From Infrastructure to Culture: A/B Testing Challenges in Large Scale Social Networks

Ya Xu, Nanyu Chen, Adrian Fernandez, Omar Sinno, Anmol Bhasin
LinkedIn Corp, 2029 Stierlin Court, Mountain View, CA 94043
{yaxu, nchen, afernandez, osinno, abhasin}@linkedin.com

ABSTRACT

A/B testing, also known as bucket testing, split testing, or controlled experiment, is a standard way to evaluate user engagement or satisfaction from a new service, feature, or product. It is widely used among online websites, including social network sites such as Facebook, LinkedIn, and Twitter to make data-driven decisions. At LinkedIn, we have seen tremendous growth of controlled experiments over time, with now over 400 concurrent experiments running per day. General A/B testing frameworks and methodologies, including challenges and pitfalls, have been discussed extensively in several previous KDD work [7, 8, 9, 10]. In this paper, we describe in depth the experimentation platform we have built at LinkedIn and the challenges that arise particularly when running A/B tests at large scale in a social network setting. We start with an introduction of the experimentation platform and how it is built to handle each step of the A/B testing process at LinkedIn, from designing and deploying experiments to analyzing them. It is then followed by discussions on several more sophisticated A/B testing scenarios, such as running offline experiments and addressing the network effect, where one user's action can influence that of another. Lastly, we talk about features and processes that are crucial for building a strong experimentation culture.

Categories and Subject Descriptors

G.3 Probability and Statistics/Experimental Design: controlled experiments, randomized experiments, A/B testing.

General Terms

Measurement, Design, Experimentation

Keywords

Controlled experiments, A/B testing, social network, online experiments, network A/B testing, measurement.

1 INTRODUCTION

A/B testing, also called controlled experiment, has become the gold standard for evaluating new product strategies and approaches in many internet companies, including Amazon, eBay, Etsy, Facebook, Google, Groupon, LinkedIn, Microsoft, Netflix and Yahoo [9, 10]. As experimentation gains popularity, so does the need for properly designing, managing and analyzing experiments.

KDD '15, August 11 - 14, 2015, Sydney, NSW, Australia

The theory of controlled experiment dates back to Sir Ronald A. Fisher's experiments at the Rothamsted Agricultural Experimental Station in England in the 1920s [11]. Since then, many textbooks and papers from different fields have provided theoretical foundations [20, 21, 32, 33] for running controlled experiments. While the theory may be straightforward, the deployment and mining of experiments in practice and at scale can be complex and challenging [13, 14]. In particular, several past KDD papers have discussed at length the experimentation systems used at Microsoft Bing and Google [8, 9], including best practices and pitfalls [7, 10]. Facebook also introduces the PlanOut language which provides a toolkit for parameter-based experiments [12].

At a high level, we follow similar practices and methodology for experimentation at LinkedIn. However, many of the challenges we face arise particularly because LinkedIn is a member-based social network (we call our logged-in users "members"). In this paper, we focus on how we address these challenges as we scale to run more experiments. We share how we built XLNT (pronounced "excellent"), the end-to-end A/B testing platform at LinkedIn, to not only meet the day-to-day A/B testing needs across the company, but to also address more sophisticated use cases that are prevalent in a social network setting.

When we launched XLNT, the platform only supported about 50 experiments per day. Today, that number has increased to more than 400. The number of metrics supported has grown from 60 to more than 1000. Such tremendous growth is not only attributed to our scalable platform but also to our continuous emphasis on embedding experimentation deeply into LinkedIn's decision-making process and culture. We include in the paper several XLNT features we built to enable us to take education and evangelization past the "classroom".

A/B testing is truly the driver behind LinkedIn's product innovation. The areas we experiment on are extremely diverse, ranging from visual changes on our home page, to improvements on our job recommendation algorithm, to personalizing the subject line of our emails. We begin this paper with two example experiments that we recently ran.

Figure 1: Guided edit experiment on profiles.

The first experiment was on members' profile pages (Figure 1). In order to encourage members to better establish their professional identity, we displayed a small module at the top of their profile. The experiment was to include an additional line of text to call out the benefits and values a complete profile provides. For instance, the example in Figure 1 encourages members to add volunteer

experience to their profile. This small change turned out to be extremely successful. Results from the A/B test showed a 14% increase in profile edits on volunteer experience!

Another experiment was an entire redesign of the Premium Subscription's payment flow (Figure 2). Apart from providing a cleaner look and feel, we reduced the number of payment checkout pages and added an FAQ. The experiment showed an increase of millions of dollars in annualized bookings, about 30% reduction in refund orders and over 10% lift in free trial orders.

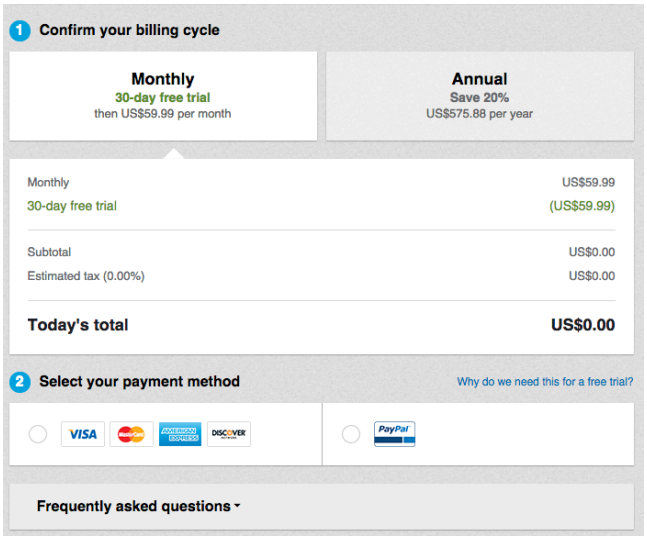

Figure 2: New Premium Subscription payment flow.

In both cases, the XLNT platform made it feasible to quickly measure the impact of both small and large changes at low cost. This enables us to identify features with a high Return-On-Investment (ROI) and moreover, to quantify the impact in a scientific and controlled manner. A screen shot of our analysis dashboard is shown in Figure 3.

