Open in app ↗

Search

Qing Zhang · Follow

Published in The Airbnb Tech Blog

10 min read · Oct 6, 2022

Listen Share More

Introduction of Airbnb interleaving experimentation framework, usage and approaches to address challenges in our unique business

Qing Zhang, Michelle Du, Reid Andersen, Liwei He

Introduction

When a user searches for a place to stay on Airbnb, we aim to show them the best results possible. Airbnb's relevance team actively works on improving search ranking experience and helps users to find and book listings that match their preference. A/B test is our approach for online assessment. Our business metrics are conversion-focused, and the frequency of guest travel transactions is lower than on other e-commerce platforms. These factors result in insufficient experiment bandwidth given the number of ideas that we want to test and there is considerable demand to develop a more efficient online testing approach.

Interleaving is an online ranking assessment approach [1–3]. In A/B tests, users are split into control and treatment groups. Those who are in each group will be consistently exposed to results from the corresponding ranker. Interleaving, on the other hand, blends the search results from both control and treatment and presents the "interleaved" results to the user (Figure 1). The mechanism enables direct comparison between the two groups by the same user, with which the impact of the treatment ranker can be evaluated by a collection of specifically designed metrics.

There are several challenges in building the framework on both engineering and data science fronts. On the engineering side, we needed to extend our current AB test framework to enable interleaving set up while adding minimum overhead to the ML engineers. Additionally, our search infrastructure is designed for single request search and required significant extension to support interleaving functionality. On the data science side, we designed user event attribution logic that' key to the effectiveness of metrics.

In 2021, we built the interleaving experimentation framework and integrated it in our experiment process and reached a 50x sensitivity in the development of our search ranking algorithm. Further validation confirms high agreement with A/B tests. We have been using interleaving for a wide range of tasks such as ranker assessment, hyperparameter tuning as well as evaluating infra-level changes. The system design and learnings detailed in this blog post should benefit readers looking to improve their experimentation agility.

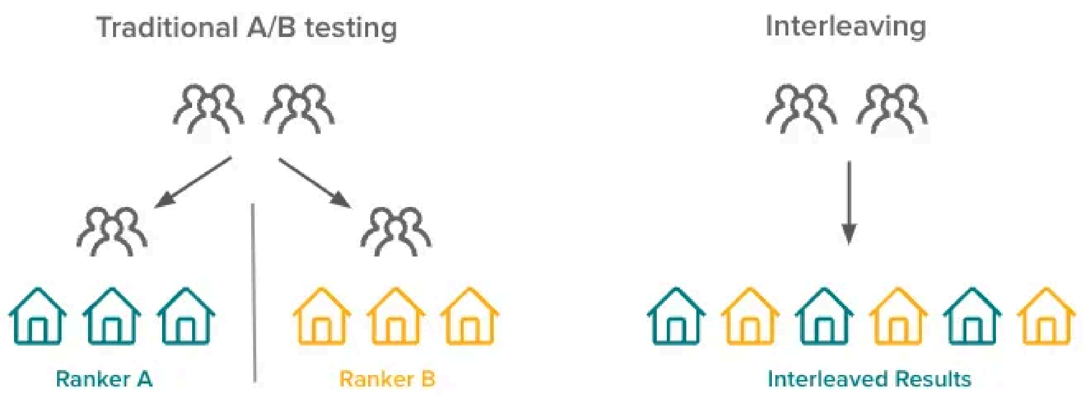

Figure 1: An illustration of A/B testing v.s. Interleaving. In traditional A/B tests, users are split into two groups and exposed to two different rankers. In Interleaving, each user is presented with the blended results from two rankers.

Search Ranking Experimentation Procedure

With interleaving, Airbnb search ranking experimentation uses a three phase procedure for faster experimentation (Figure 2). First, we run standard offline evaluation on the ranker with NDCG (normalized discounted cumulative gain). Rankers with reasonable results move on to online evaluation with interleaving. The ones that get promising results go on for the A/B test.

