A Comparative Analysis of Interleaving Methods for Aggregated Search

ALEKSANDR CHUKLIN and ANNE SCHUTH, University of Amsterdam
KE ZHOU, Yahoo Labs London
MAARTEN DE RIJKE, University of Amsterdam

A result page of a modern search engine often goes beyond a simple list of "10 blue links." Many specific user needs (e.g., News, Image, Video) are addressed by so-called aggregated or vertical search solutions: specially presented documents, often retrieved from specific sources, that stand out from the regular organic Web search results. When it comes to evaluating ranking systems, such complex result layouts raise their own challenges. This is especially true for so-called interleaving methods that have arisen as an important type of online evaluation: by mixing results from two different result pages, interleaving can easily break the desired Web layout in which vertical documents are grouped together, and hence hurt the user experience.

We conduct an analysis of different interleaving methods as applied to aggregated search engine result pages. Apart from conventional interleaving methods, we propose two vertical-aware methods: one derived from the widely used Team-Draft Interleaving method by adjusting it in such a way that it respects vertical document groupings, and another based on the recently introduced Optimized Interleaving framework. We show that our proposed methods are better at preserving the user experience than existing interleaving methods while still performing well as a tool for comparing ranking systems. For evaluating our proposed vertical-aware interleaving methods, we use real-world click data as well as simulated clicks and simulated ranking systems.

Categories and Subject Descriptors: H3 [**Information Storage and Retrieval**]: H.3.3 Information Search and Retrieval

General Terms: Algorithms, Experimentation, Measurement

This research was partially supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreements 288024 (LiMoSINe) and 312827 (VOX-Pol); the Netherlands Organisation for Scientific Research (NWO) under project numbers 727.011.005, 612.001.116, HOR-11-10, and 640.006.013; the Center for Creation, Content and Technology (CCCT); the QuaMerdes project funded by the CLARIN-nl program; the TROVe project funded by the CLARIAH program; the Dutch national program COMMIT; the ESF Research Network Program ELIAS; the Elite Network Shifts project funded by the Royal Dutch Academy of Sciences (KNAW); the Netherlands eScience Center under project number 027.012.105; the Yahoo! Faculty Research and Engagement Program; the Microsoft Research Ph.D. program; and the HPC Fund.

This article extends work published previously in Chuklin et al. [2013a]. We extend our earlier work by including a thorough study of the optimized interleaving framework that complements our previous constructive approach. We also extend our interleaving methods to handle heterogeneous pages with multiple vertical blocks of different type. By doing so, we remove limitations present in Chuklin et al. [2013a] and extend interleaving methods to all currently possible search result pages. We also broaden our study by considering different aspects of the user experience and different definitions of sensitivity, and by studying new datasets of model rankers in addition to previously studied simulated and real rankings.

Authors' addresses: A. Chuklin (corresponding author), A. Schuth, and M. De Rijke, University of Amsterdam; emails: {a.chuklin, anne.schuth, derijke}@uva.nl; K. Zhou, Yahoo Labs London. Some parts of this work were done while the first author was at Yandex Russia and Google Switzerland (current affiliation); email: zhouke.nlp@gmail.com.

DOI: http://dx.doi.org/10.1145/2668120

Additional Key Words and Phrases: Information retrieval, interleaved comparison, interleaving, clicks, online evaluation, aggregated search

ACM Reference Format:
Aleksandr Chuklin, Anne Schuth, Ke Zhou, and Maarten de Rijke. 2015. A comparative analysis of interleaving methods for aggregated search. ACM Trans. Inf. Syst. 33, 2, Article 5 (February 2015), 38 pages.
DOI: http://dx.doi.org/10.1145/2668120

1. INTRODUCTION

There is a general trend to aggregate search results from different verticals (e.g., News, Image, Video) and present them in one search result page together with "general Web" or "organic Web" results. This new search paradigm is called *aggregated search* or *federated search* and has been adopted by all major commercial search engines. As much as one third of all queries have an intent that can be answered by a specific vertical, whereas 10% to 30% of the queries require at least two different types of vertical. This was first pointed out in a study by Arguello et al. [2009], where human editors found at least two relevant verticals for 30% of the queries in a random sample of Yahoo! Search queries. More conservative estimations from other search engines were later reported by Chen et al. [2012] and Styskin [2013]. With respect to presentation style, it has become standard to group the results coming from the same vertical and present them as one coherent block. As was shown by Dumais et al. [2001], presenting results in a grouped manner simplifies browsing result lists and helps users to navigate faster.

As web search engines constantly evolve, their quality needs to be evaluated. *Interleaving* [Joachims 2002, 2003] is an evaluation method for comparing the relative quality of two ranking systems. Users are presented with a mixed or *interleaved* list of results from both rankings, and the users' clicks are used to determine which system is better. Interleaving methods have proved to be an efficient tool; they allow us to find preferred system from user clicks faster than other methods, such as *A*/*B*-testing [Radlinski et al. 2008].

How can interleaving methods be used to measure the quality of aggregated search systems that combine Web and vertical results? We want to achieve the efficiency of conventional interleaving methods while preserving the conventional aggregated search user experience, in which vertical results are organized in blocks. In this article, we propose several methods for extending interleaving methods to complex heterogeneous pages comprising results from different verticals.

Our main research questions are as follows:

RQ1 *Influence on the user experience*: What *effect* do different interleaving methods (both conventional and newly introduced vertical aware) have *on the user experience* in the context of aggregated search? Do any of these methods run the risk of degrading the quality of the results or altering the user experience?
RQ2 *Correctness and sensitivity*: Do different interleaving methods always draw *correct* conclusions about the better system? How *fast*, in terms of the number of impressions and amount of feedback needed, can they detect that one aggregated search system is to be preferred over another?
RQ3 *Unbiasedness*: Do the interleaving methods that we consider provide a fair and *unbiased* comparison, or do some of them erroneously infer a preference for one aggregated search system over another in situations where implicit feedback is provided by a randomly clicking user?

To answer RQ1, we consider different pairs of aggregated result pages and analyze the effect on the user experience imposed by the interleaving methods that we consider. We go beyond the restricted situation of a single vertical block present in both rankings

that was considered in Chuklin et al. [2013a] and analyze a more general and more complex scenario where the aggregated result pages are allowed to have multiple blocks that originate from different verticals. We also add an evaluation of the quality of the interleaved list as captured by both click metrics and offline rater-based metrics.

RQ2 concerns the ability of an interleaving method to correctly capture the difference between two rankers by using the minimal amount of implicit user feedback. We extend previous work by analyzing the ability to notice not only strong differences between rankers (formulated in terms of Pareto dominance) but also subtle differences reported by offline and online quality metrics, including those specific to aggregated search.

Finally, to answer RQ3, we need to check that none of the evaluation methods infer statistically significant preferences when we assume a randomly clicking user. By removing limitations on the number of vertical blocks per result page and considering additional datasets, we make the study substantially more thorough and complete than in the earlier publication on which this article is based [Chuklin et al. 2013a].

The rest of the article is organized as follows. We discuss related work in Section 2. In Section 3, we introduce conventional interleaving methods and their limitations as applied to modern search result pages. In Section 4, we present two vertical-aware interleaving methods capable of dealing with not just one but many vertical blocks of different types. In Section 5, we introduce the experimental setups that we use and discuss their strengths and weaknesses. In Sections 6, 7, and 8, we address the research questions formulated earlier. We conclude in Section 9.

2. RELATED WORK

Two lines of research relate to this work. One focuses on evaluating aggregated search engine result pages using different evaluation paradigms (Section 2.1). The other explores utilizing interleaving methods for evaluating search rankings (Section 2.2).

