Ad Clustered User Randomized Trials

Edvard Bakhitov[1], Congshan Zhang[1], Ivan Sherstobitov[2], Brett Daley[3], Daniel Ting[2], Houssam Nassif[2], and Sergei Leonenkov[2]

[1]Central Applied Science, Meta
[2]Ads Online Experimentation, Meta
[3]University of Alberta

1 Introduction

Cluster randomization has become a commonly used approach to address interference issues in A/B testing for various social network and two-sided marketplace settings [6, 3]. In this paper, we propose a novel application of clustering to boost experimentation throughput, that is, the ability to conduct more A/B tests at any given time while maintaining a certain level of minimum detectable effect (MDE).

Major internet companies carry out a large number of online A/B tests each year and this number has been consistently increasing. The trend puts significant strain on experimentation platforms to deliver higher throughput. For a feed-based advertising businesses like Meta, the main focus is on measuring the user treatment effect. However, the standard design of user-randomized A/B tests has hit the throughput bottleneck, and it has become increasingly more challenging to meet extra demand. While various forms of regression adjustment [1, 7, 4, 5] have been proposed as a means of improving estimation efficiency which can be equivalently translated to additional throughput, the benefits of these methods have saturated.

Under such contexts, we propose a novel structural experiment design: *Ad Clustered User Randomized Trial* (ACURT). Unlike traditional user-randomized A/B testing, ACURT involves differentiated user randomization on various ad clusters. This unique segmentation design allows for more efficient utilization of user information, resulting in substantial throughput benefits. In the ACURT framework, users are first randomized within each ad cluster and the user-cluster pair level metric values are then aggregated onto each treatment condition for treatment effect estimation. As illustrated in Figure 1, ads are grouped into two clusters based on their auction participation graph, ensuring that the majority of auctions only contain candidates of the same cluster. Based on real A/A test data at Meta, the ACURT design with two ad clusters leads to 30% throughput increase for two-week experiments and around 40% for three-week experiments, holding MDE unchanged.

Our contributions are twofold: first, we present the ACURT experiment design, which significantly increases throughput. Second, we introduce a heterogeneous balanced partitioning algorithm and a real-time ML approach designed to incrementally maintain cluster quality in support of the new design.

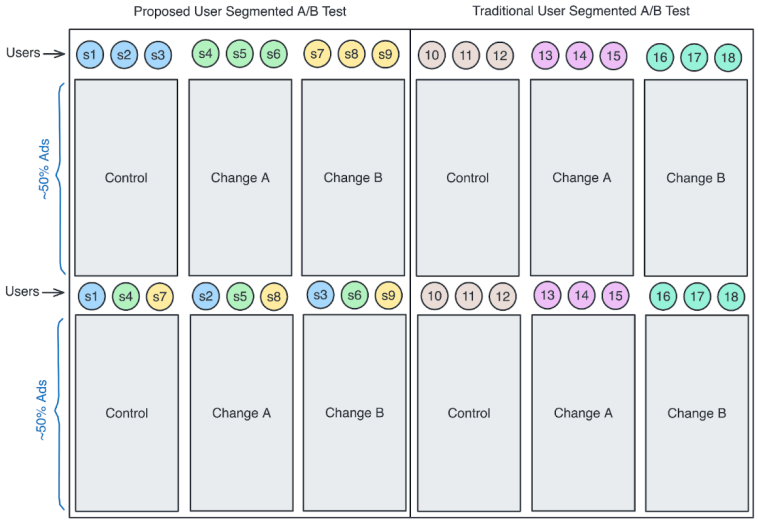

Figure 1: A comparison between ACURT and the traditional user-randomized A/B test. The key difference between them is that in ACURT users are reshuffled between two clusters of ads.