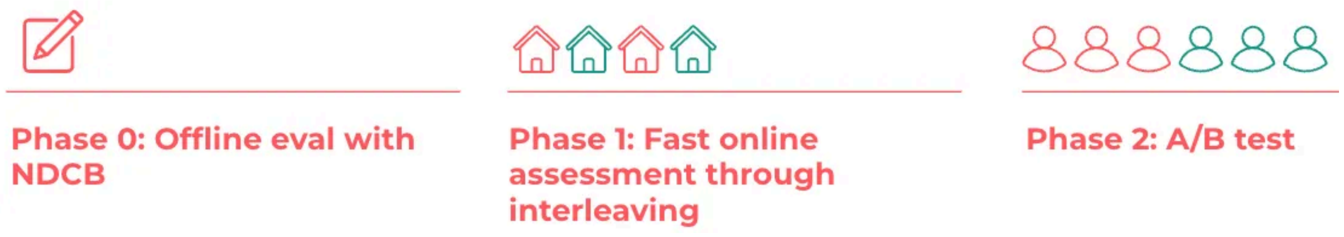
Phase 0: Offline eval with NDCB
Phase 1: Fast online assessment through interleaving
Phase 2: A/B test

Figure 2: Ranking experimentation procedure. We use interleaving to get preliminary online results in order to enable fast iteration

Currently, we split our search traffic into two portions, and use the vast majority for regular A/B tests and remaining for interleaving experiments. We divide the interleaving traffic into buckets (called interleaving lanes) and each lane is used for one interleaving experiment. Each interleaving experiment takes up about 6% of regular A/B test traffic, and one-third of running length. We achieve a 50x speedup over an A/B test given the same amount of traffic. The team now has the luxury to test out multiple variations of the idea in a short time frame and identify the promising routes to move forward.

Airbnb Interleaving Framework

The interleaving framework controls the experimentation traffic and generates interleaved results to return to the user as illustrated in Figure 3. Specifically, for users who are subject to interleaving, the system creates parallel search requests that correspond to control and treatment rankers and produce responses. The results generation component blends the two responses with team drafting algorithms, returns the final response to the user, and creates logging. A suite of metrics were designed to measure impact.

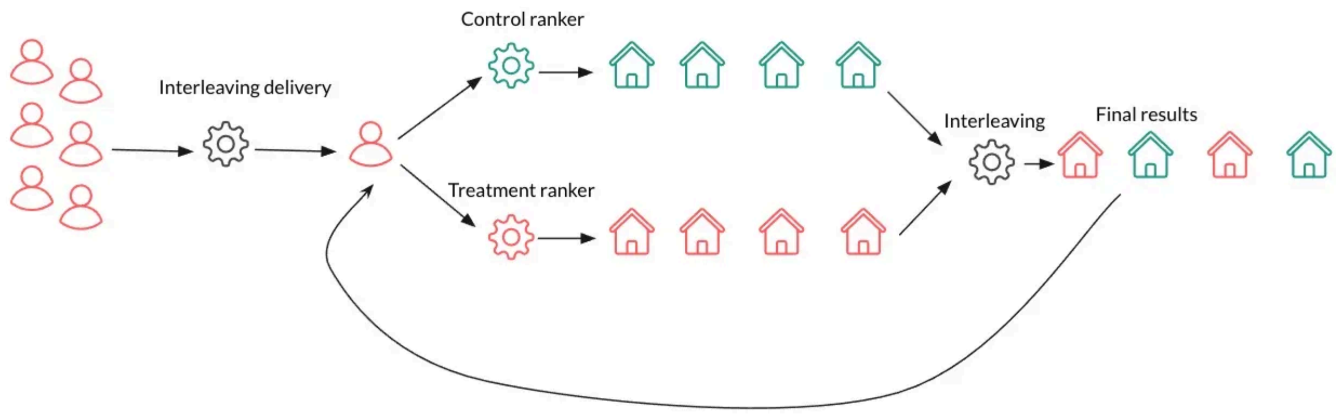

Figure 3: Interleaving system overview. The interleaving framework controls the experimentation traffic and generates interleaved results to return to the user

Team Drafting and Competitive Pairs

The framework employs the *team drafting algorithm* to “interleave” the results from control and treatment (we call them teams). For the purpose of generalizability, we demonstrate the drafting process with two teams A and B. The steps of the algorithm are as follows:

1 Flip a coin to determine if team A goes first

2 Start with an empty merged list. Repeat the following step until desired size is reached,

2. 1 From each of the two rankers A and B take the highest-ranked result that has not yet been selected (say listing a from ranker A and e from ranker B).

2.2 If the two listings are different, then select listings a and e, with assigned a to A and e assigned B. We will call (a, e) a *competitive pair*. Add the pair to the merged list with the order decided in Step 1

2.3 If the two listings are the same, then select that listing and do not assign it to either team. Figure 4 demonstrates the process.