2.1. Aggregated Search

Aggregated search deals with presenting results from different verticals in one unified interface. The search engine that introduced this aggregated or faceted search early on was Naver; there, result pages allow many more than 10 results per query, and results are always grouped into separate panels, one for each vertical [Seo et al. 2011]. Yahoo! and Bing historically inserted blocks of vertical documents on fixed positions (slots) in addition to the organic Web search documents [Arguello et al. 2011a; Ponnuswami et al. 2011]. Yandex allows vertical blocks to appear in any position [Chuklin et al. 2013c], and the same appears to hold for Google.

Evaluation of aggregated search is complex and challenging, as there is a variety of compounding factors. Four key components of aggregated search are the key influence of user's experience: vertical selection (VS), vertical diversity (VD), item selection (IS), and result presentation (RP). VS and VD deal with deciding which set of diverse verticals are implicitly intended by a query. IS deals with selecting a subset of items from each vertical to present on the aggregated page. RP deals with organizing and embedding the various types of result on the result page.

There is a growing body of work on evaluating aggregated search engine result pages. Relatively early work considers VS to be the main criterion of evaluation, and much of the research [Arguello et al. 2009; Zhou et al. 2012b] aims to measure the quality of the set of selected verticals, compared with an annotated set obtained by collecting manual labels from assessors underlying different assumptions [Zhou et al. 2013a, 2014].

In later work, to evaluate both VS and RP, Arguello et al. [2011b] propose to use pairwise preference evaluation to rank the vertical blocks for a given query. Targeting a similar objective, Ponnuswami et al. [2011] describe the process of manual assessments of both vertical and organic documents, and propose a click-based assessment

of vertical block placement similar to Joachims [2002]. However, they neither discuss the combined effect of these two ranking aspects nor suggest a way to compare vertical documents inside one block to each other (IS). Most of these evaluation methods assume that the vertical block is atomic—that is, it is considered to have no internal structure. However, this is not always the case. For example, a news block usually contains a list of headlines. Each of them can be judged separately and compared to organic Web results.

Recently, Zhou et al. [2012a] followed the Cranfield paradigm [Cleverdon et al. 1966] and proposed an evaluation framework for measuring aggregated search engine result page quality by modeling all, or a subset, of the four aggregated search key components discussed earlier (VS, VD, IS, and RP). The main differences between these proposed metrics are the way in which they model each factor and the way that they combine them. A list of diversity metrics such as α-NDCG [Clarke et al. 2008] is also adapted to evaluate aggregated search. By meta-evaluating those metrics on their *reliability* and *intuitiveness*, Zhou et al. [2013b] conclude that the AS_{RBP} metric [Zhou et al. 2012a] is the preferred metric.

In contrast to previous work, we base our methods on an efficient interleaving method, which was proven to accurately infer user preferences from implicit feedback. Most of the preceding evaluation approaches rely on offline editorial assessments; in contrast, our proposed approach only needs naturally occurring user interactions, which is cheap and useful in an online environment. Similar to Zhou et al. [2012a], we aim to capture the four key components of aggregated search together when we conduct our evaluations. To answer RQ1, we also utilize a set of offline metrics (e.g., AS_{RBP}) to quantify the quality of the results to verify that our proposed approaches do not degrade the user experience.

2.2. Interleaving

Although the traditional Cranfield approach [Cleverdon et al. 1966] to ranker evaluation is still widely used, there is a growing trend to use implicit feedback from users to evaluate and learn ranking models. Starting with the work of Joachims [2002, 2003], the idea of interleaving methods has become increasingly popular.

The two most commonly used interleaving methods are Team-Draft Interleaving (TDI) [Radlinski et al. 2008] and Balanced Interleaving (BI) [Joachims 2003]. TDI can be described as follows. For each user query, we build an interleaved list L whose documents are contributed by rankings A and B—the two rankings that we want to compare. This interleaved list is then shown to the user, and the user's clicks are recorded. The system that contributes most of the documents clicked by the user is inferred to be the winner for the particular query-user pair; the system that wins for most such pairs is then considered to be the better system. BI uses a different algorithm to build an interleaved list L and a more sophisticated procedure for determining the winner. These interleaving methods, as well as their modifications, were extensively studied in Chapelle et al. [2012] and Hofmann et al. [2013b].

Other interleaving methods include Document Constraints Interleaving (DCI) by He et al. [2009] and Probabilistic Interleaving (PI) by Hofmann et al. [2011]. DCI produces interleaved lists similar to BI, but it has a different and more involved way of determining the winner by computing which ranking violates the smallest number of constraints. PI has the advantage that historical interaction data can be reused using importance sampling, such as in an online learning to rank setting [Hofmann et al. 2013a] or potentially also in an evaluation setting. In principle, PI with importance sampling could also be adapted to be vertical aware. However, because PI relies on probabilistic rankers, it risks showing the user poor rankers that are not related to the original rankers being interleaved, which may affect user experience [Hofmann et al. 2013a].

Optimized Interleaving (OI), a very recent approach to generating interleaved pages, is proposed by Radlinski and Craswell [2013]. This method does not have the drawbacks of PI. Unlike TDI, it first enumerates all possible interleaved lists and then assigns probabilities according to which an interleaved list is drawn. This probability distribution is selected such that the comparison is unbiased and has optimal sensitivity. We build vertical-aware interleaving methods on top of both TDI and OI.

Most interleaving methods presented until now were evaluated in terms of *sensitivity* and *correctness*, which corresponds to our RQ2. Radlinski and Craswell [2010] show that interleaving is more accurate and more sensitive than standard offline metrics such as NDCG or MAP. Radlinski and Craswell [2013] apply the same evaluation method to show that OI is more sensitive than TDI if the right credit function is used. They also analyze typical biases of different interleaving methods (related to our RQ3).

Another meta-evaluation method was proposed by He et al. [2009], where different interleaving methods were compared in terms of their effect on the user experience (related to our RQ1). Unfortunately, they do not report any difference between interleaving methods nor do they compare the quality of the interleaved system to the original A and B systems being interleaved. We substantially extend this meta-evaluation method in Section 6 (RQ1).

A comprehensive study with a systematic analysis of interleaving methods and their meta-evaluation was suggested by Hofmann et al. [2013b]. The main questions that we aim to answer are related to, but different from, the notions of fidelity, soundness, and efficiency studied by Hofmann et al. [2013b]. First of all, we put more focus on the *user experience* aspect (RQ1), which was not analyzed in Hofmann et al. [2013b]. Second, our RQ2 unites two closely related questions of sensitivity and correctness, which in Hofmann et al. [2013b] were split across definitions of efficiency and fidelity (their Definitions 4.4 and 4.2(2), respectively). Finally, we separate the question of *unbiasedness,* which in Hofmann et al. [2013b] was bundled with correctness (their Definition 4.2(1)). In addition, we do not study the notion of soundness introduced by Hofmann et al. [2013b], as it is trivially satisfied for the interleaving methods that we use.

Our work differs in important ways from the work just discussed. In contrast to previous work on evaluating aggregated search, we perform online evaluations based on interleaving methods that allow us to evaluate aggregated system as a whole and to do so efficiently. In contrast to previous work on interleaving, we propose an interleaving method that preserves the user experience on complex aggregated search engine result pages while maintaining a high degree of sensitivity and preserving unbiasedness.

3. CONVENTIONAL INTERLEAVING

To design a new interleaving method, one can follow one of two possible ways. The first way, followed by the majority of the interleaving methods [Hofmann et al. 2011; Joachims 2003; Radlinski et al. 2008], assumes that the designer of the interleaving method specifies the probability distribution over possible interleaved lists. This probability distribution was made explicit in PI [Hofmann et al. 2011] and left implicit, although easily reconstructible, in BI [Joachims 2003] and TDI [Radlinski et al. 2008].[1] Another approach introduced by Radlinski and Craswell [2013] specifies an interleaving method as an optimization problem that allows us to explicitly tune the method for better sensitivity.