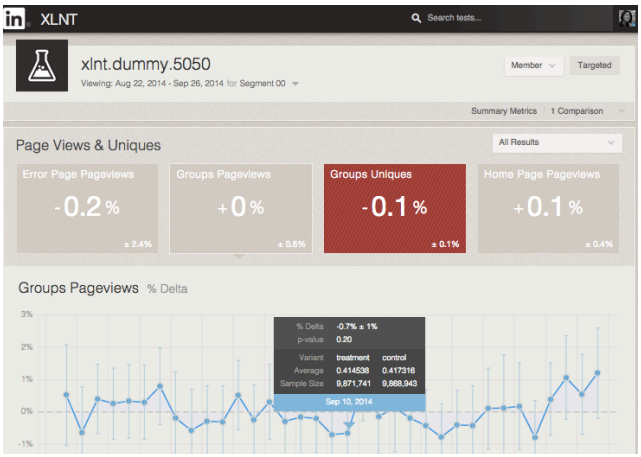

Figure 3: XLNT analysis dashboard.

Here is a summary of our contributions in this paper:

- We share the details of how we built our experimentation platform, including the engineering challenges we faced and how we addressed them.
- We discuss several challenging A/B testing scenarios we face at LinkedIn. Some of these challenges, particularly the ones that are specific to experimentation on social networks, are discussed and shared in a public paper for the first time.
- We discuss several concepts and novel features we introduced at LinkedIn that have significantly shaped our experimentation culture.
- Many real A/B test examples are shared for the first time in public. Even though the examples may be LinkedIn specific, most of the lessons and best practices we share are applicable to experimenting on social networks in general.

The paper is organized as follows. Section 2 introduces XLNT and how it is built to address several fundamental challenges. Section 3 discusses several more sophisticated A/B testing use cases. Section 4 discusses the XLNT features that help create a stronger experimentation culture and Section 5 concludes.

2 THE XLNT PLATFORM

We realized early on that ad-hoc A/B testing was not a scalable approach to sustain the high speed of innovation at LinkedIn. We required an A/B testing platform to allow us to quickly quantify the impact of features. Additionally, this needed to be achieved in a scientific and controlled manner across the company. Thus, we built XLNT.

The XLNT platform is aimed at encompassing every step of the testing process, from designing and deploying experiments to analyzing them. In particular, it was built to address the following concerns and challenges:

1. **Scalability.** We continue to see tremendous growth in both the number of concurrent experiments and the amount of data collected per experiment. The platform needs to scale to handle not only today's data volume but also tomorrow's.
2. **Incorporating Existing Practices**. Over time LinkedIn developed many A/B testing practices. For instance, we have a strong tradition on targeting (Section 2.1.2) as we believe each one of our members is special and unique. It is important to incorporate these practices as part of the platform.
3. **Engineering Integration**. The platform has to be well integrated into LinkedIn's engineering infrastructure. The experimentation platform architecture that works at other companies is unlikely to work for us due to different structure and tooling constraints.
4. **Flexibility**. Although the basic A/B testing requirements are similar across the organization, teams usually have their own special needs, given the diversity of the products they work on. The platform needs to offer enough flexibility to accommodate such customization.
5. **Usability.** A/B testing is not limited only to the R&D organizations. To make it truly a platform for everyone, we needed to provide an intuitive User Interface (together with APIs) for designing, deploying and analyzing experiments.

Taking these challenges into consideration, we share the details of the XLNT platform in this section, with the overall architecture outlined in Figure 4 below.

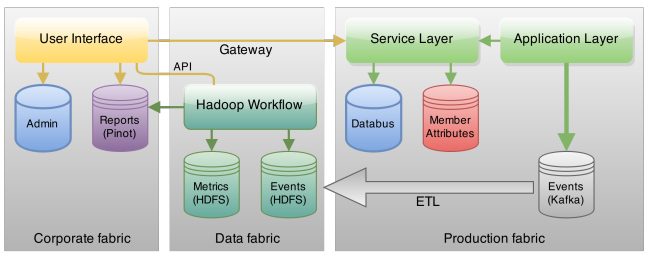

Figure 4: XLNT Platform overall architecture.

2.1 Design

Experimental design is arguably the most important step in the testing workflow to get good and meaningful results. As Sir R. A. Fisher put it [27] "To consult the statistician after an experiment is finished is often merely to ask him to conduct a post mortem examination. He can perhaps say what the experiment died of." To this end, we have built the platform to incorporate the standard practice at LinkedIn while providing capabilities to enable better designs and prevent common pitfalls. In this section, we first start with introducing a few key concepts that are fundamental to our experiment model and then focus on targeting, a critical component used in designing experiments at LinkedIn.

2.1.1 Experiment Definitions

Most experimentation terminologies used at LinkedIn are standard and can be found in any experimental design textbooks [28]. We focus here on only a few definitions that are key to our platform.

To run an experiment, one starts by creating a *testKey*, which is a unique identifier that represents the concept or the feature to be tested. An actual *experiment* is then created as an instantiation of the testKey. Such hierarchical structure makes it easy to manage experiments at various stages of the testing process. For example, we want to investigate the benefits of adding a background image. We begin by diverting only 1% of US users to the treatment, then increasing the allocation to 50% and eventually expanding to users outside of the US market. Even though the feature being tested remains the same throughout the ramping process, it requires different experiment instances as the traffic allocations and targeting changes. In other words, an experiment acts as a realization of the testKey, and only one experiment per testKey can be active at a time.

Every experiment is comprised of one or more *segments*, with each segment identifying a subpopulation to experiment on. A common practice is to set up an experiment with a "whitelist" segment containing only the team members developing the product, an "internal" segment consisting of all LinkedIn employees and additional segments targeting external users. Because each segment defines its own traffic allocation, the treatment can be ramped to 100% in the whitelist segment, while still running at 1% in the external segments. Note that segment ordering matters because members are only considered as part of the first eligible segment. After the experimenters input their design through an intuitive User Interface, all the information is then concisely stored in a DSL (Domain Specific Language). For example, the line below indicates a single segment experiment targeting English-speaking users in the US where 10% of them are in the treatment variant while the rest in control.

```
(ab (= (locale) "en_US")[treatment 10% control 90%])
```

It is important to mention that each experiment is associated with a *hashID*, which serves as an input to an MD5 based algorithm used to randomize users into variants. By default, all experiments of the same testKey share the same hashID, and different testKeys have different hashIDs. This ensures that a user receives consistent experience as we ramp up a treatment. More importantly, as we have hundreds of experiments running in parallel, different hashIDs imply that the randomizations between active experiments are orthogonal. The platform also allows manually overwriting the hashIDs, and the applicable usage cases will be discussed in Section 3.1.

2.1.2 Targeting

We recognize that not only are our products diverse, each one of our users is special and unique. With that in mind, many of the experiments we run at LinkedIn focus on how to provide the most improved user experience possible for specific user groups. This is achieved by creating different segments in an experiment targeting different subpopulations, as we have mentioned in Section 2.1.1. Deciding on the right population to target is the most important part of experiment design. There are three targeting capabilities provided by the platform:

Built-in Member Attributes. The platform provides more than 40 built-in member attributes for experimenters to leverage. They range from static attributes such as a member's country to dynamic attributes such as a member's last login date. These attributes are computed daily as part of our data pipelines and pushed to Voldemort, a distributed key-value data storage system [22], for real-time targeting.