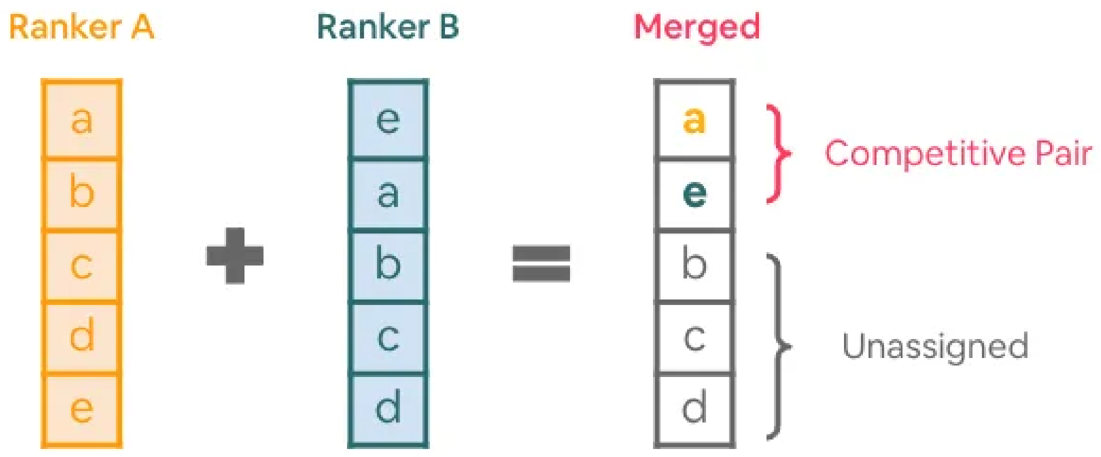

Figure 4: Team drafting example with competitive pair explained. Here we assume that team A goes first based on coin flip.

The *team drafting algorithm* enables us to measure user preference in a fair way. For each request we flip a coin to decide which team (control or treatment) has the priority in the ordering of a *competitive pair*. This means that position bias is minimized as listings from each team are ranked above the one from the other team in the competitive pair half of the time.

Creating *competitive pairs* makes variance reduction (a procedure to speed up experimentation by increasing the precision of the point estimates) more intuitive, since it deduplicates items with the same rank and only assigns scores to the impression of competitive pairs instead of to each impression. In the example in Figure 4, the comparison between ranker A and ranker B reduces to a referendum on whether *a* is better than *e*. Leaving the other results unassigned improves the sensitivity in this case. In an extreme case where two rankers produce lists with exactly the same order, traditional interleaving would still associate clicks to teams and add noise to the result; while with competitive pairs, the entire search query can be ignored since the preference is exactly zero. This allows us to focus on the real difference with sensitivity improvement.

Furthermore, competitive pairs enable us to allocate credits to various user activities downstream much more easily. Again unlike traditional interleaving, which mostly assigns credits for clicks [3–5], we assign credits by bookings, which is a downstream activity. The flexibility in credit association has empowered us to design complicated metrics without having to rely on click signals. For example, we are able to define metrics that measure the booking wins over competition with certain types of listings (e.g. new listings) in the pairs. This enabled us to further understand whether changes to the ranking of a specific category of listings played its role in interleaving overall.

To determine a winning ranker in our interleaving approach, we compare the *preference margin* (margin of victory for the winning team) on target events and apply a 1-sample t-test over it to obtain the p-value. Validation studies confirmed that our framework produces results that are both reliable and robust — with a consistently low false positive rate, and minimum carryover effect between experiments.

Attribution

Attribution logic is a key component of our measurement framework. As mentioned earlier, a typical scenario that is more unique to Airbnb compared to cases like Web search or streaming sites is that our guests can issue multiple search requests before booking, and the listing they book may have been viewed or clicked multiple times when owned by different interleaving teams, which is different from use cases where the primary goal is click-based conversion.

Let's use a toy example to demonstrate the concept. As shown in Figure 5, the guest clicked the booked listing 3 times with each ranker having the listing on their team multiple times (2 times on team A, 1 time on team B) throughout the search journey. For this single guest alone, we see how the different attribution methods can end up with different conclusions:

- If we attribute the booking to the team when it was first clicked, we should assign it to team B and declare team B as the winner for this guest;
- If we attribute the booking to the team when it was last clicked, we should assign it to team A and declare team A as the winner for the guest;
- If we attribute the booking every time it was clicked, we should assign it twice to team A and once to team B, and end up declaring team A being the winner for the guest.