2 Experiment Design

We begin by discussing the intuition of the ACURT design and its mechanism for reducing variance. Imagine that we are given two predefined ad clusters. Let's consider the difference-in-means estimator for the average treatment effect in a standard user randomized A/B trial. We only focus on one treatment group; the analysis for other treatment groups is analogous. The average outcome can be written as

$$(Np)^{-1}\sum_{i=1}^{N}1(W_i=1)Y_i=(Np)^{-1}\sum_{i=1}^{N}1(W_i=1)(Y_{i1}+Y_{i2}), \tag{1}$$

where N is the total number of users in the experiment, W indicates the treatment received, p is the probability of being selected into the treatment group, and Y_{i1} and Y_{i2} are the outcome values under user i for cluster 1 and 2, respectively. Under this two-cluster setup, the within-user variance equals $Var(Y_{i1})+Var(Y_{i2})+2Cov(Y_{i1},Y_{i2})$, where the covariance term contributes positively to the user-level variance and thus the variance of the group mean for a typical business metric. Now, imagine that we could replace Y_{i2} with Y_{j2}, where j is a different user than i, then $Cov(Y_{i1},Y_{j2})=0$, and the group mean estimator in cluster 2, computed by averaging over either Y_{i2} or Y_{j2}, can be shown unbiased under certain conditions. Motivated by this observation, in the ACURT design we randomly reshuffle users in the second cluster, as depicted in fig. 1 and use the following estimator to replace Eq. 1

$$(Np)^{-1}\left(\sum_{i=1}^{N}1(W_{i1}=1)Y_{i1}+\sum_{i=1}^{N}1(W_{i2}=1)Y_{i2}\right), \tag{2}$$

where W_{i1} and W_{i2} are two independent Bernoulli draws in two clusters for the same user i. Under the stable unit treatment value type assumption (SUTVA) that the potential outcome in one cluster is not affected by the treatment received in the other, that is, following the standard

Rubin's potential outcome notation, $Y_{i1}(W_{i1}, W_{i2}) = Y_{i1}(W_{i1})$ and $Y_{i2}(W_{i1}, W_{i2}) = Y_{i2}(W_{i2})$ for all i, the estimator in Eq. 2 is unbiased for $E[Y(1)]$. In marketplace environments, such as online advertising, advertisers compete for user impression slots through bidding in auctions. For the SUTVA to hold in such setup, a crucial requirement is that clusters must isolate auctions instead of individual ad candidates, and that a user's conversion behavior in one cluster does not influence their behavior in the other cluster. While these are generally considered strong assumptions in practice, they can be argued reasonable when testing certain (e.g., late-stage) ranking models.

3 Role of Ads Clustering

Unlike regression adjustment methodologies, ACURT requires making an explicit trade-off between bias (from potential violation of the SUTVA assumption) and variance, hence, its performance heavily hinges on the properties of the underlying ad clusters. In order to achieve both low bias and low variance, we need the ad clusters to have both **high auction purity** and **high intra-user mixture**. We say that the auction is pure if all (competitive) ads participating in the auction belong to the same cluster. Hence, if the majority of auctions are pure, the auction stage inference would be low, leading to close-to-negligible bias. In the pure case, the auction outcome is only affected by a single treatment, not considering budget effect, throttling, or users changing their behaviors across auctions due to inconsistent treatment assignment. Similarly, we say that the user is mixed if auctions under the same user belong to different clusters. Intuitively, the more mixed users we have, the more within-user variance will be removed, leading to smaller platform MDE. Thus, achieving the optimal bias-variance trade-off boils down to trading off auction purity for intra-user mixture.

To this end, we propose a new clustering algorithm called *Heterogeneous Balanced Partitioning* (HBP) that naturally extends the Social Hash partitioning algorithm [2] by incorporating a user mixture penalty to the composite probabilistic fanout (p-fanout) objective function. The modified objective function explicitly models the trade-off between auction purity and intra-user mixture through the penalty parameter.