According to Joachims [2003], a good interleaving method should be blind to the user and have a low usability impact. This is particularly important for vertical search,

[1]The distribution is uniform among allowed rankings in these two methods.

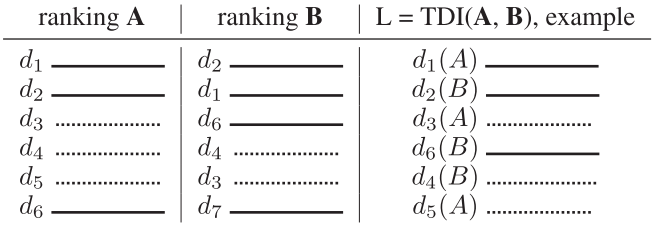

Fig. 1. Two rankings with a vertical block and one of the possible interleaved lists (TDI). Vertical documents are shown as dotted lines.

because as we will show later, conventional interleaving methods do not fully satisfy these conditions.

We will use two conventional interleaving methods as baseline methods: TDI and OI. The former is widely used in commercial settings [Chuklin et al. 2013b; Radlinski and Craswell 2010], whereas the latter has good flexibility that enables us to naturally extend it to the aggregated search setup. The same two interleaving methods were also extended in Schuth et al. [2014] to allow for comparisons of multiple rankers at once.

3.1. Team-Draft Interleaving

Conventional TDI (Algorithm 1) allows rankings A and B to contribute documents turn by turn and keeps track of which side contributes which document ($Team_A$ or $Team_B$). By flipping a coin (after each side has added two documents), the algorithm ensures a fair and unbiased team assignment and sufficient variability in the result list L.

ALGORITHM 1: Conventional Team-Draft Interleaving (TDI).

function TDI(ranking A, ranking B)
  $L \leftarrow []$; $Team_A \leftarrow \varnothing$; $Team_B \leftarrow \varnothing$
  while $|L| < N$ **do**
    if $|Team_A| < |Team_B| +$ RANDOMINTEGER$(0, 1)$ **then**
      $k \leftarrow \min\{i : A[i] \notin L\}$
      $Team_A \leftarrow Team_A + A[k]$ ▷ add $A[k]$ to $Team_A$
      $L \leftarrow L + A[k]$ ▷ append $A[k]$ to the list L
    else
      $k \leftarrow \min\{i : B[i] \notin L\}$
      $Team_B \leftarrow Team_B + B[k]$ ▷ add $B[k]$ to $Team_B$
      $L \leftarrow L + B[k]$ ▷ append $A[k]$ to the list L
  return L

As we can see in Algorithm 1, there is nothing that prevents result list L from mixing up documents from different verticals with Web documents. Even when we have only one vertical block of the same type in A and B, the algorithm may give rise to two blocks in the interleaved list and therefore affect the user experience. For instance, Figure 1 shows a schematic representation of two ranked lists A, B and one of the possible interleaved lists. In this example, vertical documents d_3, d_4, and d_5 are not grouped, which deviates from the traditional user experience.

3.2. Optimized Interleaving

Radlinski and Craswell [2013] propose formalizing the "low usability impact" constraint (our RQ1) using a *prefix condition* (called c' there):

$$\forall k \; \exists i, j \text{ such that } L^k = A^i \cup B^j, \tag{3.1}$$

where A^i is the set of top i documents returned by system A, B^j is the set of top j documents returned by system B, and L^k is the set of top k documents of the interleaved ranking L.[2]

When we fix a prefix condition, all we need to do is assign probabilities to all possible interleaved lists L that conform to this condition (we denote the set of possible interleaved lists as $\mathcal{L}$). Radlinski and Craswell [2013] suggest choosing this probability distribution to maximize the method's sensitivity while preserving its unbiasedness.[3] We also need to specify a credit function $\delta(d)$ that assigns a particular credit to document d depending on its rank in rankings A and B. If we assume for now that the credit function is fixed, we can write down the following optimization problem [Radlinski and Craswell 2013, Section 3]:

$$p_i \in [0, 1] \tag{3.2}$$

$$\sum_{i=1}^{|\mathcal{L}|} p_i = 1 \tag{3.3}$$

$$\forall k \in \{1, \ldots, N\} : \sum_{i=1}^{|\mathcal{L}|} p_i \delta_k(L_i) = 0 \tag{3.4}$$

$$\sum_{i=1}^{|\mathcal{L}|} p_i s(L_i) \to max, \tag{3.5}$$

where s is an estimated sensitivity function defined as

$$\begin{aligned} s(L) &= -\frac{1 - w_T}{w_A + w_B} \cdot (w_A \log w_A + w_B \log w_B - (w_A + w_B) \log(w_A + w_B)) \\ w_A &= \sum_{i:\delta_i > 0} f(i) \\ w_B &= \sum_{i:\delta_i < 0} f(i) \\ w_T &= \sum_{i:\delta_i = 0} f(i) \\ f(i) &= \frac{1/i}{\sum_{k=1}^{N} 1/k}. \end{aligned}$$

We use the same constraints as Radlinski and Craswell [2013] for the credit function. Given a document d, let $\delta(d)$ denote the credit assigned to system A for this document.[4] Thus, when $\delta(d)$ is positive, then A receives credit; if it is negative, then B receives credit. The credit function should satisfy the following:

$$\text{rank}(d, A) < \text{rank}(d, B) \Leftrightarrow \delta(d) > 0 \tag{3.6}$$

$$\text{rank}(d, A) > \text{rank}(d, B) \Leftrightarrow \delta(d) < 0, \tag{3.7}$$

[2]Note that interleaved lists produced by TDI also satisfy the prefix condition (3.1).

[3]Radlinski and Craswell use a different formalization of the unbiasedness, which is, however, mathematically equivalent to our condition (3.4).

[4]For simplicity, we use the same names for rankers (ranking systems) and the ranking lists that they generate. It should be clear from the context what we refer to.

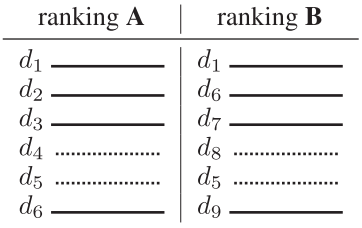

Fig. 2. Two rankings with a vertical block at the same position.

where rank(d, X) is a position of the document d in ranking X (we set rank$(d, X) = +\infty$ if $d \notin X$). In particular, we set δ to linear rank difference, which is the most sensitive setting in Radlinski and Craswell [2013, Equation (14)]:

$$\delta_k(L) = \text{rank}(d_k, A) - \text{rank}(d_k, B). \tag{3.8}$$

The interleaving method then proceeds as specified in Algorithm 2.

ALGORITHM 2: Conventional Optimized Interleaving (OI).

1: **function** OI(ranking A, ranking B, credit function δ_i)
2: $\mathcal{L} \leftarrow \{L \mid L \text{ satisfies (3.1)}\}$
3: Find a distribution $\{p_L \mid L \in \mathcal{L}\}$ that conforms to the constraints (3.2), (3.3), (3.4) and maximizes the average sensitivity (3.5).
4: Sample L from $\mathcal{L}$ according to the probability distribution p_L
5: **return** L

If we are to add a constraint that discards all rankings L that split vertical documents of the same type, we will end up with a very biased list of rankings $\mathcal{L}$. Consider the example provided in Figure 2. In this example, each ranking A and B has exactly one vertical block—say, News, which consists of different documents and resides on positions 4 and 5 in both rankings A and B. We also assume that organic rankings are sufficiently different in A and B. We can easily see that the prefix condition (3.1) does not allow system B to contribute its vertical documents until it has contributed all documents before the block. The same holds for system A. This means that if we want both systems to contribute to the resulting vertical block of L, we are forced to place the block lower in the ranking because we first need to show the union of before-block documents from A and B. This implies that, in Figure 2, we need to present d_1, d_2, d_3, d_6, d_7 before the vertical block whenever we want both d_4 and d_8 to both be present in the resulting ranking. But if we do so, we change the user experience by pushing the vertical block to the bottom of the result page and hence influence the user experience (RQ1) in a negative way. We also risk having a bigger vertical block than in either ranking A or ranking B for the same reason.