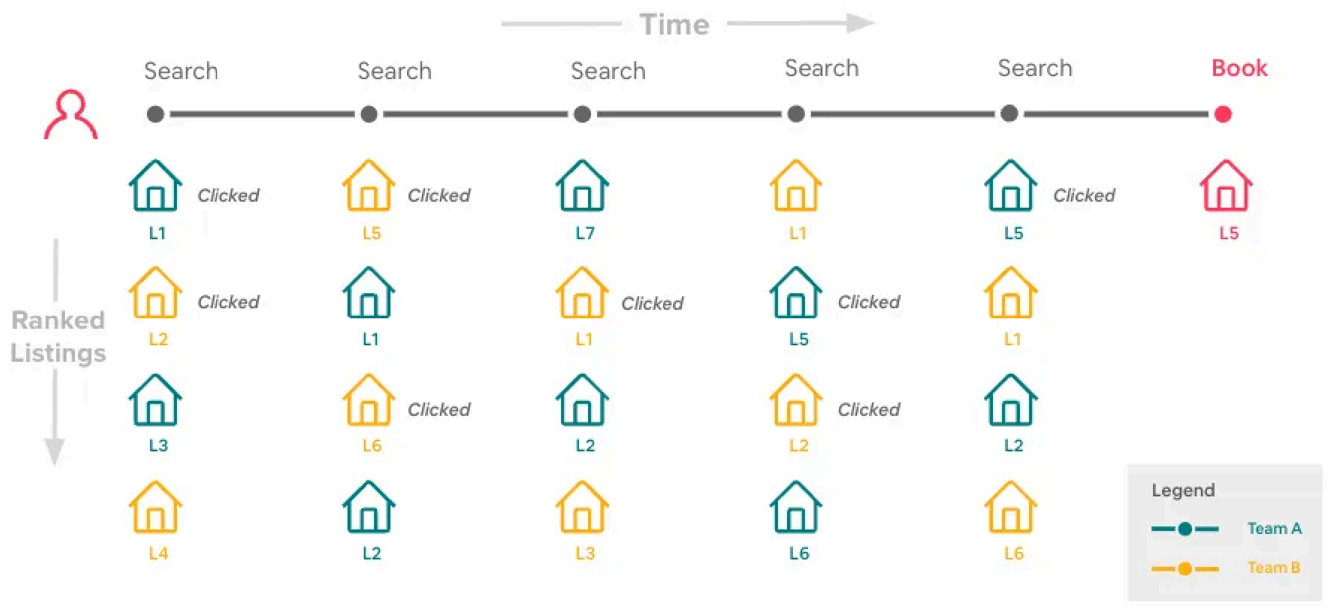

Figure 5: A simplified example of guest journey. The guest emits multiple searches and views the booked listing multiple times before finally making a booking.

We created multiple attribution logic variations and evaluated them on a wide collection of interleaving experiments that also had A/B runs as "ground truth". We set our primary metric to be the one that has best alignment between interleaving and A/B tests.

Alignment with A/B tests

To further evaluate the consistency between interleaving and A/B tests, we tracked eligible interleaving and A/B pairs and confirmed that the two are consistent with each other 82% of the time (Figure 6). The experiments are also highly sensitive as noted in previous work from other companies like Netflix. To provide a concrete example, we have a ranker that randomly picks a listing in the top 300 results and inserts it to the top slot. It takes interleaving only 0.5% of the A/B running time and 4% of A/B traffic to get to the same conclusion as its corresponding A/B test.

