Sensitive and Scalable Online Evaluation with Theoretical Guarantees

Harrie Oosterhuis
University of Amsterdam
Amsterdam, The Netherlands
oosterhuis@uva.nl

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
derijke@uva.nl

ABSTRACT

Multileaved comparison methods generalize interleaved comparison methods to provide a scalable approach for comparing ranking systems based on regular user interactions. Such methods enable the increasingly rapid research and development of search engines. However, existing multileaved comparison methods that provide reliable outcomes do so by degrading the user experience during evaluation. Conversely, current multileaved comparison methods that maintain the user experience cannot guarantee correctness. Our contribution is two-fold. First, we propose a theoretical framework for systematically comparing multileaved comparison methods using the notions of *considerateness*, which concerns maintaining the user experience, and *fidelity*, which concerns reliable correct outcomes. Second, we introduce a novel multileaved comparison method, Pairwise Preference Multileaving (PPM), that performs comparisons based on document-pair preferences, and prove that it is *considerate* and has *fidelity*. We show empirically that, compared to previous multileaved comparison methods, PPM is more *sensitive* to user preferences and *scalable* with the number of rankers being compared.

1 INTRODUCTION

Evaluation is of tremendous importance to the development of modern search engines. Any proposed change to the system should be verified to ensure it is a true improvement. Online approaches to evaluation aim to measure the actual utility of an Information Retrieval (IR) system in a natural usage environment [14]. Interleaved comparison methods are a within-subject setup for online experimentation in IR. For interleaved comparison, two experimental conditions ("control" and "treatment") are typical. Recently, multileaved comparisons have been introduced for the purpose of efficiently comparing large numbers of rankers [2, 27]. These multileaved comparison methods were introduced as an extension to interleaving and the majority are directly derived from their interleaving counterparts [27, 28]. The effectiveness of these methods has thus far only been measured using simulated experiments on public datasets. While this gives some insight into the general *sensitivity* of a method, there is no work that assesses under what

circumstances these methods provide correct outcomes and when they break. Without knowledge of the theoretical properties of multileaved comparison methods we are unable to identify when their outcomes are reliable.

In prior work on interleaved comparison methods a theoretical framework has been introduced that provides explicit requirements that an interleaved comparison method should satisfy [13]. We take this approach as our starting point and adapt and extend it to the setting of multileaved comparison methods. Specifically, the notion of *fidelity* is central to Hofmann et al. [13]'s previous work; Section 3 describes the framework with its requirements of *fidelity*. In the setting of multileaved comparison methods, this means that a multileaved comparison method should always recognize an unambiguous winner of a comparison. We also introduce a second notion, *considerateness*, which says that a comparison method should not degrade the user experience, e.g., by allowing all possible permutations of documents to be shown to the user. In this paper we examine all existing multileaved comparison methods and find that none satisfy both the *considerateness* and *fidelity* requirements. In other words, no existing multileaved comparison method is correct without sacrificing the user experience.

To address this gap, we propose a novel multileaved comparison method, Pairwise Preference Multileaving (PPM). PPM differs from existing multileaved comparison methods as its comparisons are based on inferred pairwise document preferences, whereas existing multileaved comparison methods either use some form of document assignment [27, 28] or click credit functions [2, 27]. We prove that PPM meets both the *considerateness* and the *fidelity* requirements, thus PPM guarantees correct winners in unambiguous cases while maintaining the user experience at all times. Furthermore, we show empirically that PPM is more *sensitive* than existing methods, i.e., it makes fewer errors in the preferences it finds. Finally, unlike other multileaved comparison methods, PPM is computationally efficient and *scalable*, meaning that it maintains most of its *sensitivity* as the number of rankers in a comparison increases.

In this paper we address the following research questions:

RQ1 Does PPM meet the *fidelity* and *considerateness* requirements?
RQ2 Is PPM more sensitive than existing methods when comparing multiple rankers?

To summarize, our contributions are:

(1) A theoretical framework for comparing multileaved comparison methods;
(2) A comparison of all existing multileaved comparison methods in terms of *considerateness*, *fidelity* and *sensitivity*;
(3) A novel multileaved comparison method that is *considerate* and has *fidelity* and is more *sensitive* than existing methods.

2 RELATED WORK

Evaluation of information retrieval systems is a core problem in IR. Two types of approach are common to designing reliable methods for measuring an IR system's effectiveness. Offline approaches such as the Cranfield paradigm [26] are effective for measuring topical relevance, but have difficulty taking into account contextual information including the user's current situation, fast changing information needs, and past interaction history with the system [14]. In contrast, online approaches to evaluation aim to measure the actual utility of an IR system in a natural usage environment. User feedback in online evaluation is usually implicit, in the form of clicks, dwell time, etc.

By far the most common type of controlled experiment on the web is A/B testing [19, 20]. This is a classic between-subject experiment, where each subject is exposed to one of two conditions, *control*—the current system—and *treatment*—an experimental system that is assumed to outperform the control.

An alternative experiment design uses a within-subject setup, where all study participants are exposed to both experimental conditions. Interleaved comparisons [15, 25] have been developed specifically for online experimentation in IR. Interleaved comparison methods have two main ingredients. First, a method for constructing interleaved result lists specifies how to select documents from the original rankings ("control" and "treatment"). Second, a method for inferring comparison outcomes based on observed user interactions with the interleaved result list. Because of their within-subject nature, interleaved comparisons can be up to two orders of magnitude more efficient than A/B tests in effective sample size for studies of comparable dependent variables [4].

For interleaved comparisons, two experimental conditions are typical. Extensions to multiple conditions have been introduced by Schuth et al. [27]. Such *multileaved* comparisons are an efficient online evaluation method for comparing multiple rankers simultaneously. Similar to interleaved comparison methods [12, 17, 24, 25], a multileaved comparison infers preferences between rankers. Interleaved comparisons do this by presenting users with interleaved result lists; these represent two rankers in such a way that a preference between the two can be inferred from clicks on their documents. Similarly, for multileaved comparisons multileaved result lists are created that allow more than two rankers to be represented in the result list. As a consequence, multileaved comparisons can infer preferences between multiple rankers from a single click. Due to this property multileaved comparisons require far fewer interactions than interleaved comparisons to achieve the same accuracy when multiple rankers are involved [27, 28].

The general approach for every multileaved comparison method is described in Algorithm 1; here, a comparison of a set of rankers $\mathcal{R}$ is performed over T user interactions. After the user submits a query q to the system (Line 4), a ranking $\mathbf{l}_i$ is generated for each ranker $\mathbf{r}_i$ in $\mathcal{R}$ (Line 6). These rankings are then combined into a single result list by the multileaving method (Line 7); we refer to the resulting list $\mathbf{m}$ as the multileaved result list. In theory a multileaved result list could contain the entire document set, however in practice a length k is chosen beforehand, since users generally only view a restricted number of result pages. This multileaved result list is presented to the user who has the choice to interact with it or

---

Algorithm 1 General pipeline for multileaved comparisons.

1: **Input**: set of rankers $\mathcal{R}$, documents D, no. of timesteps T.
2: $P \leftarrow \mathbf{0}$ // *initialize $|\mathcal{R}| \times |\mathcal{R}|$ preference matrix*
3: **for** $t = 1, \ldots, T$ **do**
4:    $q_t \leftarrow wait\_for\_user()$ // *receive query from user*
5:    **for** $i = 1, \ldots, |\mathcal{R}|$ **do**
6:      $\mathbf{l}_i \leftarrow \mathbf{r}_i(q, D)$ // *create ranking for query per ranker*
7:    $\mathbf{m}_t \leftarrow combine\_lists(\mathbf{l}_1, \ldots, \mathbf{l}_R)$ // *combine into multileaved list*
8:    $\mathbf{c} \leftarrow display(\mathbf{m}_t)$ // *display to user and record interactions*
9:    **for** $i = 1, \ldots, |\mathcal{R}|$ **do**
10:      **for** $j = 1, \ldots, |\mathcal{R}|$ **do**
11:        $P_{ij} \leftarrow P_{ij} + infer(i, j, \mathbf{c}, \mathbf{m}_t)$ // *infer pref. between rankers*
12: **return** P

---

not. Any interactions are recorded in $\mathbf{c}$ and returned to the system (Line 8). While $\mathbf{c}$ could contain any interaction information [18], in practice multileaved comparison methods only consider clicks. Preferences between the rankers in $\mathcal{R}$ can be inferred from the interactions and the preference matrix P is updated accordingly (Line 11). The method of inference (Line 11) is defined by the multileaved comparison method (Line 7). By aggregating the inferred preferences of many interactions a multileaved comparison method can detect preferences of users between the rankers in $\mathcal{R}$. Thus it provides a method of evaluation without requiring a form of explicit annotation.