To formalize the problem, suppose we have an undirected heterogeneous graph that connects ad campaigns ($\mathcal{A}$), users ($\mathcal{U}$) and user requests/auctions ($\mathcal{R}$). We can define this graph as a collection of vertices (nodes) and edges, i.e., $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of graph vertices and $\mathcal{E}$ is a set of edges. Since $\mathcal{G}$ is a heterogeneous graph, we have three types of disjoint sets of vertices, $\mathcal{V} = \mathcal{A} \cup \mathcal{R} \cup \mathcal{U}$, and edges, $\mathcal{E} = E_{\mathcal{A}\mathcal{R}} \cup E_{\mathcal{A}\mathcal{U}} \cup E_{\mathcal{R}\mathcal{U}}$. Our goal is to partition ads into 2 buckets of approximately the same size such that we achieve high levels of auction purity and intra-user mixture.[1]

For a given partitioning $P = \{A_1, A_2\}$ of $\mathcal{A}$ and vertex $v \in V$, with $V \in \{\mathcal{R}, \mathcal{U}\}$, we define the fanout of v as the number of distinct buckets having an ad node incident to v, $\text{fanout}(P, v) = |\{A_i : \exists\{v, a\} \in E_{A_i V}, a \in A_i\}|$, where $E_{A_i V}$ is a set of edges between V and A_i vertices. To ensure the objective function is smooth, we follow [2] and measure the quality of partitioning P using the average probabilistic fanout,

$$\text{p-fanout}(P, v) = \sum_{i=1}^{k=2} \left(1 - (1-p)^{n_i(v)}\right),$$

where $p \in (0, 1)$ is the probability[2] that vertex a is connected to vertex v, and $n_i(v) = |\{a : a \in$

[1] The approach trivially generalizes to an arbitrary number of buckets.

[2] This probability is a tuning parameter which affects the behavior of the objective function. In practice, we set it to 0.5. We refer the reader to [2] for a more detailed discussion.

A_i and $\{v, a\} \in E_{A_i V}\}|$ is the number of ad vertices in bucket A_i adjacent to vertex v.

Note that whenever $\text{fanout}(P, r) = 1$, request r is pure. Similarly, if $\text{fanout}(P, u) = 2$, user u is mixed. Thus, fanout can be seen as a measure of impurity/mixture. This motivates the following optimization problem where we minimize the composite p-fanout,

$$\min_{\{A_1, A_2\}} \left\{ \underbrace{\frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \sum_{i=1}^{k=2} \left(1 - (1-p)^{n_i(r)}\right)}_{\text{auction impurity}} - \beta \cdot \underbrace{\frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \sum_{i=1}^{k=2} \left(1 - (1-p)^{n_i(u)}\right)}_{\text{intra-user mixture}} \right\}, \tag{3}$$

where $\beta \geq 0$ is a tuning parameter that balances out auction purity and intra-user mixture. Note that if we set β to 0, we will get the standard balanced partitioning fanout minimization objective function in [2]. When $\beta > 0$, higher levels of intra-user mixture will lead to smaller values of the composite objective function creating an incentive to trade off some auction purity for intra-user mixture.

4 Maintaining Clustering Quality: Real-time ML Approach

When there is a high turnover rate of ads, it must be decided how to efficiently assign new ads to clusters throughout the experiment. We achieve it by a real-time incremental clustering algorithm based on a supervised machine-learning (ML) approach. By refraining from the direct use of graph as training data in this process, we prevent the introduction of latency that comes from waiting for the auction graph to materialize. We first generate a labeled dataset of ads and cluster assignments using an incremental graph-based algorithm, similar to the one discussed in Section 3. In parallel, we extract ad-targeting features (e.g., user demographics). The data is randomly sub-sampled from both clusters to create a balanced dataset of feature-label pairs. A small, sigmoid-output neural network is trained to predict cluster assignments from ad details by minimizing the cross-entropy loss. Compared to graph-based clustering, training the ML model is inexpensive. After training, the model is exported to serve cluster assignments until the next iteration of graph-based clustering. Notably, no latency is introduced by this process because the previous iteration's model can continuously assign ads while the latest model is being trained.