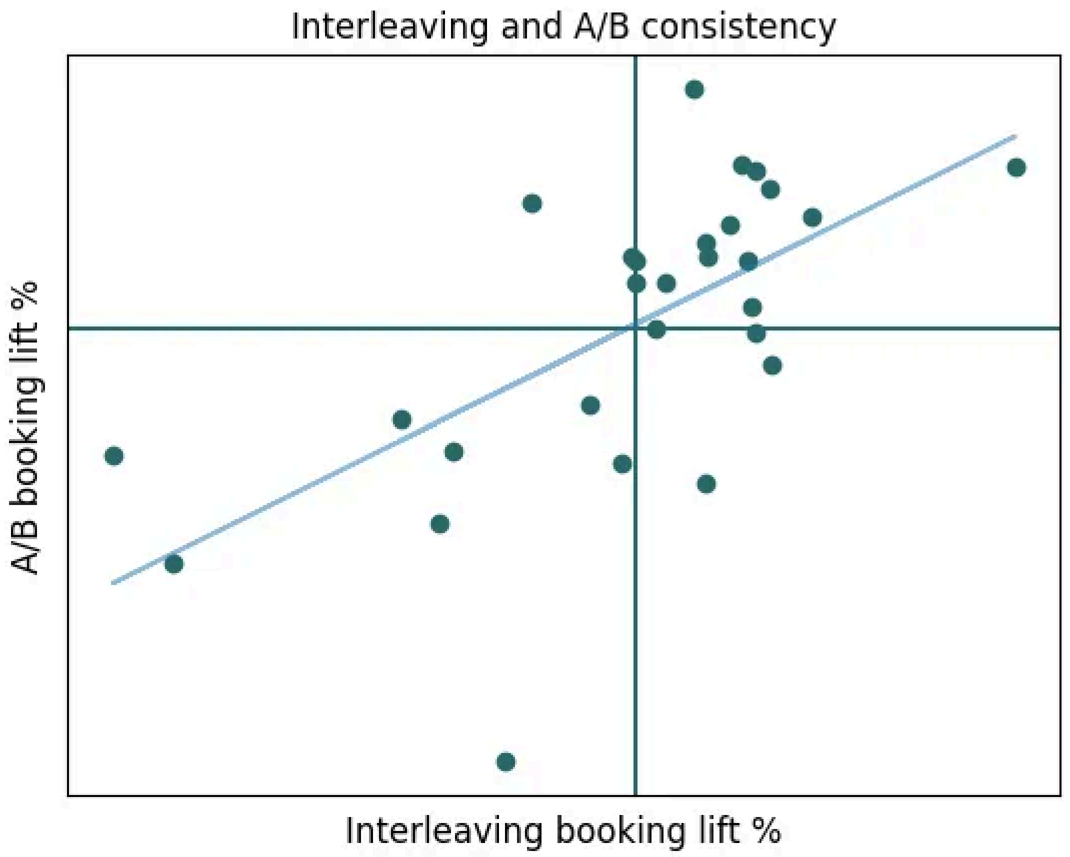

Figure 6: Interleaving and A/B consistency. We tracked eligible interleaving and A/B pairs and the results demonstrate that the two are consistent with each other 82% of the time

In most cases where interleaving turned out to be inconsistent with traditional A/B testing, we found that the reason was set-level optimization. For example, one ranker relies on a model to determine how strongly it will demote listings with high host rejection probability and the model is the booking probability given the current page. Interleaving breaks this assumption and leads to inaccurate results. Based on our learnings, we advise that rankers that involve set-level optimization should use interleaving on a case by case basis.

Conclusion

Search ranking quality is key for an Airbnb user to find their desired accommodation and iterating on the algorithm efficiently is our top priority. The interleaving experimentation framework tackles our problem of limited A/B test bandwidth and provides up to 50x speed up on the search ranking algorithm iteration. We conducted comprehensive validation which demonstrated that interleaving is highly robust and has strong correlation with traditional A/B. Interleaving is currently part of our experimentation procedure, and is the main evaluation technique before the A/B test. The framework opens a new field of online experimentation for the company and can be applied to other product surfaces such as recommendations.

Interested in working at Airbnb? Check out our open roles HERE.

Acknowledgments

We would like to thank Aaron Yin for the guidance on the implementations of algorithms and metrics, Xin Liu for continuously advising us on optimizing and extending the framework to support more use cases, Chunhow Tan for valuable suggestions on improving the computational efficiency of interleaving metrics and Tatiana Xifara for advice on experiment delivery design.

The system won't be possible without the support from our search backend team, especially Yangbo Zhu, Eric Wu, Varun Sharma and Soumyadip (Soumo) Banerjee. We benefit tremendously from their design advice and close collaboration on the operations.

We would also like to thank Alex Deng, Huiji Gao and Sanjeev Katariya for valuable feedback on the interleaving and this article.

References

[1] JOACHIMS, T. Optimizing Search Engines Using Clickthrough Data. In Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD). ACM, New York, NY, 132–142. 2002.

[2] JOACHIMS, T. Evaluating Retrieval Performance using Clickthrough Data. In Text Mining, J. Franke, G. Nakhaeizadeh, and I. Renz, Eds., Physica/Springer Verlag, 79–96. 2003.

[3] RADLINSKI, F., KURUP, M., AND JOACHIMS, T. How does clickthrough data reflect retrieval quality. In Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM'08). ACM, New York, NY, 43–52. 2008.

[4] Radlinski, Filip, and Nick Craswell. "Optimized interleaving for online retrieval evaluation." Proceedings of the sixth ACM international conference on Web search and data mining. 2013.

[5] Hofmann, Katja, Shimon Whiteson, and Maarten De Rijke. "A probabilistic method for inferring preferences from clicks." Proceedings of the 20th ACM international conference on Information and knowledge management. 2011.