By instantiating the general pipeline for multileaved comparisons shown in Algorithm 1, i.e., the combination method at Line 6 and the inference method at Line 11, we obtain a specific multileaved comparison method. We detail all known multileaved comparison methods in Section 4 below.

What we add on top of the work discussed above is a theoretical framework that allows us to assess and compare multileaved comparison methods. In addition, we propose an accurate and scalable multileaved comparison method that is the only one to satisfy the properties specified in our theoretical framework and that also proves to be the most efficient multileaved comparison method in terms of much reduced data requirements.

3 A FRAMEWORK FOR ASSESSING MULTILEAVED COMPARISON METHODS

Before we introduce a novel multileaved comparison method in Section 5, we propose two theoretical requirements for multileaved comparison methods. These theoretical requirements will allow us to assess and compare existing multileaved comparison methods. Specifically, we introduce two theoretical properties: *considerateness* and *fidelity*. These properties guarantee correct outcomes in *unambiguous* cases while always maintaining the user experience. In Section 4 we show that no currently available multileaved comparison method satisfies both properties. This motivates the introduction of a method that satisfies both properties in Section 5.

3.1 Considerateness

Firstly, one of the most important properties of a multileaved comparison method is how **considerate** it is. Since evaluation is done online it is important that the search experience is not substantially

altered [15, 24]. In other words, users should not be obstructed to perform their search tasks during evaluation. As maintaining a user base is at the core of any search engine, methods that potentially degrade the user experience are generally avoided. Therefore, we set the following requirement: the displayed multileaved result list should never show a document d at a rank i if every ranker in $\mathcal{R}$ places it at a lower rank. Writing $r(d, \mathbf{l}_j)$ for the rank of d in the ranking $\mathbf{l}_j$ produced by ranker $\mathbf{r}_j$, this boils down to:

$$\mathbf{m}_i = d \rightarrow \exists \mathbf{r}_j \in \mathcal{R}, r(d, \mathbf{l}_j) \leq i. \tag{1}$$

Requirement 1 guarantees that a document can never be displayed higher than any ranker would. In addition, it guarantees that if all rankers agree on the top n documents, the resulting multileaved result list $\mathbf{m}$ will display the same top n.

3.2 Fidelity

Secondly, the preferences inferred by a multileaved comparison method should correspond with those of the user with respect to retrieval quality, and should be robust to user behavior that is unrelated to retrieval quality [15]. In other words, the preferences found should be correct in terms of ranker quality. However, in many cases the relative quality of rankers is unclear. For that reason we will use the notion of **fidelity** [13] to compare the correctness of a multileaved comparison method. *Fidelity* was introduced by Hofmann et al. [13] and describes two general cases in which the preference between two rankers is unambiguous. To have *fidelity* the expected outcome of a method is required to be correct in all matching cases. However, the original notion of *fidelity* only considers two rankers as it was introduced for interleaved comparison methods, therefore the definition of *fidelity* must be expanded to the multileaved case. First we describe the following concepts:

Uncorrelated clicks. Clicks are considered *uncorrelated* if relevance has no influence on the likelihood that a document is clicked. We write $r(d_i, \mathbf{m})$ for the rank of document d_i in multileaved result list $\mathbf{m}$ and $P(\mathbf{c}_l \mid q, \mathbf{m}_l = d_i)$ for the probability of a click at the rank l at which d_i is displayed: $l = r(d_i, \mathbf{m})$. Then, for a given query q

$$uncorrelated(q) \Leftrightarrow$$
$$\forall l, \forall d_{i,j}, P(\mathbf{c}_l \mid q, \mathbf{m}_l = d_i) = P(\mathbf{c}_l \mid q, \mathbf{m}_l = d_j). \tag{2}$$

Correlated clicks. We consider clicks correlated if there is a positive correlation between document relevance and clicks. However we differ from Hofmann et al. [13] by introducing a variable k that denotes at which rank users stop considering documents. Writing $P(\mathbf{c}_i \mid rel(\mathbf{m}_i, q))$ for the probability of a click at rank i if a document relevant to query q is displayed at this rank, we set

$$correlated(q, k) \Leftrightarrow \forall i \geq k, P(\mathbf{c}_i) = 0$$
$$\wedge \, \forall i < k, P(\mathbf{c}_i \mid rel(\mathbf{m}_i, q)) > P(\mathbf{c}_i \mid \neg rel(\mathbf{m}_i, q)). \tag{3}$$

Thus under correlated clicks a relevant document is more likely to be clicked than a non-relevant one at the same rank, if they appear above rank k.

Pareto domination. Ranker $\mathbf{r}_1$ *Pareto dominates* ranker $\mathbf{r}_2$ if all relevant documents are ranked at least as high by $\mathbf{r}_1$ as by $\mathbf{r}_2$ and $\mathbf{r}_1$ ranks at least one relevant document higher. Writing *rel* for the set of relevant documents that are ranked above k by at least one

ranker, i.e., $rel = \{d \mid rel(d, q) \wedge \exists \mathbf{r}_n \in \mathcal{R}, r(d, \mathbf{l}_n) > k\}$, we require that the following holds for every query q and any rank k:

$$Pareto(\mathbf{r}_i, \mathbf{r}_j, q, k) \Leftrightarrow$$
$$\forall d \in rel, r(d, \mathbf{l}_i) \leq r(d, \mathbf{l}_j) \wedge \exists d \in rel, r(d, \mathbf{l}_i) < r(d, \mathbf{l}_j). \tag{4}$$

Then, *fidelity* for multileaved comparison methods is defined by the following two requirements:

(1) Under uncorrelated clicks the expected outcome may find no preferences between any two rankers in $\mathcal{R}$:

$$\forall q, \forall (\mathbf{r}_i, \mathbf{r}_j) \in \mathcal{R}, uncorrelated(q) \Rightarrow E[P_{ij} \mid q] = 0. \tag{5}$$

(2) Under correlated clicks, a ranker that Pareto dominates all other rankers must win the multileaved comparison in expectation:

$$\forall k, \forall q, \forall \mathbf{r}_i \in \mathcal{R}, \big(correlated(q, k) \wedge$$
$$\forall \mathbf{r}_j \in \mathcal{R}, i \neq j \rightarrow Pareto(\mathbf{r}_i, \mathbf{r}_j, q, k) \big) \tag{6}$$
$$\Rightarrow \big(\forall \mathbf{r}_j \in \mathcal{R}, i \neq j \rightarrow E[P_{ij} \mid q] > 0 \big) .$$

Note that for the case where $|\mathcal{R}| = 2$ and if only $k = |D|$ is considered, these requirements are the same as for interleaved comparison methods [13]. The k parameter was added to allow for *fidelity* in *considerate* methods, since it is impossible to detect preferences at ranks that users never consider without breaking the *considerateness* requirement. We argue that differences at ranks that users are not expected to observe should not affect comparison outcomes. *Fidelity* is important for a multileaved comparison method as it ensures that an unambiguous winner is expected to be identified. Additionally, the first requirement ensures unbiasedness when clicks are unaffected by relevancy.

3.3 Additional properties

In addition to the two theoretical properties listed above, considerateness and fidelity, we also scrutinize multileaved comparison methods to determine whether they accurately find preferences between all rankers in $\mathcal{R}$ and minimize the number of user impressions required do so. This empirical property is commonly known as **sensitivity** [13, 27]. In Section 6 we describe experiments that are aimed at comparing the sensitivity of multileaved comparison methods. Here, two aspects of every comparison are considered: the level of error at which a method converges and the number of impressions required to reach that level. Thus, an interleaved comparison method that learns faster initially but does not reach the same final level of error is deemed worse.