Customized Member Attributes. Frequently experimenters need a targeting criterion beyond the default ones provided by XLNT. The platform provides a seamless onboarding process to include member attributes generated regularly from external data pipelines. It is even more straightforward if this is a static list generated from a one-off job, as one can simply "upload" it to the platform. These customized attributes are pushed to Voldemort on a daily basis and can be used the same way as any of the built-in ones.

Real-time Attributes. These attributes are only available at runtime, such as the browser type or mobile device. XLNT provides an integrated way to target using these attributes, or any parameters passed during a runtime request. For example, to target only requests coming from iPhones, one just needs to inform the platform that an attribute called "osName" is to be evaluated at runtime, and target only those with the value equal to "iPhone". This feature is used extensively for mobile experiments, as new mobile features are usually only rolled out for particular mobile app versions. This is also beneficial when experimenting on guest users where no information is available prior to the request. Section 3.2.1 includes more discussions on this case.

2.2 Deployment

The XLNT A/B testing platform is a key component of LinkedIn's Continuous Deployment framework [23]. It spans across every fabric and stack of LinkedIn's engineering infrastructure, providing A/B testing capabilities universally. Once the design is completed, deploying an experiment involves the following two components:

1. **Application Layer.** This includes any production artifacts, e.g. web applications and offline libraries. Each application requires a thick client dependency in order to run experiments locally, track experiment result events and interface with the service layer. The implementation in the application layer includes two parts: (1) making a simple one-line call to determine the variant, and (2) creating a code path to reflect the new variant behavior accordingly. For example, to decide the right color to show to a user in a "buttonColor" experiment, we just need to include the line below

   ```
   String color = client.getTreatment(memberID, "buttonColor").
   ```

   The second step is then simply changing the color of the button depending on the value of "color" returned above. This is the same across all application stacks including frontend, backend, mobile or even email experiments.

2. **Service Layer.** This is a distributed cache and experiment definition provider that implements *Rest.li* endpoints [29]. It is capable of executing experiments remotely and querying the built-in member attributes store described in Section 2.1.2. After the internal testing phase is passed, the experiment owner requests to activate the experiment. An SRE (Site Reliability Engineer) then reviews the specifications and, if no red flags are found, deploys the experiment to production. Experiment deployments are propagated via the Databus [24] relay and listeners. The new experiment definition is then distributed across LinkedIn's service stacks with updates sent to application clients every 5 minutes. This makes A/B testing totally independent of application code releases and can easily be managed through a centralized configuration UI.

At runtime, a simple experiment that does not involve targeting on pre-defined attributes can be executed locally at the application layer, which takes no time delay at all. Experiments that require member attributes for targeting (see Section 2.1.2) are sent to execute at the service layer. The results are then communicated back to the application client with a total delay of 1msec on average. Because these are high throughput services with about 20k to 30k QPS, we need to establish strict SLAs and enforce it with timeouts. These timeout durations can be fully customized according to experiment specific latencies.

2.2.1 Logging

To support monitoring and analysis, an event is logged during the "getTreatment" call at the application layer, with information such as the timestamp, testKey, experiment name, ID (experimental unit), variant, etc. These events are stored in Kafka topics [25] and periodically ETL'd to our HDFS clusters to be used in our data workflows.

It is important to note that these experiment events are fired only when the "getTreatment" code is called, and not for every request to LinkedIn.com. This not only reduces the logs footprint, but also enables us to do *triggered analysis,* where only users who were actually impacted by the experiment are included in the A/B test analysis. For example, LinkedIn.com could have 20 million daily users, but only 2 million of them visited the "jobs" page where the experiment is actually on. Without such trigger information, it is hard to isolate the real impact of the experiment from the noise, especially for experiments with low trigger rates.

Even with triggered logging, the volume of events generated presents a challenge. Currently, an average of 10 billion events are generated daily, and that number is growing quickly as more experiments are run on an increasing user base. We have visited the event schema and generation conditions several times in the past to remove derivable attributes and encode values. Continued effort is necessary for incremental improvement and to identifying new solutions addressing this challenge going forward.

2.3 Offline Analysis

Automated analytics is crucial in popularizing experimentation. It not only saves teams from time-consuming ad hoc analysis, but also ensures that the methodology behind the reports is solid, consistent and scientifically founded.

To paint with a broad brush, the analytics pipeline computes user engagement metrics such as pageviews and clicks, and joins them with the experiment assignment information from online logs described in Section 2.2.1. The data are then aggregated based on the experiment and time range to produce summary statistics that are sufficient to compute not only the difference between any two variants, but also the statistical significance information such as p-values and confidence intervals.

Approximately 4TB of metrics data and 6TB of experiment assignment data are processed every day to produce over 150 million summary records. Much of this computation utilizes the large scale joins-and-aggregations solution provided by the Cubert framework [34, 35]. All these data are stored in Pinot [26], our in-house distributed storage system, to be queried by the UI applications.

2.3.1 Metrics

LinkedIn has many diverse products. Even though there are a handful of company metrics that everyone optimizes towards, every product has several product-specific metrics that are most likely impacted by experiments in their "area". As LinkedIn's products evolve and new products emerge, it is impossible for the experimentation team to create and maintain all metrics for all products (currently more than 1000 of them). Therefore, to maintain the metrics, we follow a hybrid of centralized and decentralized model.

Metrics are categorized into 3 tiers: 1) Company wide 2) Product Specific 3) Feature Specific. A central team maintains tier 1 metrics. Ownership of tier 2 & 3 metrics is decentralized – each team owns the logic for these metrics while the central team is responsible for the daily computation and operations. XLNT computes all tier 1 and tier 2 metrics for all experiments while tier 3 metrics are only computed on an ad-hoc basis.

2.3.2 Multi-Dimensional Deep-Dive

When a metric is impacted, experimenters frequently want to dig deeper and get more actionable insights. For this reason, XLNT provides several slicing and dicing capabilities. Experimenters can leverage both user and non-user based dimensions and even apply multiple dimensions at once. As an example, the metric total pageviews can be narrowed down to homepage pageviews on the iOS app for Spanish speaking members across the US, South America and Spain.