4. VERTICAL-AWARE INTERLEAVING

In this section, we discuss vertical-aware extensions of the two interleaving methods that we study: TDI and OI. We modify these interleaving methods such that the vertical documents of the same type are guaranteed to be grouped.

4.1. Vertical-Aware Team-Draft Interleaving

We describe an algorithm that may be viewed as a generalization of the conventional TDI method by Radlinski et al. [2008]. The intuition is to start with the standard TDI method and then alter it to meet the following requirements:

(1) Both of the ranked lists being interleaved should contribute to the vertical block placement size and position in the interleaved list;
(2) Both ranked lists should contribute vertical and organic Web documents;
(3) Team assignment should be "fair"; and
(4) The resulting interleaved list should not degrade the user experience.

We propose the vertical-aware team-draft interleaving (VA-TDI) method (Algorithm 3), which generalizes Algorithm 1 in Chuklin et al. [2013a] to the case of multiple vertical blocks. The main idea is to enforce the grouping of vertical documents. Therefore, our algorithm proceeds like the conventional TDI method until it hits the first vertical document. After that, it interleaves only vertical documents of this type (line 29) until the block has been formed (line 18)—that is, there are no vertical documents left or the desired block size is reached.[5] After that, the algorithm continues as usual but ignores the vertical documents whose blocks have already been formed (line 33).

If we look back to our original goals, we see that Algorithm 3 explicitly chooses block sizes between those of A and B (with some smoothing to do exploration) while the position of the block is contributed implicitly (although both ranked lists can influence it). Requirements 2 and 3 are met automatically due to the TDI procedure that we reuse (after one ranked list wins the coin flip and contributes the document, the other ranked list has to contribute the next document), although we also verify them along with requirement 4 in our experiments.

On the other hand, the algorithm proposed has some weaknesses. First of all, it assumes a rejection sampling procedure: if the selected block size cannot be achieved, we reject the list being built and start a new one with another randomization (line 31). This usually happens when A and B have very different block sizes and we cannot build a big enough vertical block, as VA-TDI assumes that A and B contribute roughly the same number of vertical documents (Algorithm 3, line 14). Using synthetic rankings (Section 5.1.3), we identify that rejection happens on average 3.8 times per interleaving function invocation with slightly over 100 in the worst case. This could lead to unwanted delays in building the interleaved search engine result page and may potentially affect the user if the search is run on slow hardware. It is worth noting, however, that the TDI method normally takes a negligible amount of query processing time, and even a 100-fold increase should not lead to a visible degradation.[6] Another issue of the VA-TDI method appears when we consider A and B with different vertical blocks. The method's nature prevents one ranked list from contributing more than two documents in a row, which means that if A has a vertical of type t and size n ($n \geq 3$) while B does not have a vertical of type t, the interleaved list will only be able to include one or two top documents from this block. As we will see later, this leads to smaller vertical blocks (Section 6.1), which may affect the user experience. This also limits the sensitivity of VA-TDI (because it may skip vertical documents)—as we will see in Section 7, VA-TDI is unable to break the limit of sensitivity 0.8 even when we consider strong cases where A Pareto dominates B. Having these limitations in mind, in the next section we propose another interleaving method, VA-OI, that does not suffer from them.

4.2. Vertical-Aware Optimized Interleaving

To extend the OI method presented in Section 3.2, we extend Equation (3.1) and add an additional constraint that guarantees that vertical documents of the same type are presented in a grouped manner (Equation (4.8)).

[5]We pick the desired block size beforehand (line 7) to ensure that requirement (1) is met.
[6]For practical purposes, we can also introduce a cut-off and skip interleaving if it takes too much time to build. However, this would lead to a biased sample and requires a separate analysis.

ALGORITHM 3: Vertical-Aware Team-Draft Interleaving (VA-TDI).

function VaTdi(ranking A, ranking B)
  $vLeft \leftarrow \varnothing$ ▷ verticals left (not yet present in L)
  for every vertical type t present in either A or B **do**
    $A_t \leftarrow \{d \in A \mid d \text{ is a vertical doc of type } t\}$
    $B_t \leftarrow \{d \in B \mid d \text{ is a vertical doc of type } t\}$
    $Size_t^A \leftarrow |A_t|$; $Size_t^B \leftarrow |B_t|$
    $Size_t^L \leftarrow$ SampleSmoothly$(Size_t^A, Size_t^B)$
    if $Size_t^L \neq 0$ **then**
      $vLeft \leftarrow vLeft \cup \{t\}$
  $L \leftarrow []$
  $Team_A \leftarrow \varnothing$; $Team_B \leftarrow \varnothing$
  $t \leftarrow$ NULL ▷ current vertical type
  while $|L| < N$ **do**
    if $|Team_A| < |Team_B| +$ RandomInteger$(0, 1)$ **then**
      AddNextDocFrom$(A, t, vLeft)$
    else
      AddNextDocFrom$(B, t, vLeft)$
    if $t \neq$ NULL and $|\{d \in L \mid d \text{ is a vertical doc of type } t\}| = Size_t^L$ **then**
      $vLeft \leftarrow vLeft \backslash \{t\}$ ▷ this vertical is finished in L
      $t \leftarrow$ NULL
  return L

function SampleSmoothly(integer a, integer b)
  if $a > b$ **then**
    Swap(a, b)
  Sample r randomly from $[a-1, b+1]$ where all integers from $[a, b]$ have equal probability p; $(a-1)$ and $(b+1)$, each has probability $\frac{p}{2}$
  return r

procedure AddNextDocFrom(ranking X, current vertical type t, $vLeft$)
  ▷ X is either A or B
  if $t \neq$ NULL **then**
    $X_{left} \leftarrow \{i \mid X[i] \in X_t \backslash L\}$
    if $X_{left} = \varnothing$ **then**
      return ▷ interleaved list is rejected, start a new one
  else
    $X_{left} \leftarrow \{i \mid X[i] \in X \backslash L \text{ and } (X[i] \text{ is a Web doc or } \exists t' \in vLeft : X[i] \in A_{t'})\}$
  $k \leftarrow \min X_{left}$ ▷ select the document with the min rank
  $Team_X \leftarrow Team_X \cup \{X[k]\}$ ▷ add the document to the team
  if $X[k]$ is a vertical doc of type t' **then**
    $t \leftarrow t'$
  $L \leftarrow L + X[k]$ ▷ append the document to L

Consider a list of documents A, containing multiple blocks of vertical documents of different types. Let A_{Web} be the list of organic Web results in A, so A stripped of all vertical blocks. For any vertical type $t \in \{\text{Image}, \text{News}, \ldots\}$, let A_t be the list of vertical documents of that type in A. This list may be empty. We assume that all vertical documents of one type occur in A in one group, so all of those documents are grouped together in the search engine result page.

Let A^k_{Web} be the list of the k documents in A_{Web} having the smallest rank values (ranked higher). So for $A_{\text{Web}} = (d_1, \ldots, d_n)$, we have $A^k_{\text{Web}} = (d_1, \ldots, d_k)$. For any vertical type t, define A^k_t in a similar way. Note that it is not necessarily so that a document d_i

in $A_{\text{Web}} = (d_1, \ldots, d_n)$ is ranked at position i, as there may be vertical documents above d_i, pushing it to a higher rank (lower position). Therefore, there may be documents in A^k that are ranked below position k. For a document d, let $\text{rank}(d, A)$ denote its rank in A, with $\text{rank}(d, A) = +\infty$ if d is not in A. For a set of documents S, we define $\text{rank}(S, A) = \min_{d \in S} \text{rank}(d, A)$.