4 AN ASSESSMENT OF EXISTING MULTILEAVED COMPARISON METHODS

We briefly examine all existing multileaved comparison methods to determine whether they meet the *considerateness* and *fidelity* requirements. An investigation of the empirical sensitivity requirement is postponed until Section 6 and 7.

4.1 Team Draft Multileaving

Team-Draft Multileaving (TDM) was introduced by Schuth et al. [27] and is based on the previously proposed Team Draft Interleaving (TDI) [25]. Both methods are inspired by how team assignments are often chosen for friendly sport matches. The multileaved result

list is created by sequentially sampling rankers without replacement; the first sampled ranker places their top document at the first position of the multileaved list. Subsequently, the next sampled ranker adds their top pick of the remaining documents. When all rankers have been sampled, the process is continued by sampling from the entire set of rankers again. The method is stops when all documents have been added. When a document is clicked, TDM assigns the click to the ranker that contributed the document. For each impression binary preferences are inferred by comparing the number of clicks each ranker received.

It is clear that TDM is *considerate* since each added document is the top pick of at least one ranker. However, TDM does not meet the fidelity requirements. This is unsurprising as previous work has proven that TDI does not meet these requirements [12, 13, 24]. Since TDI is identical to TDM when the number of rankers is $|\mathcal{R}| = 2$, TDM does not have *fidelity* either.

4.2 Optimized Multileaving

Optimized Multileaving (OM) was proposed by Schuth et al. [27] and serves as an extension of Optimized Interleaving (OI) introduced by Radlinski and Craswell [24]. The allowed multileaved result lists of OM are created by sampling rankers with replacement at each iteration and adding the top document of the sampled ranker. However, the probability that a multileaved result list is shown is not determined by the generative process. Instead, for a chosen credit function OM performs an optimization that computes a probability for each multileaved result list so that the expected outcome is unbiased and sensitive to correct preferences.

All of the allowed multileaved result lists of OM meet the *considerateness* requirement, and in theory instantiations of OM could have *fidelity*. However, in practice OM does not meet the *fidelity* requirements. There are two main reasons for this. First, it is not guaranteed that a solution exists for the optimization that OM performs. For the interleaving case this was proven empirically when $k = 10$ [24]. However, this approach does not scale to any number of rankers. Secondly, unlike OI, OM allows more result lists than can be computed in a feasible amount of time. Consider the top k of all possible multileaved result lists; in the worst case this produces $|\mathcal{R}|^k$ lists. Computing all lists for a large value of $|\mathcal{R}|$ and performing linear constraint optimization over them is simply not feasible. As a solution, Schuth et al. [27] propose a method that samples from the allowed multileaved result lists and relaxes constraints when there is no exact solution. Consequently, there is no guarantee that this method does not introduce bias. Together, these two reasons show that the *fidelity* of OI does not imply fidelity of OM. It also shows that OM is computationally very costly.

4.3 Probabilistic Multileaving

Probabilistic Multileaving (PM) [28] is an extension of Probabilistic Interleaving (PI) [12], which was designed to solve the flaws of TDI. Unlike the previous methods, PM considers every ranker as a distribution over documents, which is created by applying a soft-max to each of them. A multileaved result list is created by sampling a ranker with replacement at each iteration and sampling a document from the ranker that was selected. After the sampled document has been added, all rankers are renormalized to account for the

removed document. During inference PM credits every ranker the expected number of clicked documents that were assigned to them. This is done by marginalizing over the possible ways the list could have been constructed by PM. A benefit of this approach is that it allows for comparisons on historical data [12, 13].

A big disadvantage of PM is that it allows any possible ranking to be shown, albeit not with uniform probabilities. This is a big deterrent for the usage of PM in operational settings. Furthermore, it also means that PM does not meet the *considerateness* requirement. On the other hand, PM does meet the *fidelity* requirements, the proof for this follows from the fact that every ranker is equally likely to add a document at each location in the ranking. Moreover, if multiple rankers want to place the same document somewhere they have to share the resulting credits.[1] Similar to OM, PM becomes infeasible to compute for a large number of rankers $|\mathcal{R}|$; the number of assignments in the worst case is $|R|^k$. Fortunately, PM inference can be estimated by sampling assignments in a way that maintains *fidelity* [22, 28].

4.4 Sample Only Scored Multileaving

Sample-Scored-Only Multileaving (SOSM) was introduced by Brost et al. [2] in an attempt to create a more scalable multileaved comparison method. It is the only existing multileaved comparison method that does not have an interleaved comparison counterpart. SOSM attempts to increase *sensitivity* by ignoring all non-sampled documents during inference. Thus, at each impression a ranker receives credits according to how it ranks the documents that were sampled for the displayed multileaved result list of size k. The preferences at each impression are made binary before being added to the mean. SOSM creates multileaved result lists following the same procedure as TDM, a choice that seems arbitrary.

SOSM meets the *considerateness* requirements for the same reason TDM does. However, SOSM does not meet the fidelity requirement. We can prove this by providing an example where preferences are found under uncorrelated clicks. Consider the two documents A and B and the three rankers with the following three rankings:

$$\mathbf{l}_1 = AB, \qquad \mathbf{l}_2 = \mathbf{l}_3 = BA.$$

The first requirement of fidelity states that under uncorrelated clicks no preferences may be found in expectation. Uncorrelated clicks are unconditioned on document relevance (Equation 2); however, it is possible that they display position bias [32]. Thus the probability of a click at the first rank may be greater than at the second:

$$P(\mathbf{c}_1 \mid q) > P(\mathbf{c}_2 \mid q).$$

Under position biased clicks the expected outcome for each possible multileaved result list is not zero. For instance, the following preferences are expected:

$$E[P_{12} \mid \mathbf{m} = AB] > 0,$$
$$E[P_{12} \mid \mathbf{m} = BA] < 0,$$
$$E[P_{12} \mid \mathbf{m} = AB] = -E[P_{12} \mid \mathbf{m} = BA].$$

Since SOSM creates multileaved result lists following the TDM procedure the probability $P(\mathbf{m} = BA)$ is twice as high as $P(\mathbf{m} = AB)$.

---

[1]Brost et al. [2] proved that if the preferences at each impression are made binary the *fidelity* of PM is lost.

Table 1: Overview of multileaved comparison methods and whether they meet the *considerateness* and *fidelity* requirements.

	Considerateness	Fidelity	Source
TDM	✓		[27]
OM	✓		[27]
PM		✓	[28]
SOSM	✓		[2]
PPM	✓	✓	this paper

As a consequence, the expected preference is biased against the first ranker:

$$E[P_{12}] < 0.$$

Hence, SOSM does not have *fidelity*. This outcome seems to stem from a disconnect between how multileaved results lists are created and how preferences are inferred.

To conclude this section, Table 1 provides an overview of our findings thus far, i.e., the theoretical requirements that each multileaved comparison method satisfies; we have also included PPM, the multileaved comparison method that we will introduce below.

5 A NOVEL MULTILEAVED COMPARISON METHOD

The previously described multileaved comparison methods are based around direct credit assignment, i.e., credit functions are based on single documents. In contrast, we introduce a method that estimates differences based on pairwise document preferences. We prove that this novel method is the only multileaved comparison method that meets the *considerateness* and *fidelity* requirements set out in Section 3.

The multileaved comparison method that we introduce is Pairwise Preference Multileaving (PPM). It infers pairwise preferences between documents from clicks and bases comparisons on the agreement of rankers with the inferred preferences. PPM is based on the assumption that a clicked document is preferred to: (a) all of the unclicked documents above it; (b) the next unclicked document. These assumptions are long-established [16] and form the basis of pairwise Learning to Rank (LTR) [15].