As one can easily imagine, this is computational extremely heavy especially when over 1000 metrics are involved across multiple days for hundreds of experiments. At a high level there are two use cases we need to address within experiment reporting: 1) Enable a *broad* understanding of the impact across LinkedIn 2) Enable a *deep* understanding in the areas most heavily impacted by the experiment.

Knowing that this functionality clearly falls in the second use case, we decided to provide multi-dimensional drill-down only for the subset of metrics likely to be impacted by the experiment. Furthermore, from the dozens of dimensions available, teams decide what dimensions are most relevant for their experiments and at what level of combinations. This saves us from unnecessarily crunching data that is not relevant and no one uses. Even with such savings, our pipeline still generates about 150 million records daily on average, where each record includes summary statistics for the tuple of experiment, variant, date range, metric and dimension combination.

3 BEYOND THE BASICS

We have described the components in XLNT that enable the basic workflow for running experiments at LinkedIn. In this section, we discuss, with real examples, how we address a few challenging scenarios. Even though the examples may be LinkedIn specific, most of the lessons and best practices we share here are applicable to experimenting on social networks in general.

3.1 Interactions between Experiments

As mentioned in Section 2.1, experiments at LinkedIn are fully overlapping *by default*. In other words, a member is simultaneously in all applicable experiments. Unique hashIDs are used for each experiment to ensure the randomizations between experiments are orthogonal. The simple parallel experimentation structure allows us to scale the number of experiments easily in a de-centralized manner. It is sufficient for most of our A/B testing needs as many of our experiments affect entirely different products and are unlikely to interfere or interact with each other. However, there are cases where interactions are expected. For example, one experiment was testing whether or not to include a LinkedIn Pulse module on the homepage, while simultaneously we had another experiment investigating the number of stories to include in the same module. Clearly, when the module does not exist, the second experiment is ineffective. Another example of potential interaction is between two email experiments. When a member receives two emails from LinkedIn, he or she is likely to open only one of them. Hence two email experiments both improving the subject line are in fact competing with each other. Each of them would have enjoyed a larger gain if the experiments were run on two disjoint user spaces.

Google's experimentation system uses layers and domains to divide up the user space to avoid such conflicts or interactions [8]. Microsoft Bing has a similar approach but their system includes detection in addition to prevention [9]. Different from these two systems, the XLNT platform takes a de-centralized approach that is closely integrated with the LinkedIn engineering infrastructure. We describe here how we use XLNT to address the three most common concerns and use cases related to interactions at LinkedIn.

Gating Key. Using the LinkedIn Pulse module example, a testKey is created that acts as a gating key to control whether the module is on or off. A second testKey is then used to split the traffic to test different number of stories. The second testKey would only be evaluated if the gating key indicates that the module is "on". This nested structure ensures that users do not see five stories without the module. The same gating key concept is also used to create disjoint traffic for multiple experiments. For instance, we always have several experiments running simultaneously on the homepage; some are improving the feed relevance while others are modifying UI elements. When one is concerned about potential interaction between N of them, he or she can create a gating key with N variants and each variant acts as a bucket that sends traffic only to one of the N experiments.

Factorial Design. Even though LinkedIn has over 300 million members, there are product areas where the user base is relatively small in comparison to other parts of the site. Some of these products are heavily experimented on, for example, the Subscription acquisition page. Small UI changes on the subscription page can make a big difference monetarily. With multiple UI experiments running in parallel, there is valid concern regarding interference. However, splitting up the traffic makes it even harder to have sufficient power to detect changes. It is also not practical to run experiments sequentially since each experiment runs for at least a month to monitor long-term user impact. On the other hand, if we set up these experiments independently (the default setup) each experiment becomes a factor in a full factorial design. We can then analyze to see if these experiments do interact and if so, what their effects would be without interaction. Of course, if there is no significant interaction, each experiment can be analyzed separately and each gets to enjoy the full amount of traffic available for maximum power.

Dependent Experiments. There are cases where people are interested in only *certain* combinations of different factors, but not all. For instance, we have one experiment testing the maximum number of sponsored updates to include in one's feeds per day, while another experiment varying the gap between two consequent sponsored updates. What is the best combination of these two factors? Some combinations, for example, a maximum of 25 sponsored updates with a gap of 3 organic updates in between, should never be launched due to bad user experience. Testing these combinations not only wastes valuable traffic, but also hurts the member experience. In most cases, creating a single test that combines the two factors is impractical, but we can align them using the same hashID so that members are randomized identically in both experiments. We can then create the variants for the second experiment such that when combined with the first they generate the right combinations. Figure 5 illustrates the idea.

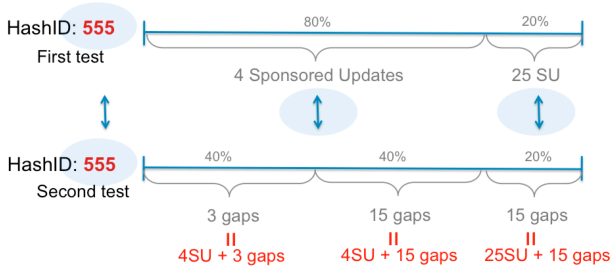

Figure 5: Using the same hashID to align two experiments.

3.2 Experimental Units

The first decision to make when running an experiment is whom to run it on. The experimental unit refers to both the entity randomly assigned to treatment and control (randomization unit) and the unit metrics are calculated and averaged over (analysis unit). We assume these two units are the same in our discussion. The most common case when they do differ (e.g. computing the metric Click-Through-Rate when experimenting on users) is well discussed in the literature [8, 13] and hence omitted in this paper.

In this section, we use guest experimentation as an example to discuss how XLNT supports different experimental units and some of the challenges we encounter. In addition, we introduce an interesting problem that arises particularly in a social network setting where the same user can play two different roles with each needing to be tested separately.

3.2.1 Guest Experiments

Most of our experiments focus on optimizing the member experience. However, we also run experiments on public pages such as the registration page and public profiles. Providing an excellent user experience on these pages is critical because they ultimately drive sign-ups.

The XLNT platform makes it easy to run an experiment on guest users (or any other unit the experimenter requires). A browser ID, instead of a member ID, is usually used to identify guest users and is passed to the "getTreatment" call the same way as a member ID (see Section 2.2). The browser ID is associated with a different URN type called "guest" which is then recorded as part of the experiment logs (Section 2.2.1). Targeting is also possible. Even though there is no rich profile data associated with a browser ID, properties that are available at run time such as the browser type, or the geographic location computed by reversing the IP address

can all be used for targeting (Section 2.1.2). Our offline analysis pipeline uses the URN type to differentiate between experimental units and automatically joins metrics with the corresponding experiment assignment information. This enables us to have a single workflow to handle analysis on various experimental units.