From two lists A and B, we generate a new list L that is a combination of the two. We define similar lists and functions for B and L as earlier for A. The constraints that should be satisfied by any generated list are as follows:

$$\forall k \; \exists i, j : L^k_{\text{Web}} = A^i_{\text{Web}} \cup B^j_{\text{Web}} \tag{4.1}$$

$$\forall t, k \; \exists i, j : L^k_t = A^i_t \cup B^j_t \tag{4.2}$$

$$\forall t \text{ s.t. } L_t \neq \varnothing : \min\left(|A_t|, |B_t|\right) \leq |L_t| \leq \max\left(|A_t|, |B_t|\right) \tag{4.3}$$

$$\forall t \text{ s.t. } L_t \neq \varnothing : \text{rank}(L_t, L) \geq \min(\text{rank}(A_t, A), \text{rank}(B_t, B)) \tag{4.4}$$

$$\forall t \text{ s.t. } L_t \neq \varnothing : \text{rank}(L_t, L) \leq \max(\text{rank}(A_t, A), \text{rank}(B_t, B)) \tag{4.5}$$

$$|\{t \mid L_t \neq \varnothing\}| \geq \min\left(|\{t \mid A_t \neq \varnothing\}|, |\{t \mid B_t \neq \varnothing\}|\right) \tag{4.6}$$

$$|\{t \mid L_t \neq \varnothing\}| \leq \max\left(|\{t \mid A_t \neq \varnothing\}|, |\{t \mid B_t \neq \varnothing\}|\right) \tag{4.7}$$

$$\forall t \; \exists m, n : \forall k \in [m, n] : \exists d \in L_t : \text{rank}(d, L) = k. \tag{4.8}$$

Constraint (4.1) is the prefix constraint introduced by Radlinski and Craswell [2013]. It requires any prefix of L_{Web} to be a combination of the top i and top j documents of A_{Web} and B_{Web}. This essentially means that the new list could be constructed by repeatedly taking the top document that is not used yet from either A_{Web} or B_{Web}. Constraint (4.2) is an adaptation of the prefix constraint for verticals. It requires that the prefix of any vertical block L_t is a combination of top documents of the lists of verticals of the same type in A and B. Constraint (4.3) requires the size of each vertical block to be between the sizes of the vertical blocks of the same type in A and in B. Constraints (4.4) through (4.5) control the position of each vertical block. The rank of the vertical block should be between the ranks of the blocks of the same type in A and B.

We need additional constraints to control the number of vertical blocks in the list L. If we only had constraints (4.1) through (4.5), it might be the case that there are three vertical blocks in A, three vertical blocks of different types in B, and six vertical blocks in the result list L. This would change the user experience a lot. Therefore, constraints (4.6) and (4.7) limit the number of vertical blocks in L to be between the number of vertical blocks in A and the number of vertical blocks in B.

Finally, constraint (4.8) ensures that ranks of the vertical documents of the same vertical type t are adjacent numbers—that is, there are no gaps in the ranks.

The interleaving method then proceeds as described in Algorithm 4.

With these constraints, we require all aspects of the interleaved list (document placement in the regular result list, document placement inside the vertical block, size of the vertical block, position of the vertical blocks, and number of vertical blocks) to be somewhere between A and B. This allows for exploration of all possible rankings that can possibly be considered a combination of A and B.

It is worth noting that constraints (4.1) through (4.8) are more limiting than the original constraints in the conventional OI method. As a consequence, the problem of

ALGORITHM 4: Vertical-Aware Optimized Interleaving (VA-OI).

1: **function** VAOI(ranking A, ranking B, credit function δ_i)
2: $\mathcal{L} \leftarrow \{L \mid L$ satisfies (4.1)–(4.8)$\}$
3: Find a distribution $\{p_L \mid L \in \mathcal{L}\}$ that conforms to constraints (3.2), (3.3), and (3.4) and maximizes the average sensitivity (3.5).
4: Sample L from $\mathcal{L}$ according to the probability distribution p_L
5: **return** L

forming an interleaved list becomes overconstrained more often than for the conventional OI method: it happens about 10% of the time for VA-OI versus less than 0.1% for OI (experiments with the real dataset, which is introduced in Section 5.1.1). In cases where the problem becomes overconstrained, we relax it by replacing constraint (3.4) by the aggregated version:

$$\sum_{k=1}^{N} \sum_{i=1}^{|\mathcal{L}|} p_i \delta_k(L_i) = 0. \tag{4.9}$$

By doing so, we guarantee that the problem has a solution. Note that A and B are always present in $\mathcal{L}$, so we have at least two variables (p_1 and p_2) and at most two linear equations, Equations (3.3) and (4.9). We can also see that the interleaving method stays unbiased for the randomly clicking user, which we also confirm experimentally in Section 8.

We also considered altering the credit function to account for the fact that we are dealing with aggregated result pages. Vertical documents are often visually more salient than organic Web documents, which changes the examination probabilities and in some cases even the examination order. We take this into account by redefining the rank(d, X) function in (3.8) as rank(d, examination(X)), which represents the rank of document d in ranking X, where X is reordered in descending order by examination probability according to a click model. Preliminary experiments with this adapted credit function showed that it makes no change for the sensitivity and correctness as measured with the offline and online reference metrics (Sections 7.2 and 7.3). We did see the difference in sensitivity if Pareto dominance is used as ground truth (Section 7.1), but we treat this result as an artifact of our experimental setup, an artifact that stems from the fact that both the click model and Pareto dominance rely on the same examination order implied by the same click model.

To summarize, we have introduced two interleaving methods that respect vertical blocks within aggregated search scenarios. In Sections 6, 7, and 8, we evaluate these methods experimentally and show how they compare to conventional interleaving methods.

5. EXPERIMENTAL SETUP

Each of the interleaving methods that we study (Algorithms 1, 2, 3, and 4) takes two ranked lists as an input, generates an interleaved list, observes user clicks on it, and infers which system's rankings are better as its outcome. Therefore, to answer the three research questions listed in the Introduction (Section 1), we need two ingredients: pairs of ranked lists (rankings) and clicks. Next, we present different ways to obtain ranked lists and clicks that we will later use in our experiments. The breakdown of the experimental setups that we use is presented in Table I. We also present a table showing which experiments use which datasets (Table II). Basically, we aim to choose the most appropriate dataset(s) that provide most valuable insights to answer each specific experiment that we perform.

Table I. Characteristics of Different Experimental Setups (Documents, Queries, Verticals, and Systems)

Data	Documents	Queries (TREC Topics)	Verticals	Aggregated Search Systems	Description *Rankings*	Description *Clicks*
Real	2 months of log data	814 (5,755*) queries	1 vertical	2 systems	5.1.1	5.2.1
Model	TREC FedWeb13 data Demeester et al. [2013]	50 topics	12 verticals†	36 model systems	5.1.2	5.2.2
Synthetic	Artificial documents with simulated relevance	Any number of synthetic topics	3 verticals	Any number of synthetic aggregated search systems	5.1.3	5.2.2

* We have more queries for the relaxed setup (see Section 5.2.1).
† Verticals are manually classified from the 108 FedWeb sources. Details of the classification can be referred in Zhou et al. [2013b].

Table II. Experiments Performed to Answer Each Research Question and the Datasets Used

Research Question	Experiment	Section	Data *Real*	Data *Model*	Data *Synthetic*
RQ1: Influence on the user	Visual aspects	6.1	✓	✓	✓
	Offline quality	6.2		✓	
	Online quality	6.3	✓	✓	
RQ2: Correctness and sensitivity	Pareto dominance	7.1		✓	✓
	Offline metrics	7.2		✓	
	Online metrics	7.3		✓	
	A/B testing	7.4	✓		
RQ3: Unbiasedness	Random clicks	8.1	✓	✓	✓

5.1. Rankings

We use three different sources of rankings: from *real rankings* of a commercial search system through *model rankings* emitted by rather simplistic model aggregated search systems to completely *synthetic rankings* targeted to approximate realistic aggregated search result pages while having more control.