We write $\mathbf{c}_{r(d_i,\mathbf{m})}$ for a click on document d_i displayed in multileaved result list $\mathbf{m}$ at the rank $r(d_i, \mathbf{m})$. For a document pair (d_i, d_j), a click $\mathbf{c}_{r(d_i,\mathbf{m})}$ infers a preference as follows:

$$\mathbf{c}_{r(d_i,\mathbf{m})} \wedge \neg \mathbf{c}_{r(d_j,\mathbf{m})} \wedge \big(\exists i, (c_i \wedge r(d_j, \mathbf{m}) < i) \vee \mathbf{c}_{r(d_j,\mathbf{m})-1}\big) \\ \Leftrightarrow d_i >_c d_j. \tag{7}$$

In addition, the preference of a ranker $\mathbf{r}$ is denoted by $d_i >_{\mathbf{r}} d_j$. Pairwise preferences also form the basis for Preference-Based Balanced Interleaving (PBI) introduced by He et al. [11]. However, previous work has shown that PBI does not meet the *fidelity* requirements [13]. Therefore, we do not use PBI as a starting point for PPM. Instead, PPM is derived directly from the *considerateness* and *fidelity* requirements. Consequently, PPM constructs multileaved result lists inherently differently and its inference method has *fidelity*, in contrast with PBI.

Algorithm 2 Multileaved result list construction for PPM.

1: **Input**: set of rankers $\mathcal{R}$, rankings $\{\mathbf{l}\}$, documents D.
2: $\mathbf{m} \leftarrow []$ // initialize empty multileaving
3: **for** $n = 1, \ldots, |D|$ **do**
4:     $\hat{\Omega}_n \leftarrow \Omega(n, \mathcal{R}, D) \setminus \mathbf{m}$ // choice set of remaining documents
5:     $d \leftarrow uniform\_sample(\hat{\Omega}_n)$ // uniformly sample next document
6:     $\mathbf{m} \leftarrow append(\mathbf{m}, d)$ // add sampled document to multileaving
7: **return** $\mathbf{m}$

Algorithm 3 Preference inference for PPM.

1: **Input**: rankers $\mathcal{R}$, rankings $\{\mathbf{l}\}$, documents D, multileaved result list $\mathbf{m}$, clicks $\mathbf{c}$.
2: $P \leftarrow \mathbf{0}$ // preference matrix of $|\mathcal{R}| \times |\mathcal{R}|$
3: **for** $(d_i, d_j) \in \{(d_i, d_j) \mid d_i >_{\mathbf{c}} d_j\}$ **do**
4:     **if** $\underline{r}(i, j, \mathbf{m}) \geq \bar{r}(i, j)$ **then**
5:        $w \leftarrow 1$ // variable to store $P(\underline{r}(i, j, \mathbf{m}) \geq \bar{r}(i, j))$
6:        $min\_x \leftarrow \min_{d \in \{d_i, d_j\}} \min_{\mathbf{r}_n \in \mathcal{R}} r(d, \mathbf{l}_n)$
7:        **for** $x = min\_x, \ldots, \bar{r}(i, j) - 1$ **do**
8:           $w \leftarrow w \cdot (1 - (|\Omega(x, \mathcal{R}, D)| - x - 1)^{-1})$
9:        **for** $n = 1, \ldots, |R|$ **do**
10:        **for** $m = 1, \ldots, |R|$ **do**
11:           **if** $d_i >_{\mathbf{r}_n} d_j \wedge n \neq m$ **then**
12:             $P_{nm} \leftarrow P_{nm} + w^{-1}$ // result of scoring function ϕ
13:           **else if** $n \neq m$ **then**
14:             $P_{nm} \leftarrow P_{nm} - w^{-1}$
15: **return** P

When constructing a multileaved result list $\mathbf{m}$ we want to be able to infer unbiased preferences while simultaneously being *considerate*. Thus, with the requirement for *considerateness* in mind we define a choice set as:

$$\Omega(i, \mathcal{R}, D) = \{d \mid d \in D \wedge \exists \mathbf{r}_j \in \mathcal{R}, r(d, \mathbf{l}_j) \leq i\}. \tag{8}$$

This definition is chosen so that any document in $\Omega(i, \mathcal{R}, D)$ can be placed at rank i without breaking the *obstruction* requirement (Equation 1). The multileaving method of PPM is described in Algorithm 2. The approach is straightforward: at each rank n the set of documents $\hat{\Omega}_n$ is determined (Line 4). This set of documents is $\Omega(n, \mathcal{R}, D)$ with the previously added documents removed to avoid document repetition. Then, the next document is sampled uniformly from $\hat{\Omega}_n$ (Line 5), thus every document in $\hat{\Omega}_n$ has a probability:

$$\frac{1}{|\Omega(n, \mathcal{R}, D)| - n + 1} \tag{9}$$

of being placed at position n (Line 6). Since $\hat{\Omega}_n \subseteq \Omega(n, \mathcal{R}, D)$ the resulting $\mathbf{m}$ is guaranteed to be *considerate*.

While the multileaved result list creation method used by PPM is simple, its preference inference method is more complicated as it has to meet the *fidelity* requirements. First, the preference found between a ranker $\mathbf{r}_n$ and $\mathbf{r}_m$ from a single interaction $\mathbf{c}$ is determined by:

$$P_{nm} = \sum_{d_i >_{\mathbf{c}} d_j} \phi(d_i, d_j, \mathbf{r}_n, \mathbf{m}, \mathcal{R}) - \phi(d_i, d_j, \mathbf{r}_m, \mathbf{m}, \mathcal{R}), \tag{10}$$

which sums over all document pairs (d_i, d_j) where interaction $\mathbf{c}$ inferred a preference. Before the scoring function ϕ can be defined we introduce the following function:

$$\bar{r}(i, j, \mathcal{R}) = \max_{d \in \{d_i, d_j\}} \min_{\mathbf{r}_n \in \mathcal{R}} r(d, \mathbf{l}_n). \tag{11}$$

For succinctness we will note $\bar{r}(i, j) = \bar{r}(i, j, \mathcal{R})$. Here, $\bar{r}(i, j)$ provides the highest rank at which both documents d_i and d_j can appear in $\mathbf{m}$. Position $\bar{r}(i, j)$ is important to the document pair (d_i, d_j), since if both documents are in the remaining documents $\hat{\Omega}_{\bar{r}(i,j)}$, then the rest of the multileaved result list creation process is identical for both. To keep notation short we introduce:

$$\underline{r}(i, j, \mathbf{m}) = \min_{d \in \{d_i, d_j\}} r(d, \mathbf{m}). \tag{12}$$

Therefore, if $\underline{r}(i, j, \mathbf{m}) \geq \bar{r}(i, j)$ then both documents appear below $\bar{r}(i, j)$. This, in turn, means that both documents are equally likely to appear at any rank:

$$\forall n, \ P(\mathbf{m}_n = d_i \mid \underline{r}(i, j, \mathbf{m}) \geq \bar{r}(i, j))$$
$$= P(\mathbf{m}_n = d_j \mid \underline{r}(i, j, \mathbf{m}) \geq \bar{r}(i, j)). \tag{13}$$

The scoring function ϕ is then defined as follows:

$$\phi(d_i, d_j, \mathbf{r}, \mathbf{m}) = \begin{cases} 0, & \underline{r}(i, j, \mathbf{m}) < \bar{r}(i, j) \\ \frac{-1}{P(\underline{r}(i,j,\mathbf{m}) \geq \bar{r}(i,j))}, & d_i <_{\mathbf{r}} d_j \\ \frac{1}{P(\underline{r}(i,j,\mathbf{m}) \geq \bar{r}(i,j))}, & d_i >_{\mathbf{r}} d_j, \end{cases} \tag{14}$$

indicating that a zero score is given if one of the documents appears above $\bar{r}(i, j)$. Otherwise, the value of ϕ is positive or negative depending on whether the ranker $\mathbf{r}$ agrees with the inferred preference between d_i and d_j. Furthermore, this score is inversely weighed by the probability $P(\underline{r}(i, j, \mathbf{m}) \geq \bar{r}(i, j))$. Therefore, pairs that are less likely to appear below their threshold $\bar{r}(i, j)$ result in a higher score than for more commonly occuring pairs. Algorithm 3 displays how the inference of PPM can be computed. The scoring function ϕ was carefully chosen to guarantee *fidelity*, the remainder of this section will sketch the proof for PPM meeting its requirements.