One challenge is to handle transitions between member and guest status of the same user during an experiment. There are mainly two cases when this happens: a guest user becoming a member, or a member transitioning between logged-in and logged-out states. The latter is less of a concern since a member's logged out activities on the site is very limited and has little influence over our experiments. The former, however, is of particular concern for experiments optimizing registration and onboarding flow as one unified experience. For example, we have one guest experiment promoting registration using the "Signup with Facebook" option, in which case certain information can then be automatically filled during the onboarding process to provide a better member experience. Even though the experiment is randomized on the browserID, we need to have a measurement based on both browserID and memberID. This is a common use case as information collected on the guest side helps us tailor the experience on the member side. To achieve the continuity, we can make a small customization on the application client (Section 2.2) to emit two pieces of tracking events, one based on the browserID (the default one) and the other on the memberID. The analysis workflow then automatically picks up both events and generates two reports, one for each ID.

Other experimental units have their unique challenges as well. Take experimenting on LinkedIn groups as an example. A member can belong to multiple groups, so we can no longer assume independence between units. This presents a similar challenge as the network A/B testing problem to be discussed in Section 3.4.

3.2.2 Dual Roles of a User

There are different experimental units even when we only consider members on the site. This is the case because every member holds two different roles in a social network as both the one performing an action and the one receiving an action. For example, a member can view others' profiles or receive profile views. Similarly, they can endorse someone or be endorsed.

It is important to know which role we are experimenting on and analyze accordingly. If an experiment is randomizing on endorsers, endorsees are going to be evenly distributed across treatment and control. Therefore, analyzing the number of endorsements received gives no indication on how the experiment performs. Similarly, an experiment that encourages members to send more invitations should not be evaluated based on the number of invitations received. Essentially, even though the total number of actions is the same counting by either role, how the actions are attributed needs to be aligned with the experimental unit. Moreover, some attributions, such as the number of invitations accepted, are time-lagged, in which case we need to create additional metrics to capture the time dimension (e.g. invitation accepted by day X).

We need to be even more careful when we have experiments aiming at assessing impacts on both sides. As an example, we recently considered allowing members to include a custom background image on their profile page. We wanted to evaluate whether members would view more profiles and receive more profile views. Two experiments (testKeys) were created, one allowing a member to upload an image (viewee experiment) and the other allowing a member to see the image if there is one (viewer experiment). However, these two experiments cannot be fully independent because viewer and viewee are simply opposite perspectives of the same member. Members who are able to upload background images should also be able to see them, especially on their own profiles. To address this issue, we applied the same hashID to both experiments to ensure the same treatment assignments. A more subtle but important point is that the experiment effect depends on the percentage of members receiving the treatment. Clearly, if only 1% of profiles have the background image, a viewer's behavior is less likely to be impacted compared to if all profiles had a background image. Therefore, the bias is smallest when the treatment is ramped close to 100%. On the other hand, running the experiments at 50/50 gives the least variance. It is thus a bias and variance trade-off that the team has to keep in mind when evaluating the experiment results.

3.3 Offline Experiments

In an online experiment, users trickle into the A/B test as they visit LinkedIn. The experimenter has no prior information on who will be in the experiment, and users' engagement is measured through their activities on the site, e.g. pageviews and clicks. Many of the experiments we run at LinkedIn are such online experiments. However, because LinkedIn is a member-based social network, there are also many experiments that are "offline", in which case the experiment users do not necessarily have an online activity. We walk through three scenarios below.

Email Experiments. Emails are usually sent to members independent of whether they visit the online site or not. For example, a member would receive a notification email if another member sends them an invitation to connect. Not everyone who gets an email would come to the site. Therefore, if we only consider active members when analyzing an email experiment we are likely to introduce a bias. For instance, if the treatment email has a more attractive subject line that encourages more members to click and visit the online site, we are likely to see a drop in many key metrics (e.g. average pageviews) if only active members are taken into account because the email tends to drive less engaged members to visit LinkedIn. In other words, the experiment population cannot be determined only with online user activity. It needs to take into account all users who received the email. To make it even more complex, we also have experiments that attempt to reduce the email footprint by filtering out emails that are less relevant to members. In these cases, it is not sufficient to only look at users who actually receive the email, but to also include users who would have received the email had there been no relevance filtering.

Email Campaign. We have recently revamped the Who Viewed My Profile page. The product team wants to measure through an A/B test if the new changes are indeed better, and if so, by how much. The marketing team wants to create a buzz around the new page by starting an email campaign. This is a very common scenario, but how can the A/B test and the email campaign coexist? Clearly, we can only send campaign emails to the treatment group as there is nothing new for members in control. However, this would contaminate the online A/B test because the campaign encourages more members from the treatment to visit. To make sure the experiment samples are unbiased, similar to an email experiment discussed earlier, we need to identify the matching population who would have received the email in the control group as well. For instance, if the emails are sent to treatment members selected through process X, the same process needs to be applied to the control group to select those who would

have received the email if they were in treatment. It is important to note that even though these two populations are unbiased, the difference between the two populations is no longer just the effect of the newer page, but a compounded effect of both the email campaign and the new page. It may be a subtle point, but this combination is in fact a better representation of the final impact we would like to measure from launching the new page. It is, however, crucial to realize that email campaigns have a strong novelty effect that dies out over time, so we need to keep the experiment running for a longer time period till the effect stabilizes. There are also occasions where one wants to measure the impact of these two factors separately, in which case we would either create a separate treatment bucket that receives no email campaign, or measure the A/B test first before setting off the campaign.

Cohort Experiments. It is common practice at LinkedIn to run A/B tests on a cohort of members who are pre-selected offline. As an example, we tested a new feature where members get a push notification on their LinkedIn profile page if they have not visited the homepage for a while. The first thing the experimenter did was to gather a list of members who have not visited the homepage for more than 7 days and used it to create a customized targeting attribute (Section 2.1.2). These targeted members are then split between treatment and control, and the notification was presented to only members in the treatment cohort. Such cohort experimentation is a great way to leverage the capability of the XLNT platform to test out ideas without having to build a full-fledged product. Another advantage of cohort experiments is to be able to measure long-term member impact without the dilution of members newly added to experiments. However, one common pitfall is that people try to update the cohort selection during the experiment, while the selection criteria is directly related to the experiment outcome. In the push notification example, the treatment encourages more members to visit LinkedIn homepage, though once a member visits the homepage, he or she would disappear from the updated cohort list. Therefore the updated treatment bucket would end up with fewer members which creates a bias. Our suggestion is to use a static cohort or to include the previous cohort in the analysis if an update is necessary.