The reason to use different setups lies in the need to find a trade-off between *reality*, *variability*, and the *amount* of data that we have (Figure 3). Although desirable, we are not in possession of a single setup that would cover all aspects (dotted line in Figure 3). Instead, we have three different setups. First, real rankings allow us to assess interleaving methods on real-world ranking systems that serve real users and give us access to a good amount of data. However, variability of the data is low (e.g., we only have access to a single vertical and hence cannot assess our interleaving methods in a multivertical environment). Second, model rankings represent near real-world ranking systems that allow for greater variability, as we use simulated users and do not run the risk of hurting real users' experiences here. The data, however, is limited by the number of model aggregated search systems (36 in our case) and the number of TREC FedWeb topics (50 in our case). Last, synthetic rankings allow us to generate endless amounts of data with any variability that we want, but the relation to the real world is limited to what we explicitly include in our synthetic ranking generation procedure.

5.1.1. Real Rankings. We use a 2-month click log of the Yandex search engine recorded during winter 2012–2013. Due to limited access to the click log data, we were only able to collect data with one vertical block. We picked the vertical of Mobile Applications

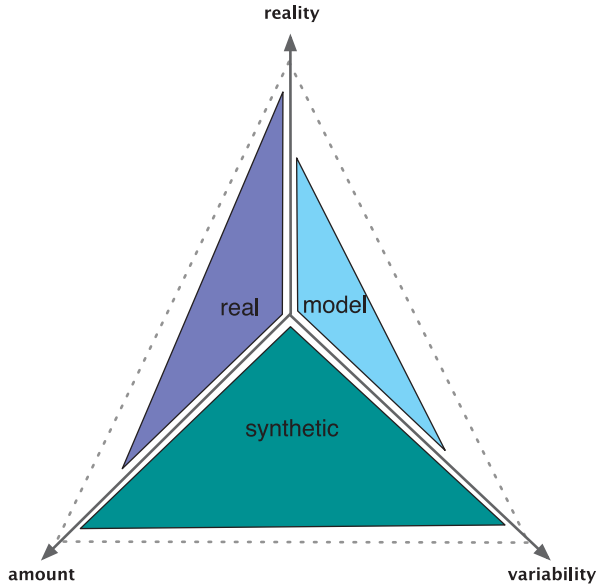

Fig. 3. Radar chart of the different data characteristics covered by the different experimental setups (real, model, and synthetic data). The farther from the center, the more pronounced this property is in a particular dataset. Note that for illustrative purposes, the intersection of the three axes denotes that all characteristics are low, but not zero. For example, model data exhibits relatively high variability and reality with amount of data being low.

Fig. 4. A vertical result item from the Mobile Applications vertical for the query "learn languages for free iphone."

because it is visually different from normal Web results and does not have additional aspects such as freshness (so we ruled out the News vertical) or horizontal document layout within a vertical block (that ruled out the Image and Video verticals). The Mobile Applications vertical is triggered for queries with an information need related to smartphone or tablet applications.[7] Although the algorithm used for information need detection is beyond the scope of this article, it is useful to name a few examples of queries triggering the appearance of this vertical. These include queries containing a mobile application name (e.g., "cut the rope") or explicit markers of mobile platforms (e.g., "Opera for iPhone"). The result items in the vertical block are presented using a standard text snippet, enhanced with an application icon, price, and rating. An example of such a snippet is shown in Figure 4.

One of the limitations of this approach is that we cannot experiment with completely new document orders that are disallowed by the current production ranking algorithms. In particular, we cannot reproduce the outcomes of an interleaving method that does not respect vertical block grouping. Another limitation is that we do not have data with multiple verticals and hence can only reproduce part of the analysis that we are able to conduct with model or synthetic rankings introduced later. On the other hand, even

[7]Currently limited to iOS or Android devices.

limited experiments with real data allow us to validate the key findings of the more comprehensive but less realistic setups.

5.1.2. Model Rankings. A broader but less realistic scope of rankings can be obtained by implementing model aggregated search systems. As in Zhou et al. [2013b], we implement near-real-world aggregated search systems that we can apply to rank documents and emit rankings with vertical blocks. The basic idea is that we implement state-of-the-art aggregated search components, and by combining different components, we develop a set of aggregated search pages with varying qualities.

First, we assume that an aggregated search page consists of 10 Web blocks (a single organic Web document is viewed as a block) and up to three vertical blocks dispersed throughout those 10 blocks (where each vertical block consists of a fixed number of three items).

Second, as described in previous work [Zhou et al. 2013b], we simulate a set of systems by implementing existing algorithms. We develop four state-of-the-art VS systems; two utilize vertical term distribution evidence and the other two use user intent information by mining query logs. For IS, we simulate three potentially different levels of relevance by using different ranking weighting schemes: Perfect (all vertical items are relevant), BM25 (with PageRank as a prior), and TF (with PageRank as a prior). We also simulate three approaches for RP: Perfect, Random, and Bad. Perfect places the vertical blocks on the page so that gain could potentially be maximized, whereas Bad reverses the perfectly presented page.

By utilizing this approach, we can generate $4 \times 3 \times 3 = 36$ aggregated search systems, which gives us $36 \times 35/2 = 630$ system pairs.

As a source of documents, we use the TREC FedWeb13 dataset [Demeester et al. 2013], which has topical relevance labels as well as vertical orientation labels—that is, the probability that the user prefers documents of this vertical type to the organic Web documents.

5.1.3. Synthetic Rankings. Although the model aggregated search systems allow us to have broader variety and a larger amount of data than real rankings, they still provide a limited setup. To make reliable judgments about the sensitivity of an interleaving method (RQ2) or about its unbiasedness (RQ3), we would like to be able to generate as much data as we need. For this purpose, we introduce a synthetic ranking generation procedure that does not require any document dataset or pool of aggregated search systems.

First, we want to mention that two ranking systems, when compared, may or may not differ in how they present vertical documents. In particular, the systems being compared may use the same method of building and presenting the vertical blocks and only differ in the nonvertical (organic) ranking. We believe that this is an important special case for search engine evaluation that deserves a separate analysis. Our early experiments with the real click log [Chuklin et al. 2013a, Table 4] showed that the outcome of the interleaving method is correlated with, but different from, the interleaving in situations where no verticals are distorting user clicks.

For the synthetic experiments, we design two different algorithms to simulate rankings: those with *fixed* vertical blocks (position and contents of the vertical blocks is the same in A and B rankings) and those with *nonfixed* vertical blocks (vertical blocks may have different contents and positions in A and B). These two settings also correspond to two different approaches to the construction of an aggregated search page. The first approach [Arguello et al. 2011a, 2011b] assumes that we have an organic Web ranking in place and that vertical documents are being added on top of it, regardless of the Web ranking. Another approach [Sushmita et al. 2010; Zhou et al. 2013b] assumes that the vertical documents also have topical relevance labels that can be compared to those of

the Web documents. This approach has started gaining popularity with the recently established TREC FedWeb track [Demeester et al. 2013; Nguyen et al. 2012].

For the *fixed* vertical blocks, we first generate a pair of organic rankings and then randomly insert blocks of vertical documents at the same place in both rankings. These blocks vary in size from zero to eight documents and in the positions of the blocks.[8] The ranking of the vertical documents within a block is always fixed (i.e., the same for both rankings). Vertical documents are either (1) nonrelevant (condition *nonrelevant*) or (2) a number of them, proportional to the relevant organic Web documents, are relevant (condition *relevant*).

For the *nonfixed* vertical blocks, we generate two lists that contain vertical documents straight away. We slightly modify the earlier procedure that we use for organic ranking generation: when constructing the ranking of all documents, both vertical and organic Web documents are generated; if the vertical documents end up not being grouped, we reorder them accordingly (see Algorithm 7).