The two requirements for *fidelity* will be discussed in order:

Requirement 1. The first fidelity requirement states that under uncorrelated clicks the expected outcome should be zero. Consider the expected preference:

$$E[P_{nm}] = \sum_{d_i, d_j} \sum_{\mathbf{m}} P(d_i >_c d_j \mid \mathbf{m}) P(\mathbf{m}) \tag{15}$$
$$(\phi(d_i, d_j, \mathbf{r}_n, \mathbf{m}) - \phi(d_i, d_j, \mathbf{r}_m, \mathbf{m})).$$

To see that $E[P_{nm}] = 0$ under uncorrelated clicks, take any multileaving $\mathbf{m}$ where $P(\mathbf{m}) > 0$ and $\phi(d_i, d_j, \mathbf{r}, \mathbf{m}) \neq 0$ with $\mathbf{m}_x = d_i$ and $\mathbf{m}_y = d_j$. Then there is always a multileaved result list $\mathbf{m}'$ that is identical expect for swapping the two documents so that $\mathbf{m}'_x = d_j$ and $\mathbf{m}'_y = d_i$. The scoring function only gives non-zero values if both documents appear below the threshold $\underline{r}(i, j, \mathbf{m}) < \bar{r}(i, j)$ (Equation 14). At this point the probability of each document appearing at any position is the same (Equation 13), thus the following holds:

$$P(\mathbf{m}) = P(\mathbf{m}'), \tag{16}$$
$$\phi(d_i, d_j, \mathbf{r}_n, \mathbf{m}) = -\phi(d_j, d_i, \mathbf{r}_n, \mathbf{m}'). \tag{17}$$

Finally, from the definition of uncorrelated clicks (Equation 2) the following holds:

$$P(d_i >_c d_j \mid \mathbf{m}) = P(d_j >_c d_i \mid \mathbf{m}'). \tag{18}$$

As a result, any document pair (d_i, d_j) and multileaving $\mathbf{m}$ that affects the expected outcome is cancelled by the multileaving $\mathbf{m}'$. Therefore, we can conclude that $E[P_{nm}] = 0$ under uncorrelated clicks, and that PPM meets the first requirement of *fidelity*.

Requirement 2. The second *fidelity* requirement states that under correlated clicks a ranker that Pareto dominates all other rankers should win the multileaved comparison. Therefore, the expected value for a Pareto dominating ranker r_n should be:

$$\forall m, n \neq m \rightarrow E[P_{nm}] > 0. \tag{19}$$

Take any other ranker $\mathbf{r}_m$ that is thus Pareto dominated by $\mathbf{r}_n$. The proof for the first requirement shows that $E[P_{nm}]$ is not affected by any pair of documents d_i, d_j with the same relevance label. Furthermore, any pair on which $\mathbf{r}_n$ and $\mathbf{r}_m$ agree will not affect the expected outcome since:

$$(d_i >_{\mathbf{r}_n} d_j \leftrightarrow d_i >_{\mathbf{r}_m} d_j) \Rightarrow$$
$$\phi(d_i, d_j, \mathbf{r}_n, \mathbf{m}) - \phi(d_i, d_j, \mathbf{r}_m, \mathbf{m}) = 0. \tag{20}$$

Then, for any relevant document d_i, consider the set of documents that $\mathbf{r}_n$ incorrectly prefers over d_i:

$$A = \{d_j \mid \neg rel(d_j) \wedge d_j >_{\mathbf{r}_n} d_i\} \tag{21}$$

and the set of documents that $\mathbf{r}_m$ incorrectly prefers over d_i and places higher than where $\mathbf{r}_n$ places d_i:

$$B = \{d_j \mid \neg rel(d_j) \wedge d_j >_{\mathbf{r}_m} d_i \wedge r(d_j, \mathbf{l}_m) < r(d_i, \mathbf{l}_n)\}. \tag{22}$$

Since $\mathbf{r}_n$ Pareto dominates $\mathbf{r}_m$, it has the same or fewer incorrect preferences: $|A| \leq |B|$. Furthermore, for any document d_j in either A or B the threshold of the pair d_i, d_j is the same:

$$\forall d_j \in A \cup B, \bar{r}(i, j) = r(d_i, \mathbf{l}_n). \tag{23}$$

Therefore, all pairs with documents from A and B will only get a non-zero value from ϕ if they both appear at or below $r(d_i, \mathbf{l}_n)$. Then using Equation 13 and the Bayes rule we see:

$$\forall (d_j, d_l) \in A \cup B, \frac{P(\mathbf{m}_x = d_j, \underline{r}(i, j, \mathbf{m}) \geq \bar{r}(i, j, \mathcal{R}))}{P(\underline{r}(i, j, \mathbf{m}) \geq \bar{r}(i, j, \mathcal{R}))}$$
$$= \frac{P(\mathbf{m}_x = d_l, \underline{r}(i, l, \mathbf{m}) \geq \bar{r}(i, l, \mathcal{R}))}{P(\underline{r}(i, l, \mathbf{m}) \geq \bar{r}(i, l, \mathcal{R}))}. \tag{24}$$

Similarly, the reweighing of ϕ ensures that every pair in A and B contributes the same to the expected outcome. Thus, if both rankers rank d_i at the same position the following sum:

$$\sum_{d_j \in A \cup B} \sum_{\mathbf{m}} P(\mathbf{m}) \cdot$$
$$\left[P(d_i >_c d_j \mid \mathbf{m})(\phi(d_i, d_j, \mathbf{r}_n, \mathbf{m}) - \phi(d_i, d_j, \mathbf{r}_m, \mathbf{m})) \right. \tag{25}$$
$$\left. + P(d_j >_c d_i \mid \mathbf{m})(\phi(d_j, d_i, \mathbf{r}_n, \mathbf{m}) - \phi(d_j, d_i, \mathbf{r}_m, \mathbf{m})) \right]$$

will be zero if $|A| = |B|$ and positive if $|A| < |B|$ under correlated clicks. Moreover, since $\mathbf{r}_n$ Pareto dominates $\mathbf{r}_m$, there will be at least one document d_j where:

$$\exists d_i, \exists d_j, rel(d_i) \wedge \neg rel(d_j) \wedge r(d_i, \mathbf{l}_n) = r(d_j, \mathbf{l}_m). \tag{26}$$

This means that the expected outcome (Equation 15) will always be positive under correlated clicks, i.e., $E[P_{nm}] > 0$, for a Pareto dominating ranker $\mathbf{r}_n$ and any other ranker $\mathbf{r}_m$.

In summary, we have introduced a new multileaved comparison method, PPM, which we have shown to be *considerate* and to have *fidelity*. We further note that PPM has polynomial complexity: to calculate $P(\underline{r}(i, j, \mathbf{m}) \geq \bar{r}(i, j))$ only the size of the choice sets Ω and the first positions at which d_i and d_j occur in Ω have to be known.

6 EXPERIMENTS

In order to answer Research Question **RQ2** posed in Section 1 several experiments were performed to evaluate the *sensitivity* of PPM. The methodology of evaluation follows previous work on interleaved and multileaved comparison methods [2, 12, 13, 27, 28] and is completely reproducible.[2]

6.1 Ranker selection and comparisons

In order to make fair comparisons between rankers, we will use the Online Learning to Rank (OLTR) datasets described in Section 6.2. From the feature representations in these datasets a handpicked set of features was taken and used as ranking models. To match the real-world scenario as best as possible this selection consists of features which are known to perform well as relevance signals independently. This selection includes but is not limited to: BM25, LMIR.JM, Sitemap, PageRank, HITS and TF.IDF.