3.4 Network A/B Testing

In our discussion thus far, we have assumed the Rubin causal model [1], a standard machinery of testing framework, when conducting and analyzing A/B tests. A key assumption made in Rubin causal model is the "Stable Unit Treatment Value Assumption" (SUTVA), which states that the behavior of each sample in the experiment depends only on their own treatment and not on the treatments of others.

This is a plausible assumption in most practical applications. For example, a user who is served better search results is more likely to click, and that behavior is entirely independent of others using the same search engine. However, such assumption does not always hold in experiments run on social networks. In a social network setting, a user's behavior is likely impacted by that of their social neighborhood [2, 3, 4]. In most cases, a user would find a new feature more valuable and hence more likely to adopt it as more of their neighbors adopt it. For example, video chat is a useless feature unless one's friends use it too. In an A/B experiment, this implies that if the treatment has a significant impact on a user, the effect would spill over to his/her social circles, regardless whether his/her neighbors are in treatment or control.

This poses a special challenge for running A/B tests in many online social and professional networks like Facebook, Twitter and LinkedIn. Many features tested there are likely to have network effects. For example, a better recommendation algorithm in treatment for the People You May Know module on LinkedIn encourages a user to send more invitations. However, users who receive such invitations can be in the control variant and when they visit LinkedIn to accept the invitation they may discover more people they know. If the primary metric of interest were the total number of invitations sent, we would see a positive gain in both the treatment and the control groups. The treatment effect estimated ignoring network effect would be biased and would not fully capture the benefit of the new algorithm. Such bias exists in testing almost any features that involve social interactions, which is truly ubiquitous in a social network environment.

To address this challenge, we need to take users' network connections into consideration when sampling them into treatment and control [19]. Essentially, we take the following two-stage procedure:

1. Partition the users into clusters.
2. Treat each cluster as a unit for randomization so all users in one cluster have the same experiment assignment.

The estimation for the treatment effect can then be based on either the sample means, or more sophisticated estimators summarized in [19]. We have noticed strong network effects from the network A/B tests we have run at LinkedIn based on this sampling and estimation framework.

4 FOSTERING AN EXPERIMENTATION CULTURE

Running large scale A/B tests is not just a matter of infrastructure and best practices, establishing a strong experimentation culture is also key to embedding A/B testing as part of the decision making process. Apart from building the basic functionalities any A/B testing platform requires, we have identified four necessary ingredients that have helped us instill experimentation deeply into our culture at LinkedIn. We share them in the rest of the section.

4.1 Integration with Business Reporting

XLNT produces over 1000 metrics for A/B test reports. Similar metrics are often used in business reporting for decision making as well. These two sets of metrics used to be computed separately and much effort was spent on investigating differences when they happened.

We realized the importance of using the same definitions of metrics across reporting and experimentation, and hence have worked with various teams at LinkedIn to evolve our metrics pipeline to be the *unified* metrics pipeline leveraged by the entire company. By making our experiment results and business reports "comparable", we have made it possible for R&D teams to relate changes in business numbers with experiment launches. Moreover, the integration also provides the foundation that enables other organizations such as Finance to bake A/B test results into business forecasting.

4.2 Leveraging Site Wide Impact

In addition to providing product development with a directional signal, A/B testing is often utilized to measure impact and assess ROI (Return-On-Investment). A report based on triggered analysis (Section 2.2.1) is great at providing a directional signal, however, it does not accurately represent the *global* lift that will occur when the winning treatment is ramped to 100% (holding everything else constant). The reason is two-fold: 1) Most experiments only *target*

a subset of the entire user population. 2) Most experiments only *trigger* for a subset of their targeted population. In other words, triggered analysis only provides evaluation of the local impact, not the global impact of an experiment.

To address this concern, we compute the Site Wide Impact, defined as the percentage delta between two parallel universes: one with treatment applied to only *targeted* users and control to the rest, the other with control applied to all. With site wide impact provided for all experiments, teams are able to compare results across experiments regardless of their targeting and triggering conditions. Moreover, Site Wide Impact from multiple segments of the same experiment can be added up to give an assessment of the total impact, which is particularly helpful when each segment is running at different percentages and has to be analyzed separately to avoid Simpson's Paradox [31].

However, computing Site Wide Impact explicitly for every metric and report we generate would be more than doubling our current computation effort. It turns out that Site Wide Impact depends only on (1) the summary statistics that are already computed as part of the triggered report, and (2) the global total with a matching analysis date range as the experiment report (see Appendix A). For most of our metrics that are additive across days, we can simply keep a daily counter of the global total and add them up for any arbitrary date range. However, there are metrics, such as the number of unique visitors, which are not additive across days. Instead of computing the global total for all date ranges we generate reports for (an $O(n^2)$ problem), we estimate them based on the daily totals (see Appendix A for more details), saving more than 99% of the computation cost without sacrificing much accuracy.

One interesting phenomenon we observe is that the local impact (percentage delta from triggered analysis) and the Site Wide Impact can sometimes disagree directionally for ratio metrics such as CTR. For example, in a recent experiment testing "connection request" emails targeting English speaking members with fewer than 50 connections, we observed a -9.4% local impact on CTR while a +0.5% Site Wide Impact. Mathematically, this is an instance of Simpon's Paradox as the overall site wide CTR is essentially a linear combination of the within-segment and outside-segment CTR. The real challenge, however, is to decide which number to use for decision making. We believe that even though Site Wide Impact has a stronger business implication, the local impact is a better indication of user satisfaction in this case.

4.3 Simplifying Multiple Testing

Multiple testing is a common problem in the field of A/B testing [30] and LinkedIn is no exception. With over 1000 metrics computed for each experiment, this issue is so prevalent that the most common question we hear from experimenters is "why is this irrelevant metric significant?"

Even though we have tried to educate people on the topic of multiple testing, many are still clueless as what they should do when a metric is unexpectedly significant. What's worse, some people even lost trust in our A/B reports because they couldn't tell whether a metric is moved for real or simply due to multiple testing.

To this end, we have come up with a two-step rule-of-thumb for experimenters to follow:

1. Divide all metrics into three groups: 1) First order metrics are those expected to be impacted by the experiment; 2) Second order metrics are those potentially impacted, e.g. through cannibalization; 3) Third order metrics are those unlikely to be impacted.
2. Apply tiered significance levels to each group. We recommend our experimenters to use 0.05, 0.01 and 0.001 respectively.