This ML approach is highly flexible, giving us the opportunity to adjust the feature representation, model architecture, or optimization technique in order to improve performance as needed. Furthermore, by continuously receiving fresh labels from the graph algorithm periodically, the model can remain relatively accurate and stable over time without requiring a strong generalization ability.

5 Empirical evaluation

To evaluate the performance of ACURT, we run an A/A test, where the clustering achieves 70% of auctions having at least 80% of candidates from the same cluster and more than 20% of users being non-trivially exposed to ads from the two clusters. Figure 2 shows that ACURT achieves a throughput increase relative to the standard user-randomized A/B test higher than 30% for two-week experiments and 40% for three-week experiments, holding MDE the same.

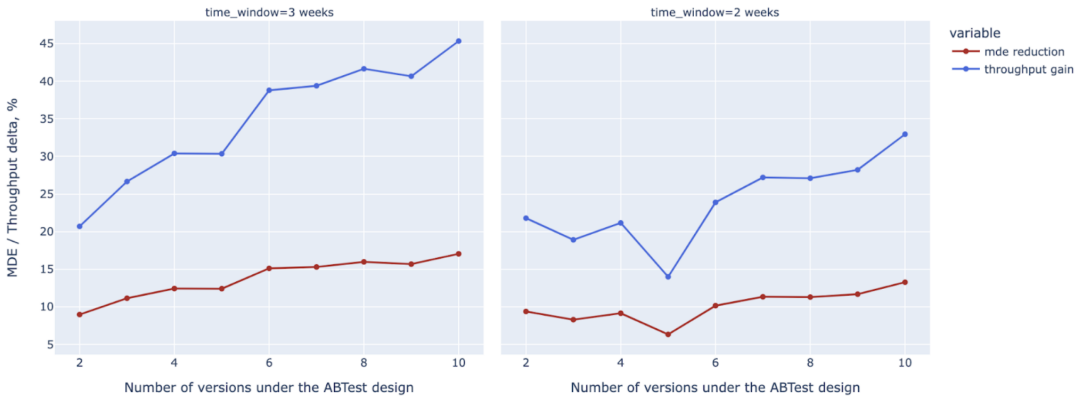

Figure 2: MDE and throughput gains for ACURT compared to the standard user-randomized A/B test: 3 weeks of data (left panel) and 2 weeks of data (right panel).

References

[1] Y. Guo, D. Coey, M. Konutgan, W. Li, C. Schoener, and M. Goldman. Machine learning for variance reduction in online experiments. *Advances in Neural Information Processing Systems, 34, 8637-8648*, 2021.

[2] I. Kabiljo, B. Karrer, M. Pundir, S. Pupyrev, and A. Shalita. Social hash partitioner: a scalable distributed hypergraph partitioner. *Proc. VLDB Endow.*, 10(11):1418–1429, aug 2017.

[3] B. Karrer, L. Shi, M. Bhole, M. Goldman, T. Palmer, C. Gelman, M. Konutgan, and F. Sun. Network experimentation at scale. *In Proceedings of the 27th acm sigkdd conference on knowledge discovery & data mining, pp. 3106–3116*, 2021.

[4] X. Li and P. Ding. Rerandomization and regression adjustment. *Journal of the Royal Statistical Society: Series B (Statistical Methodology), 82(1):241–268*, 2020.

[5] W. Lin. Agnostic notes on regression adjustments to experimental data: Reexamining freedman's critique. *Annals of Applied Statistics, 7(1):295–318*, 2013.

[6] J. Ugander, B. Karrer, L. Backstrom, and J. Kleinberg. Graph cluster randomization: Network exposure to multiple universes. *In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '13, pp. 329–337, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450321747. doi: 10.1145/2487575.2487695*, 2013.

[7] C. Zhang, D. Coey, M. Goldman, and B. Karrer. Regression adjustment with synthetic controls in online experiments. *In 2021 Conference on Digital Experimentation at MIT, Parallel Session 4B: Methods IV–Variance Reduction*, 2021.