Then the ground-truth comparisons between the rankers are based on their NDCG scores computed on a held-out test set, resulting in a binary preference matrix P_{nm} for all ranker pairs $(\mathbf{r}_n, \mathbf{r}_m)$:

$$P_{nm} = NDCG(\mathbf{r}_n) - NDCG(\mathbf{r}_m). \qquad (27)$$

The metric by which multileaved comparison methods are compared is the *binary error*, E_{bin} [2, 27, 28]. Let $\hat{P}_{nm}$ be the preference inferred by a multileaved comparison method; then the error is:

$$E_{bin} = \frac{\sum_{n, m \in \mathcal{R} \wedge n \neq m} \mathrm{sgn}(\hat{P}_{nm}) \neq sgn(P_{nm})}{|\mathcal{R}| \times (|\mathcal{R}| - 1)}. \qquad (28)$$

6.2 Datasets

Our experiments are performed over ten publicly available OLTR datasets with varying sizes and representing different search tasks. Each dataset consists of a set of queries and a set of corresponding documents for every query. While queries are represented only by their identifiers, feature representations and relevance labels are available for every document-query pair. Relevance labels are graded differently by the datasets depending on the task they model, for instance, navigational datasets have binary labels for not relevant (0), and relevant (1), whereas most informational tasks have labels ranging from not relevant (0), to perfect relevancy (5). Every dataset consists of five folds, each dividing the dataset in different training, validation and test partitions.

The first publicly available *Learning to Rank* datasets are distributed as LETOR 3.0 and 4.0 [21]; they use representations of 45, 46, or 64 features encoding ranking models such as TF.IDF, BM25, Language Modelling, PageRank, and HITS on different parts of the documents. The datasets in LETOR are divided by their tasks, most of which come from the TREC Web Tracks between 2003 and

Table 2: Instantiations of Cascading Click Models [10] as used for simulating user behaviour in experiments.

R	$P(click = 1 \mid R)$					$P(stop = 1 \mid R)$				
	0	1	2	3	4	0	1	2	3	4
perf	0.0	0.2	0.4	0.8	1.0	0.0	0.0	0.0	0.0	0.0
nav	0.05	0.3	0.5	0.7	0.95	0.2	0.3	0.5	0.7	0.9
inf	0.4	0.6	0.7	0.8	0.9	0.1	0.2	0.3	0.4	0.5

2008 [7, 8]. *HP2003, HP2004, NP2003, NP2004, TD2003* and *TD2004* each contain between 50 and 150 queries and 1,000 judged documents per query and use binary relevance labels. Due to their similarity we report average results over these six datasets noted as *LETOR 3.0*. The *OHSUMED* dataset is based on the query log of the search engine on the MedLine abstract database, and contains 106 queries. The last two datasets, *MQ2007* and *MQ2008*, were based on the Million Query Track [1] and consist of 1,700 and 800 queries, respectively, but have far fewer assessed documents per query.

The *MLSR-WEB10K* dataset [23] consists of 10,000 queries obtained from a retired labelling set of a commercial web search engine. The datasets uses 136 features to represent its documents, each query has around 125 assessed documents.

Finally, we note there are more *OLTR* datasets available [3, 9], but there is no public information about their feature representations. Therefore, they are unfit for our evaluation as no selection of well performing ranking features can be made.

6.3 Simulating user behavior

While experiments using real users are preferred [4, 6, 18, 31], most researchers do not have access to search engines. As a result the most common way of comparing online evaluation methods is by using simulated user behaviour [2, 12, 13, 27, 28]. Such simulated experiments show the performance of multileaved comparison methods when user behaviour adheres to a few simple assumptions.

Our experiments follow the precedent set by previous work on online evaluation: First, a user issues a query simulated by uniformly sampling a query from the static dataset. Subsequently, the multileaved comparison method constructs the multileaved result list of documents to display. The behavior of the user after receiving this list is simulated using a *cascade click model* [5, 10]. This model assumes a user to examine documents in their displayed order. For each document that is considered the user decides whether it warrants a click, which is modeled as the conditional probability $P(click = 1 \mid R)$ where R is the relevance label provided by the dataset. Accordingly, *cascade click model* instantiations increase the probability of a click with the degree of the relevance label. After the user has clicked on a document their information need may be satisfied; otherwise they continue considering the remaining documents. The probability of the user not examining more documents after clicking is modeled as $P(stop = 1 \mid R)$, where it is more likely that the user is satisfied from a very relevant document. At each impression we display $k = 10$ documents to the user.

Table 2 lists the three instantiations of cascade click models that we use for this paper. The first models a *perfect* user (*perf*) who considers every document and clicks on all relevant documents and nothing else. Secondly, the *navigational* instantiation (*nav*) models a user performing a navigational task who is mostly looking

for a single highly relevant document. Finally, the *informational* instantiation (*inf*) models a user without a very specific information need who typically clicks on multiple documents. These three models have increasing levels of noise, as the behavior of each depends less on the relevance labels of the displayed documents.

6.4 Experimental runs

Each experimental run consists of applying a multileaved comparison method to a sequence of $T = 10,000$ simulated user impressions. To see the effect of the number of rankers in a comparison, our runs consider $|\mathcal{R}| = 5$, $|\mathcal{R}| = 15$, and $|\mathcal{R}| = 40$. However only the *MSLR* dataset contains $|\mathcal{R}| = 40$ rankers. Every run is repeated for every click model to see how different behaviours affect performance. For statistical significance every run is repeated 25 times per fold, which means that 125 runs are conducted for every dataset and click model pair. Since our evaluation covers five multileaved comparison methods, we generate over 393 million impressions in total. We test for statistical significant differences using a two tailed t-test. Note that the results reported on the LETOR 3.0 data are averaged over six datasets and thus span 750 runs per datapoint.

The parameters of the baselines are selected based on previous work on the same datasets; for OM the sample size $\eta = 10$ was chosen as reported by Schuth et al. [27]; for PM the degree $\tau = 3.0$ was chosen according to Hofmann et al. [12] and the sample size $\eta = 10,000$ in accordance with Schuth et al. [28].

7 RESULTS AND ANALYSIS

We answer Research Question **RQ2** by evaluating the *sensitivity* of PPM based on the results of the experiments detailed in Section 6.

The results of the experiments with a smaller number of rankers: $|\mathcal{R}| = 5$ are displayed in Table 3. Here we see that after 10,000 impressions PPM has a significantly lower error on many datasets and at all levels of interaction noise. Furthermore, for $|\mathcal{R}| = 5$ there are no significant losses in performance under any circumstances.

When $|\mathcal{R}| = 15$ as displayed in Table 4, we see a single case where PPM performs worse than a previous method: on *MQ2007* under the *perfect* click model SOSM performs significantly better than PPM. However, on the same dataset PPM performs significantly better under the *informational* click model. Furthermore, there are more significant improvements for $|\mathcal{R}| = 15$ than when the number of rankers is the smaller $|\mathcal{R}| = 5$.

Finally, when the number of rankers in the comparison is increased to $|\mathcal{R}| = 40$ as displayed in Table 5, PPM still provides significant improvements.

We conclude that PPM always provides a performance that is at least as good as any existing method. Moreover, PPM is robust to noise as we see more significant improvements under click-models with increased noise. Furthermore, since improvements are found with the number of rankers $|\mathcal{R}|$ varying from 5 to 40, we conclude that PPM is scalable in the comparison size. Additionally, the dataset type seems to affect the relative performance of the methods. For instance, on *LETOR 3.0* little significant differences are found, whereas the *MSLR* dataset displays the most significant improvements. This suggests that on more artificial data, i.e., the smaller datasets simulating navigational tasks, the differences are
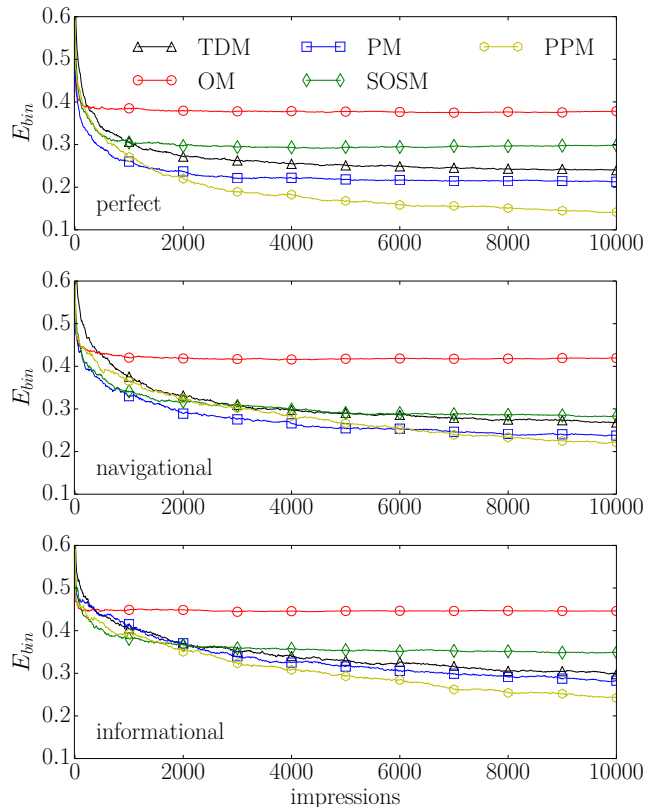