When generating pairs of rankings A and B, we also ensure that the difference between a generated pair of rankings resembles the typical differences that we encounter in real-world interleaving experiments while allowing for ample variety. A particular procedure of generating synthetic ranking pairs is described in Appendix A.

5.2. Clicks

For clicks, we use both *real click* data from the 2-month click log described earlier (Section 5.1.1) and *simulated clicks* derived from a state-of-the-art click model for aggregated search. The click log setup is closely tied to the ranking setup: the real clicks can only be used with the real rankings, whereas the other ranking setups require modeling the user clicks with a click model.

5.2.1. Real Clicks. One problem that we face with click data is that we need to have clicks on the interleaved document list. If the particular interleaving method that we want to study has not been deployed to production, we do not necessarily have the clicks that we need. To address this problem, we adopt the setup proposed by Radlinski and Craswell [2013] that makes use of historic user clicks to evaluate new interleaving methods. The main idea of the method is to look at queries with sufficient variation in the ranked lists and then pretend that these different ranked lists are the interleaved lists for some rankers A and B.

Let us call a query together with a result list a *configuration*. Each configuration that has been shown at least once to a user counts as an *impression*. Each impression has zero or more clicks. We proceed in the following steps:

(1) Keep only impressions that have at least one click on their top N documents.
(2) Keep only configurations that have at least K such impressions.
(3) Keep only queries that have at least M such configurations.

After that, we name two configurations to be rankings A and B for each query. Following Radlinski and Craswell [2013], we call the most frequent configuration *ranking A* and the one that differs at the highest rank *ranking B*. In case we have several candidates for B, we choose the most frequent one. Once we have our rankings A and B, we compute all possible interleaved lists that can be produced by Algorithm 3 and proceed with the filtering:

[8]We randomly assign block positions based on the distribution obtained from the TREC FedWeb data [Demeester et al. 2013].

Table III. Filtering Parameters Setup

	N	K	M
Radlinski and Craswell [2013]	4	10	4
This article	10	4	2

(4) Keep only queries for which we have all interleaved lists that can be produced by VA-TDI in the log.[9]

To fully define the experimental setup, we have to define the parameters K, M, and N. We summarize the parameters that we use in Table III. Unlike Radlinski and Craswell [2013], we cannot use only the top four documents, as this is highly likely to be in between the vertical block. This is why we are forced to decrease K and M to have a sufficient amount of log data. With these parameters, we have 814 unique queries, which we use in all experiments in this section. If we relax the last filtering step and only require at least one interleaved list to be present in our query log, we obtain 5,755 queries with which to experiment (we consider the missing interleaved lists as ties). The reason to consider this *relaxed* setup is as follows: if we required all possible interleaved lists to be present in the click log, we would end up in a situation where only very similar rankings A and B are left, which is an additional bias that we want to avoid.

The main limitation of this approach is that we have a very limited amount of data, which is not the case for the click simulation (Section 5.2.2). We should also take into account that the data we get using this method is skewed toward a relatively small number of highly frequent queries. The variety of rankings is also limited by those extracted from the click log (Section 5.1.1)—we cannot combine real clicks with model or synthetic rankings.

5.2.2. Simulated Clicks. We simulate users' click behavior on an interleaved list using click models. Simulated users are always presented with the top 10 documents from the interleaved list. We have two types of user simulations.

Random user. The *random click model* assumes that users click on each document in the presented ranking with probability 0.5, such that—in expectation—half the documents are clicked. Relevance or presentation of documents is not taken into account. This model is used to answer RQ3.

Federated click model. The *multivertical federated click model* (mFCM) [Chuklin et al. 2014] is used to answer RQ1 and RQ2. This click model is designed to capture user behavior when result pages contain vertical documents of different vertical types. It assumes that user attention may be attracted to vertical documents as well as documents adjacent to a block. Formally, the model can be described as follows:

$$P(E_i = 1 \mid A) = \phi_i + (1 - \phi_i)\beta_i(A) \tag{5.1}$$

$$P(C_i = 1 \mid E_i = 0) = 0 \tag{5.2}$$

$$P(C_i = 1 \mid E_i = 1) = r_i \tag{5.3}$$

$$P(A_j = 1) = orient(vert_j, q) \cdot hpos_{vert_j}. \tag{5.4}$$

Here, C_i is a set of observed variables corresponding to a click on the i-th document; E_i is a hidden variable corresponding to a user's examination of the i-th document; and A is the vector of independent binary random variables A_j, attention bias values for each

[9]For historical reasons, we report only on VA-TDI, as it was the only interleaving method considered during the period when the authors had access to the click log.

vertical $vert_j$. Further, r_i is the relevance of the document (binary), and $orient(vert_j, q)$ is a vertical orientation given the query q to the vertical $vert_j$ (spans from 0 to 1). ϕ, β, and *hpos* are the model parameters that we instantiate as in Chuklin et al. [2014]. This model is an extension of the FCM model used in Chuklin et al. [2013a] for the case of multiple verticals. Unlike the simple model of position bias used in Hofmann et al. [2013b], we also take into account the fact that vertical blocks and adjacent documents draw additional attention of the user, and more so if the vertical orientation is high.

6. INFLUENCE ON THE USER EXPERIENCE (RQ1)

When one wants to evaluate an interleaving method, its effect on the user experience needs to be evaluated first. We formulate this as one of our research questions (RQ1) and conduct three experiments.

First, we look at the visual changes of the interleaved list in Section 6.1. Second, we evaluate the quality of the interleaved list as captured by offline editorial metrics and compare it to the quality of the systems being interleaved (Section 6.2). Finally, in Section 6.3, we perform a similar quality comparison using absolute click metrics based on data produced by real search engine users as well as by simulated users.

6.1. Visual Changes in Result Page

6.1.1. Setup. A good online evaluation method should be invisible to the user. In other words, the user should not notice any interface changes when interacting with an interleaved list as opposed to a base ranking system. We identified several particularly prominent features of aggregated search systems related to vertical blocks, namely the number of vertical blocks per vertical and the size of a vertical block, and track how they change when we use different interleaving methods. If an interleaving method shows smaller or bigger vertical blocks, or especially if it splits a vertical block into two (thus, increasing the number of blocks), then we consider this to affect the user, which we want to avoid.

As a source of rankings we use real, model, and synthetic rankings (see Sections 5.1.1, 5.1.2, and 5.1.3).

6.1.2. Results.

Synthetic rankings. We start with synthetic rankings (Section 5.1.3), as they allow us to vary the target block size of the ranking lists being interleaved. Namely, we can set the block size exactly if we use *fixed* vertical placement (see Algorithm 6, each vertical has a block of the specified size). For *nonfixed* placement, the target block size is an expected number of vertical documents per block, specified using the distribution q in Algorithm 7 (i.e., for target block size 2, we set $q_t = 2/10$ for each vertical t). We also considered the case of up to one vertical type and a multivertical case where up to three verticals are allowed. In the latter case, the smaller target block sizes are considered, because a result page of 10 documents cannot have more than 10 vertical documents, as shown later in Algorithm 7.

Figures 5 and 6 show our results for different numbers of blocks per vertical[10] and Figures 7 and 8 for vertical size. For VA-TDI, we see that the number of generated vertical blocks is typically close to and never higher than 1, as designed. Due to Equations (4.6) and (4.7), VA-OI always has exactly one vertical block per vertical if the fixed placement scheme is used. When the vertical blocks in the lists that we interleave are small, the block may not be included in the interleaved list at all, resulting in an

[10]Figure 5 differs from the corresponding figure in Chuklin et al. [2013a] (Figure 4 there) due to different settings of the synthetic generation procedure. Previous results can be obtained if we set $d = 0$ and $\tau = 0$—that is, two generated rankings are just two random permutations of the same documents. However, we believe that the settings we use in this article (Appendix A) are more realistic.

