is not uncommon to overlook some differences between the variants besides the factor(s) the experiment intended to test. More ways to validate experimental data from multiple dimensions is critical for valid analyses and challenging to implement.

4.3 Automatic detection of subgroups with different treatment effect

Identifying subgroups, e.g. users visiting the website at certain time of the day from a specific browser, with different treatment effect can be helpful to highlight the opportunities for personalization or potential problems. It also enhances the understanding of user behavior. Given the large sample sizes and the many possible dimensions for segmenting the data, it is important to have an automatic methodology that could run in the background, account for multiple testing error and alert if a subset of users behaves differently to the treatment than everyone else.

4.4 How to optimally use online surveys as response

Survey data or customer responses are valuable source of information to improve the quality of the services. However, online survey provides users the options to choose whether to participate and the response rate is usually very low. The self-selected nature of the responses and small sample size make it difficult to derive statistical conclusions due to bias and low power. It is important to develop online survey methods that give the site the needed information while maximizing response rate and minimizing bias for online experiments.

4.5 Novelty Effect

As mentioned above, typically we recommend running an experiment for 4 weeks or longer if novelty effects are considered a possibility. Some research questions:

1. If we are not sure whether there will be such effects, how do we detect potential novelty effect quickly so we can decide whether to extend an experiment for 4 weeks or longer?
2. How is our estimate affected by weekend effect and other instabilities?
3. What is the best way to estimate the asymptote? Should we use daily summaries for metric values or a continuous approach?
4. Should only returning users be considered in estimating the trend?

4.6 Loss of UIDs

Loss of UIDs can be intentional by the site (due to site policy), intentional by the user (some users clear cookies) or "random" (accidental software deletion of cookies). Since UIDs are the unit of randomization in most experiments, how does this UID churn affect experimental results? Are there ways to adjust the treatment effect for this churn? What is the implication for long term experiments (6 weeks and longer)?

Acknowledgements

We are deeply grateful to Ronny Kohavi for getting us the involved with Microsoft's Experimentation Platform and the interesting problems in that space. We are also grateful for the many interactions with team members that have pushed us and helped refine our thinking.

References

Brutlag, J. (2009, June 22). *Speed Matters for Google Web Search.* Retrieved from code.google.com: http://code.google.com/speed/files/delayexp.pdf

Crook, T., Frasca, B., Kohavi, R., & Longbotham, R. (2009). Seven Pitfalls to Avoid when Running Controlled Experiments on the Web. *KDD2009* (pp. 1105-1114). http://portal.acm.org/citation.cfm?doid=1557019.1557139.

Kobayashi, M., & Takeda, K. (2000). Information retrieval on the web. *ACM Computing Surveys (ACM Press)*, 32 (2): 144-173.

Kohavi, R., Longbothm, R., Sommerfield, D., & Henne, R. (2009). Controlled experiments on the web: survey and practical guide. *Data Min Knowl Disc*, 140-181.

Oehlert, G. W. (1992). A Note on the Delta Method. *The American Statistician*, Vol. 46, No. 1, p. 27-29.

Peterson, E. (2004). *Web analytics demystified: a marketer's guide to understanding how your web site.* Celilo Group Media and CafePress, ISBN: 0974358428.

Peterson, E. (2005). *Web Site Measurement Hacks.* O'Reilly.

Rao, J. W. (1988). Resampling Inference With Complex Survey Data. *Journal of the American Statistical Association*, 231-241.

Schurman, E., & Brutlag, J. (2009, June 23). *The User and Business Impact of Server Delays, Additional Bytes, and HTTP Chunking in Web Search.* Retrieved from O'Reilly Velocity Web Performance and Operations Conference: http://en.oreilly.com/velocity2009/public/schedule/detail/8523

Tan, P.-N., & Kumar, V. (2002). Discovery of web robot sessions based on their navigational patterns. *Data Min Knowl Dis*, Volume 6, Number 1, 9-35.

Tang, D. A. (2010). Overlapping Experiment Infrastructure: More, Better, Faster Experimentation. *KDD'10*, (pp. 17-26). Washington, DC, USA.