Figure 1: The binary error of different multileaved comparison methods on comparisons of $|\mathcal{R}| = 15$ rankers on the MSLR-WEB10k dataset.

fewer, while on the other hand on large commercial data the preference for PPM increases further. Lastly, Figure 1 displays the binary error of all multileaved comparison methods on the *MSLR* dataset over 10,000 impressions. Under the *perfect* click model we see that all of the previous methods display converging behavior around 3,000 impressions. In contrast, the error of PPM continues to drop throughout the experiment. The fact that the existing methods converge at a certain level of error in the absence of click-noise is indicative that they are lacking in *sensitivity*.

Overall, our results show that PPM reaches a lower level of error than previous methods seem to be capable of. This feat can be observed on a diverse set of datasets, various levels of interaction noise and for different comparison sizes. To answer Research Question **RQ2**: from our results we conclude that PPM is more sensitive than any existing multileaved comparison method.

8 CONCLUSION

In this paper we have examined multileaved comparison methods for evaluating ranking models online.

We have presented a new multileaved comparison method, Pairwise Preference Multileaving (PPM), that is more sensitive to user preferences than existing methods. Additionally, we have proposed a theoretical framework for assessing multileaved comparison methods, with *considerateness* and *fidelity* as the two key requirements.

Table 3: The binary error E_{bin} of all multileaved comparison methods after 10,000 impressions on comparisons of $|\mathcal{R}| = 5$ rankers. Average per dataset and click model; standard deviation in brackets. The best performance per click model and dataset is noted in bold, statistically significant improvements of PPM are noted by ▼ ($p < 0.01$) and ▽ ($p < 0.05$) and losses by ▲ and △ respectively or ○ for no difference, per baseline.

	TDM	OM	PM	SOSM	PPM
perfect					
LETOR 3.0	0.16 (0.13)	**0.14 (0.15)**	0.15 (0.15)	0.16 (0.15)	**0.14 (0.13)** ○ ○ ○ ○
MQ2007	0.19 (0.16)	0.22 (0.18)	**0.16 (0.14)**	0.18 (0.16)	**0.16 (0.14)** ○ ▼ ○ ○
MQ2008	0.15 (0.12)	0.19 (0.14)	0.16 (0.12)	0.18 (0.15)	**0.14 (0.12)** ○ ▼ ○ ▽
MSLR-WEB10k	0.23 (0.13)	0.27 (0.17)	0.20 (0.14)	0.25 (0.18)	**0.14 (0.13)** ▼ ▼ ▼ ▼
OHSUMED	0.14 (0.12)	0.19 (0.15)	**0.11 (0.09)**	**0.11 (0.10)**	**0.11 (0.10)** ▼ ▼ ○ ○
navigational					
LETOR 3.0	0.16 (0.13)	**0.15 (0.15)**	**0.15 (0.14)**	0.17 (0.15)	0.16 (0.14) ○ ○ ○ ○
MQ2007	0.21 (0.17)	0.33 (0.21)	0.18 (0.12)	0.29 (0.23)	**0.17 (0.14)** ○ ▼ ○ ▼
MQ2008	0.17 (0.14)	0.21 (0.20)	0.17 (0.15)	0.23 (0.18)	**0.15 (0.13)** ○ ▼ ○ ▼
MSLR-WEB10k	0.24 (0.14)	0.32 (0.20)	0.24 (0.17)	0.31 (0.19)	**0.20 (0.15)** ▽ ▼ ▽ ▼
OHSUMED	**0.12 (0.11)**	0.27 (0.19)	0.14 (0.12)	0.23 (0.17)	0.13 (0.12) ○ ▼ ○ ▼
informational					
LETOR 3.0	0.16 (0.14)	0.22 (0.19)	**0.14 (0.11)**	0.17 (0.15)	0.15 (0.13) ○ ▼ ○ ○
MQ2007	0.23 (0.15)	0.41 (0.26)	0.23 (0.15)	0.37 (0.23)	**0.17 (0.16)** ▼ ▼ ▼ ▼
MQ2008	0.18 (0.13)	0.28 (0.19)	0.18 (0.16)	0.23 (0.18)	**0.17 (0.14)** ○ ▼ ○ ▼
MSLR-WEB10k	0.27 (0.18)	0.42 (0.23)	0.24 (0.17)	0.36 (0.20)	**0.19 (0.17)** ▼ ▼ ▽ ▼
OHSUMED	0.13 (0.10)	0.37 (0.24)	**0.12 (0.11)**	0.27 (0.21)	**0.12 (0.10)** ○ ▼ ○ ▼

Table 4: The binary error E_{bin} after 10,000 impressions on comparisons of $|\mathcal{R}| = 15$ rankers. Notation is identical to Table 3.

	TDM	OM	PM	SOSM	PPM
perfect					
LETOR 3.0	0.16 (0.07)	**0.14 (0.08)**	0.15 (0.07)	0.17 (0.08)	0.16 (0.08) ○ ○ ○ ○
MQ2007	0.20 (0.07)	0.25 (0.09)	0.18 (0.06)	**0.15 (0.07)**	0.19 (0.07) ○ ▼ ○ ▲
MQ2008	0.16 (0.05)	0.17 (0.05)	0.16 (0.05)	**0.15 (0.07)**	**0.15 (0.06)** ○ ▽ ○ ○
MSLR-WEB10k	0.24 (0.07)	0.38 (0.11)	0.21 (0.06)	0.30 (0.08)	**0.14 (0.05)** ▼ ▼ ▼ ▼
OHSUMED	0.14 (0.03)	0.18 (0.05)	0.13 (0.03)	0.13 (0.03)	**0.11 (0.03)** ▼ ▼ ▼ ▼
navigational					
LETOR 3.0	0.16 (0.08)	0.16 (0.09)	**0.15 (0.08)**	0.17 (0.08)	0.17 (0.08) ○ ○ ○ ○
MQ2007	0.24 (0.07)	0.33 (0.11)	**0.20 (0.07)**	0.22 (0.08)	0.21 (0.08) ▼ ▼ ○ ○
MQ2008	0.19 (0.05)	0.21 (0.07)	**0.16 (0.05)**	0.18 (0.06)	**0.16 (0.06)** ▼ ▼ ○ ▼
MSLR-WEB10k	0.27 (0.07)	0.42 (0.12)	0.24 (0.06)	0.28 (0.09)	**0.22 (0.08)** ▼ ▼ ○ ▼
OHSUMED	0.14 (0.04)	0.25 (0.07)	**0.13 (0.03)**	0.18 (0.06)	**0.13 (0.04)** ○ ▼ ○ ▼
informational					
LETOR 3.0	0.18 (0.07)	0.20 (0.11)	0.17 (0.08)	**0.16 (0.08)**	0.18 (0.08) ○ ▽ ○ ○
MQ2007	0.28 (0.07)	0.42 (0.14)	0.26 (0.08)	0.28 (0.11)	**0.21 (0.08)** ▼ ▼ ▼ ▼
MQ2008	0.23 (0.06)	0.26 (0.11)	0.18 (0.06)	0.20 (0.06)	**0.15 (0.06)** ▼ ▼ ▼ ▼
MSLR-WEB10k	0.30 (0.09)	0.45 (0.12)	0.28 (0.08)	0.35 (0.11)	**0.24 (0.08)** ▼ ▼ ▼ ▼
OHSUMED	0.15 (0.03)	0.42 (0.09)	**0.13 (0.03)**	0.25 (0.06)	**0.13 (0.04)** ▼ ▼ ○ ▼

We have shown that no method published prior to PPM has *fidelity* without lacking *considerateness*. In other words, prior to PPM no multileaved comparison method has been able to infer correct preferences without degrading the search experience of the user. In contrast, we prove that PPM has both *considerateness* and *fidelity*, thus it is guaranteed to correctly identify a Pareto dominating ranker without altering the search experience considerably. Furthermore, our experimental results spanning ten datasets show that PPM is more sensitive than existing methods, meaning that it can reach a lower level of error than any previous method. Moreover, our experiments show that the most significant improvements are obtained on the more complex datasets, i.e., larger datasets with more grades of relevance. Additionally, similar improvements are

Table 5: The binary error E_{bin} of all multileaved comparison methods after 10,000 impressions on comparisons of $|\mathcal{R}| = 40$ rankers. Averaged over the *MSLR-WEB10k*, notation is identical to Table 3.