Our rule-of-thumb is in fact based on an interesting Bayesian interpretation. It boils down to how much we believe the null hypothesis (H_0) is true before we even run the experiment. Believing that the posterior probability on H_0 should be consistent across all metrics, given different priors on H_0, we essentially vary the probability of observing the data given H_0, which is the definition of p-value in the frequentist interpretation. Since the significance level is the cutoff used for p-values, this leads to adjusting the thresholds accordingly. In particular, the three significance levels 0.05, 0.01 and 0.001 reflect that we believe an experiment having a prior probability of 1/2, 1/6 and 1/51 impacting each of the three metric groups.

This interpretation establishes a simple mathematical equivalence between people's prior belief on whether a metric is impacted and the significance level. Thus for those who are a bit more sophisticated they can easily come up with their own significance level tiers to match their prior beliefs.

4.4 Tracking the Most Impactful Experiments

LinkedIn is a deeply interconnected site. When it comes to experimentation, each team in fact plays two different roles. On the one hand, they run experiments to improve their metrics; on the other hand, they are also responsible for the global health of the same metrics, even when they may be impacted by experiments from other teams. Recognizing the importance of the second responsibility, we launched a feature called "Most Impactful Experiments" (MIE) to allow metric owners to follow experiments that impact the metrics they care about.

MIE is a major milestone for experimentation at LinkedIn. Not only does it drive greater transparency regarding experiment launch decisions, it also encourages more discussions to occur which in turn increases the overall knowledge of experimentation in the company.

To ensure MIE only listed the most relevant experiments, we use the following three-step algorithm:

1. Filter out all the experiments that have potential quality issues based on our alerting system.
2. For each metric, control False Discovery Rate using the Benjamini-Hochberg algorithm [17].
3. Score the experiments from step 2 based on three factors: Site Wide Impact, treatment percentage and experiment duration. These factors are then combined using the Analytical Hierarchy Process [18]. To scale to over 1000 metrics, we use empirical cumulative density functions to automatically control the fraction of experiments selected for each metric. Furthermore, historical experiments are used to ensure the relevance of selected experiments is not affected by the number of high-impact experiments available at any given time.

There are a couple of interesting observations and lessons learnt while developing this capability:

- **Large Initial Impact**. We observed that it is common for newly activated experiments to have overly positive or negative impacts. However, for these experiments the impact tends to shrink over time or become statistically

insignificant. While such scenario can be due to novelty effect, most of them are just statistical artifact as explained in [7]. Controlling the false positive rate was helpful in eliminating these, but to further alleviate the problem, we only consider experiments with results over at least three days and penalize short experiments in our ranking algorithm.

- **Personalizing Threshold**. Picking the optimal threshold presents a challenging trade-off between precision and recall. Interestingly, three months after launching MIE, we have noticed that there are two well-separated groups of users. The first group is the functional users, including engineers, product managers and data scientists who themselves own experiments. The second group is the managerial users, including top-level managers and executives. Most users from the first group tend to follow metrics that are mostly impacted by their experiments. A higher recall can allow them to see more experiments beyond the ones they own. On the other hand, we want to keep a higher threshold for the second group of users to maintain a high precision.

5 CONCLUSION

In this paper, we shared the details of building a powerful and flexible experimentation platform that enables teams across LinkedIn to make informed decisions faster and at scale. Furthermore, we discussed several challenging A/B testing scenarios and shared many real examples applicable to social networks. Realizing the importance of a strong experimentation culture, we discussed four ingredients that helps us instill experimentation deeply into every decision making process at LinkedIn, even across roles outside of R&D.

One challenge we have not discussed here is how to provide guided insights to the experiment owners. Traditionally, controlled experiments have been used to answer "what" has been impacted, but not to answer "why". By utilizing the amount of information we have available for each experiment, in terms of both the metrics and dimensions, we hope to automatically generate insights that can better guide product development.

There are also cases where it is not possible to run an A/B test due to practical reasons, so we have to leverage quasi-experimental designs where users in treatment and control groups are matched post hoc using techniques such as propensity score matching.

Lastly, we hope that the topics we covered will lead to further research and development of A/B testing in large scale social networks.

6 ACKNOWLEDGMENTS

The authors wish to thank June Andrews, Drew Moxon, Karan Ahuja, Alexis Baird, Caroline Gaffney, Udi Milo and Xin Fu for insightful discussions on several of the examples. We have been fortunate to be part of the team developing the experimentation platform and we wish to thank *all* members of the XLNT team, including Adam Smyczek who made significant contributions before leaving LinkedIn. Finally, we wish to thank many people for being strong advocates of experimentation at LinkedIn, especially Igor Perisic.

7 REFERENCES

[1] **Rubin, Donald B.** Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66(5):688, 1974.

[2] **Ugander, Johan, Karrer, Brian, Backstrom, Lars and Kleinberg, Jon.** Graph cluster randomization: network exposure to multiple universes. *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 329–337. ACM, 2013.

[3] **Katzir, Liran, Liberty, Edo and Somekh Oren.** Framework and algorithms for network bucket testing. *Proceedings of the 21st international conference on World Wide Web*, pages 1029–1036. ACM, 2012.

[4] **Toulis, Panos and Kao, Edward.** Estimation of causal peer influence effects. *Proceedings of The 30th International Conference on Machine Learning,* pages 1489–1497, 2013.

[5] **Eckles, Dean, Karrer, Brian and Ugander, Johan.** Design and analysis of experiments in networks: Reducing bias from interference. arXiv preprint arXiv:1404.7530, 2014.

[6] **Aronow, Peter M, and Samii, Cyrus.** Estimating average causal effects under general interference. arXiv preprint arXiv:1305.6156, 2013

[7] **Kohavi, Ron, et al.** Trustworthy online controlled experiments: Five puzzling outcomes explained. *Proceedings of the 18th Conference on Knowledge Discovery and Data Mining.* 2012, www.exp-platform.com/Pages/PuzzingOutcomesExplained.aspx.

[8] **Tang, Diane, et al.** Overlapping Experiment Infrastructure: More, Better, Faster Experimentation. *Proceedings 16th Conference on Knowledge Discovery and Data Mining.* 2010.

[9] **Kohavi, Ron, et al.** Online Controlled Experiments at Large Scale. KDD 2013: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. 2013*. http://bit.ly/ExPScale.

[10] **Kohavi, Ron, et al.** Seven Rules of Thumb for Web Site Experimenters. KDD 2014: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining.* 2014.

[11] **Yates, Frank,** Sir Ronald Fisher and the Design of Experiments. *Biometrics,* 20(2):307–321, 1964.