	TDM	OM	PM	SOSM	PPM
perfect	0.26 (0.03)	0.43 (0.02)	0.23 (0.02)	0.31 (0.02)	**0.18** (0.04) ▼▼▼▼
navigational	0.31 (0.03)	0.44 (0.01)	0.25 (0.03)	**0.23** (0.03)	0.24 (0.05) ▼▼▼○
informational	0.37 (0.04)	0.47 (0.01)	0.30 (0.05)	0.34 (0.05)	**0.27** (0.06) ▼▼▼▼

observed under different levels of noise and numbers of rankers in the comparison, indicating that PPM is robust to interaction noise and scalable to large comparisons. As an extra benefit, the computational complexity of PPM is polynomial and, unlike previous methods, does not depend on sampling or approximations.

The theoretical framework that we have introduced allows future research into multileaved comparison methods to guarantee improvements that generalize better than empirical results alone. In turn, properties like *considerateness* can further stimulate the adoption of multileaved comparison methods in production environments; future work with real-world users may yield further insights into the effectiveness of the multileaving paradigm. Rich interaction data enables the introduction of multileaved comparison methods that consider more than just clicks, as has been done for interleaving methods [18]. These methods could be extended to consider other signals such as *dwell-time* or *the order of clicks in an impression*, etc.

Furthermore, the field of OLTR has depended on online evaluation from its inception [30]. The introduction of multileaving and following novel multileaved comparison methods brought substantial improvements to both fields [22, 29]. Similarly, PPM and any future extensions are likely to benefit the OLTR field too.

Finally, while the theoretical and empirical improvements of PPM are convincing, future work should investigate whether the sensitivity can be made even stronger. For instance, it is possible to have clicks from which no preferences between rankers can be inferred. Can we devise a method that avoids such situations as much as possible without introducing any form of bias, thus increasing the sensitivity even further while maintaining theoretical guarantees?

REFERENCES

[1] J. Allan, B. Carterette, J. A. Aslam, V. Pavlu, B. Dachev, and E. Kanoulas. Million query track 2007 overview. In *TREC*. NIST, 2007.

[2] B. Brost, I. J. Cox, Y. Seldin, and C. Lioma. An improved multileaving algorithm for online ranker evaluation. In *SIGIR*, pages 745–748. ACM, 2016.

[3] O. Chapelle and Y. Chang. Yahoo! Learning to Rank Challenge Overview. *Journal of Machine Learning Research*, 14:1–24, 2011.

[4] O. Chapelle, T. Joachims, F. Radlinski, and Y. Yue. Large-scale validation and analysis of interleaved search evaluation. *TOIS*, 30(1):Article 6, 2012.

[5] A. Chuklin, I. Markov, and M. de Rijke. *Click Models for Web Search*. Morgan & Claypool Publishers, 2015.

[6] A. Chuklin, A. Schuth, K. Zhou, and M. de Rijke. A comparative analysis of interleaving methods for aggregated search. *TOIS*, 33(2):Article 5, 2015.

[7] C. L. Clarke, N. Craswell, and I. Soboroff. Overview of the TREC 2009 Web Track. In *TREC*. NIST, 2009.

[8] N. Craswell, D. Hawking, R. Wilkinson, and M. Wu. Overview of the TREC 2003 Web Track. In *TREC*. NIST, 2003.

[9] D. Dato, C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, N. Tonellotto, and R. Venturini. Fast ranking with additive ensembles of oblivious and non-oblivious regression trees. *TOIS*, 35(2):15, 2016.

[10] F. Guo, C. Liu, and Y. M. Wang. Efficient multiple-click models in web search. In *WSDM*, pages 124–131. ACM, 2009.

[11] J. He, C. Zhai, and X. Li. Evaluation of methods for relative comparison of retrieval systems based on clickthroughs. In *CIKM*, pages 2029–2032. ACM, 2009.

[12] K. Hofmann, S. Whiteson, and M. de Rijke. A probabilistic method for inferring preferences from clicks. In *CIKM*, pages 249–258. ACM, 2011.

[13] K. Hofmann, S. Whiteson, and M. de Rijke. Fidelity, soundness, and efficiency of interleaved comparison methods. *TOIS*, 31(4):17, 2013.

[14] K. Hofmann, L. Li, and F. Radlinski. Online evaluation for information retrieval. *Foundations and Trends in Information Retrieval*, 10(1):1–117, 2016.

[15] T. Joachims. Optimizing search engines using clickthrough data. In *KDD*, pages 133–142. ACM, 2002.

[16] T. Joachims. Unbiased evaluation of retrieval quality using clickthrough data. In *SIGIR Workshop on Mathematical/Formal Methods in Information Retrieval*, volume 354, 2002.

[17] T. Joachims. Evaluating retrieval performance using clickthrough data. In *Text Mining*. Physica/Springer, 2003.

[18] E. Kharitonov, C. Macdonald, P. Serdyukov, and I. Ounis. Generalized team draft interleaving. In *CIKM*, pages 773–782. ACM, 2015.

[19] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne. Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery*, 18(1):140–181, 2009.

[20] R. Kohavi, A. Deng, B. Frasca, T. Walker, Y. Xu, and N. Pohlmann. Online controlled experiments at large scale. In *KDD*, pages 1168–1176. ACM, 2013.

[21] T.-Y. Liu, J. Xu, T. Qin, W. Xiong, and H. Li. LETOR: Benchmark dataset for research on learning to rank for information retrieval. In *LR4IR '07*, 2007.

[22] H. Oosterhuis, A. Schuth, and M. de Rijke. Probabilistic multileave gradient descent. In *ECIR*, pages 661–668. Springer, 2016.

[23] T. Qin and T. Liu. Introducing LETOR 4.0 datasets. *CoRR*, abs/1306.2597, 2013.

[24] F. Radlinski and N. Craswell. Optimized interleaving for online retrieval evaluation. In *WSDM*, pages 245–254. ACM, 2013.

[25] F. Radlinski, M. Kurup, and T. Joachims. How does clickthrough data reflect retrieval quality? In *CIKM*, pages 43–52. ACM, 2008.

[26] M. Sanderson. Test collection based evaluation of information retrieval systems. *Foundations and Trends in Information Retrieval*, 4(4):247–375, 2010.

[27] A. Schuth, F. Sietsma, S. Whiteson, D. Lefortier, and M. de Rijke. Multileaved comparisons for fast online evaluation. In *CIKM*, pages 71–80. ACM, 2014.

[28] A. Schuth, R.-J. Bruintjes, F. Büttner, J. van Doorn, et al. Probabilistic multileave for online retrieval evaluation. In *SIGIR*, pages 955–958. ACM, 2015.

[29] A. Schuth, H. Oosterhuis, S. Whiteson, and M. de Rijke. Multileave gradient descent for fast online learning to rank. In *WSDM*, pages 457–466. ACM, 2016.

[30] Y. Yue and T. Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *ICML '09*, 2009.

[31] Y. Yue, Y. Gao, O. Chapelle, Y. Zhang, and T. Joachims. Learning more powerful test statistics for click-based retrieval evaluation. In *SIGIR*, pages 507–514. ACM, 2010.

[32] Y. Yue, R. Patel, and H. Roehrig. Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data. In *WWW*, pages 1011–1018. ACM, 2010.