[12] **Bakshy, Eytan, Echles, Dean and Bernstein, Michael S.** Designing and Deploying Online Field Experiments. *Proceedings of the 23rd international conference on World Wide Web,* pages 283-292, ACM, 2014

[13] **Kohavi, Ron. et al.** Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery.* February 2009, Vol. 18, 1, pp. 140-181. http://www.exp-platform.com/Pages/hippo_long.aspx.

[14] **Crook, Thomas, et al.** Seven Pitfalls to Avoid when Running Controlled Experiments on the Web. [ed.] Peter Flach and Mohammed Zaki. KDD '09: *Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining.* 2009, pp. 1105-1114. http://www.exp-platform.com/Pages/ExPpitfalls.aspx.

[15] **Ioannidis, John PA.** "Why most published research findings are false." *PLoS medicine* 2.8 (2005): e124.

[16] **Wacholder, Sholom, et al.** "Assessing the probability that a positive report is false: an approach for molecular epidemiology studies." *Journal of the National Cancer Institute* 96.6 (2004): 434-442.

[17] **Benjamini, Yoav, and Yosef Hochberg.** "Controlling the false discovery rate: a practical and powerful approach to

multiple testing." *Journal of the Royal Statistical Society. Series B (Methodological)* (1995): 289-300.

[18] **Saaty, Thomas L.** "How to make a decision: the analytic hierarchy process."*European journal of operational research* 48.1 (1990): 9-26.

[19] **Gui, Huan, Xu, Ya, Bhasin, Anmol, Han Jiawei.** Network A/B Testing: From Sampling to Estimation. *Proceedings of the 24rd international conference on World Wide Web,* ACM, 2015

[20] **Box, George EP, J. Stuart Hunter, and William G. Hunter.** "Statistics for experimenters: design, innovation, and discovery." *AMC* 10 (2005): 12.

[21] **Gerber, A. S., and Green, D. P.** Field Experiments: Design, Analysis, and Interpretation. WW Norton, 2012

[22] **Sumbaly, Roshan, et al.** "Serving large-scale batch computed data with project voldemort." *Proceedings of the 10th USENIX conference on File and Storage Technologies*. USENIX Association, 2012.

[23] **Tate, Ryan.** The Software Revolution Behind LinkedIn's Gushing Profits. [Online] http://www.wired.com/2013/04/linkedin-software-revolution

[24] **Auradkar, Aditya, et al.** "Data infrastructure at LinkedIn." *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*. IEEE, 2012.

[25] **Kreps, Jay, Neha Narkhede, and Jun Rao.** "Kafka: A distributed messaging system for log processing." *Proceedings of 6th International Workshop on Networking Meets Databases (NetDB), Athens, Greece*. 2011.

[26] **Naga, Praveen Neppalli,** Real-time Analytics at Massive Scale with Pinot. [Online] September 29, 2014 http://engineering.linkedin.com/analytics/real-time-analytics-massive-scale-pinot

[27] **Fisher, Ronald A.** Presidential Address. *Sankhya: The Indian Journal of Statistics.* 1938, Vol. 4, 1. http://www.jstor.org/stable/40383882.

[28] **Montgomery, Douglas C.** *Design and analysis of experiments*. John Wiley & Sons, 2008.

[29] **Betz, Joe, Tagle, Moira.** Rest.li:RESTful Service Architecture at Scale. [Online] February, 19, 2013 https://engineering.linkedin.com/architecture/restli-restful-service-architecture-scale

[30] **Romano, Joseph P. Azeem M. Shaikh and Michael Wolf.** 2010b Multiple Testing. *New Palgrave Dictionary of Economics.* http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.418.4975&rep=rep1&type=pdf

[31] **Wikipedia.** Simpson's Paradox. [Online] http://en.wikipedia.org/wiki/Simpson%27s_paradox

[32] **McFarland, Colin.** Experiment!: Website conversion rate optimization with A/B and multivariate testing. s.1. : New Riders, 2012.978-0321834607

[33] **Eisenberg, Bryan.** How to Improve A/B Testing. ClickZ Network. [Online] April 29, 2005. www.clickz.com/clickz/column/1717234/how-improvem-a-b-testing.

[34] **Vemuri, Srinivas, Varshney, Maneesh, Puttaswamy, Krishna and Liu, Rui.** Execution Primitives for Scalable Joins and Aggregations in Map Reduce. *Proceedings of the VLDB Endowment, Vol. 7, No. 13*

[35] **Varshney, Maneesh, Vemuri, Srinivas.** Open Sourcing Cubert: A High Performance Computation Engine for Complex Big Data Analytics [Online] November 11, 2014 https://engineering.linkedin.com/big-data/open-sourcing-cubert-high-performance-computation-engine-complex-big-data-analytics

APPENDIX

A. Site Wide Impact

We use the average number of clicks as an example metric to show how we compute Site Wide Impact. Let Let $X_t, X_c, X_{seg} and\ X_{global}$ denote the total number of clicks in the treatment group, the control group, the whole segment (including the treatment, the control and potentially other variants) and globally across the site, respectively. Similarly, let $n_t, n_c, n_{seg} and\ n_{global}$ denote the sample sizes for each of the four groups mentioned above.

It is easy to see that the total number of clicks in the *treatment (control) universe* can be estimated as

$$X_{tUniverse} = \frac{X_t}{n_t} n_{seg} + (X_{global} - X_{seg})$$

$$X_{cUniverse} = \frac{X_c}{n_c} n_{seg} + (X_{global} - X_{seg})$$

Then the Site Wide Impact is computed as

$$SWI = \left(\frac{X_{tUniverse}}{n_{tUniverse}} - \frac{X_{cUniverse}}{n_{cUniverse}}\right) / \frac{X_{cUniverse}}{n_{cUniverse}}$$

$$= \left(\frac{\frac{X_t}{n_t} - \frac{X_c}{n_c}}{\frac{X_c}{n_c}}\right) \times \left(\frac{\frac{X_c}{n_c} n_{seg}}{\frac{X_c}{n_c} n_{seg} + X_{global} - X_{seg}}\right)$$

$$= \Delta \times \alpha$$

which indicates that the Site Wide Impact is essentially the local impact Δ scaled by a factor of α, and X_{global} is the only ingredient that is not already computed as part of summary statistics for the triggered analysis.

For metrics such as average number of clicks, X_{global} for any arbitrary date range can be computed by summing over clicks from corresponding single days. However, for metrics such as average number of unique visitors, de-duplication is necessary across days. To avoid having to compute α for all date ranges we generate reports for, we estimate cross-day α by averaging the single-day α's.

There is one other group of metrics that are ratio of two metrics. One example is Click-Through-Rate, which equals Clicks over Impressions. The derivation of Site Wide Impact for ratio metrics is similar, with the sample size replaced by the denominator metric.