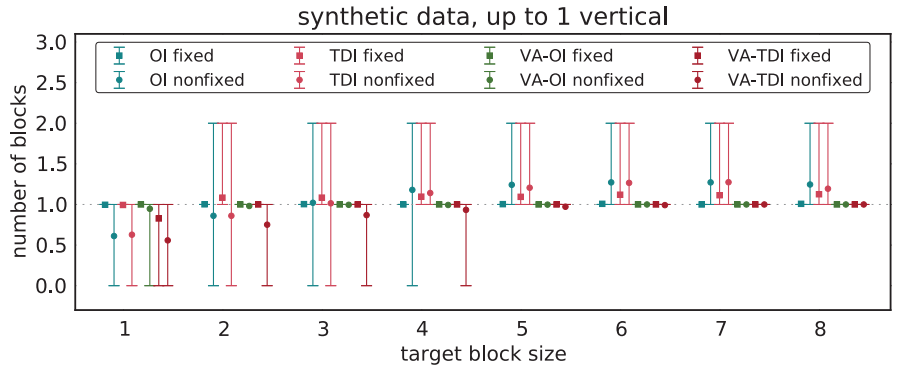

Fig. 5. Average number of vertical blocks per vertical; target block sizes 1 through 8, up to one vertical type. Error bars correspond to 5th and 95th percentiles.

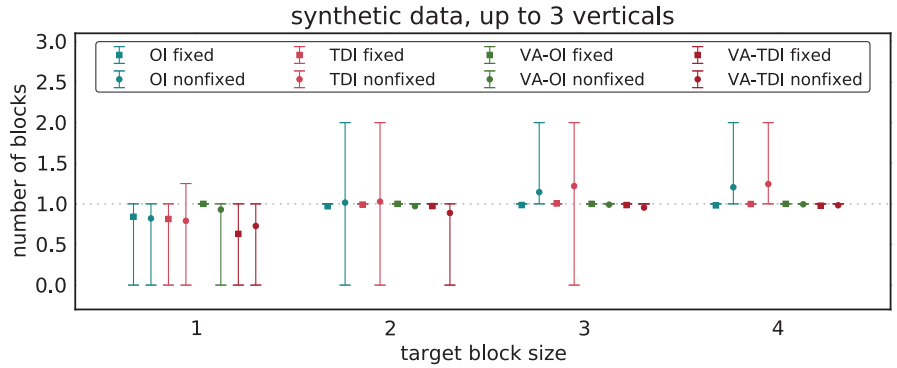

Fig. 6. Average number of vertical blocks per vertical; target block sizes 1–4, up to three different vertical types. Error bars correspond to 5th and 95th percentiles.

average number of blocks smaller than one (cf. Algorithm 3, line 22). This can also occur when the vertical blocks are placed in the lower halves of the original lists. For TDI, we observe different behaviors under fixed and nonfixed block placement. When the rankers that we compare have different vertical blocks (condition *nonfixed*), TDI tends to generate several smaller blocks that result in a higher average number of blocks. The same goes for OI.

All interleaving methods behave similarly with respect to block size. When two verticals have the same vertical block (fixed block placement), the block size can only become smaller after the ranked lists in which they are contained have been interleaved (values less than 1 in Figures 7 and 8) because some of the vertical documents are pushed outside the top 10. This effect is more visible when multiple vertical blocks from multiple verticals are present (Figure 8). When the verticals are generated using the nonfixed scheme (see Algorithm 7), the resulting block size to target block size ratio on average is the same or even smaller than in the fixed scheme. It also has a wider distribution. The reason is that the target block size is not always equal to the exact size of the corresponding vertical block in A and B rankers; in fact, A and B can have different block sizes.

Real and model rankings. Now we repeat the same experiments for the model and real datasets (Sections 5.1.2 and 5.1.3). In those cases, we no longer have control over the size, position, or contents of the vertical blocks in the A and B rankers, so we simply plot the overall picture that includes blocks of different sizes.

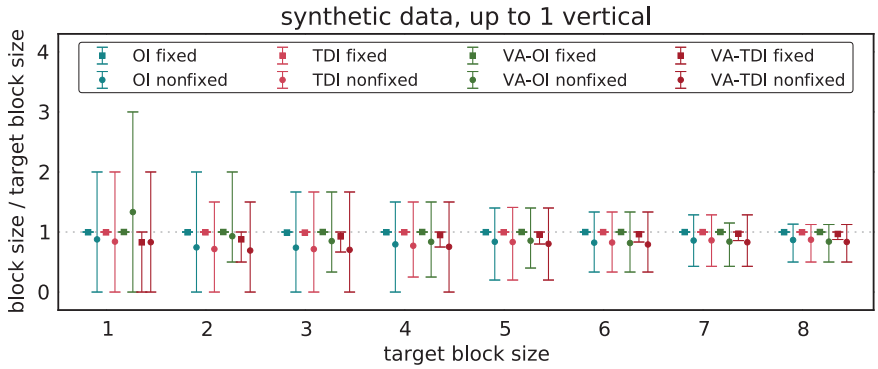

Fig. 7. Average vertical block size divided by the target block size; target block sizes 1 through 8, up to one vertical type. Error bars correspond to 5th and 95th percentiles.

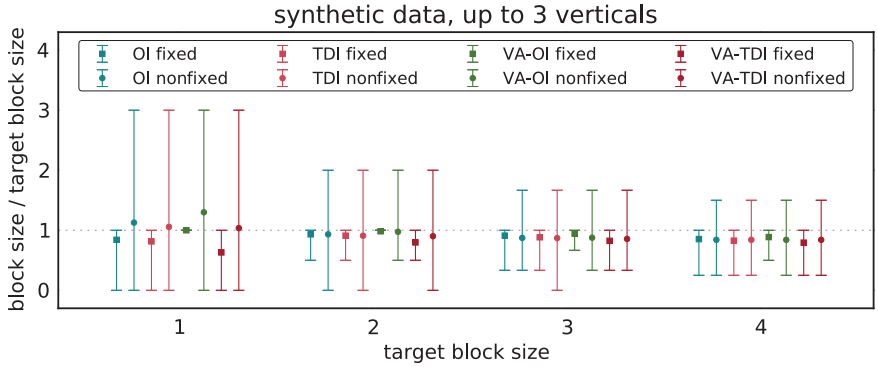

Fig. 8. Average vertical block size divided by the target block size; target block sizes 1 through 4, up to three different vertical types. Error bars correspond to 5th and 95th percentiles.

Figure 9 shows the results for the real data. We can see that VA-TDI only has slightly less than one block per vertical on average, whereas it never exceeds 1—that is, the vertical block is not broken. We can also confirm that VA-TDI results in smaller vertical blocks than the other interleaving methods (TDI, OI, VA-OI). VA-OI always yields one vertical block, as is guaranteed by Equations (4.6) and (4.7), and the fact that we only have one vertical in the real dataset.

Figure 10 reports similar results for the model data. We see the same pattern that VA-TDI tends to have smaller blocks and misses a block completely (number of blocks zero) more often than TDI and OI, which in turn often have a vertical block split up into two or even three smaller blocks. Now VA-OI also has less than one block per vertical on average, which is normal because A and B can have different verticals and not all of them should be present in the interleaved list. The difference between average number of blocks of OI and TDI is not observed for the real data but is observed for the model data, which suggests that the difference between these methods depends on the ranking systems being compared and number of different verticals present in the dataset (remember that real data rankings have only one vertical). In addition, vertical-aware methods (VA-TDI and VA-OI) are more conservative about the vertical blocks and yield smaller blocks and fewer blocks than conventional interleaving methods. We believe this to be a less visible and less disturbing effect on the user experience than splitting the block of the same vertical type.

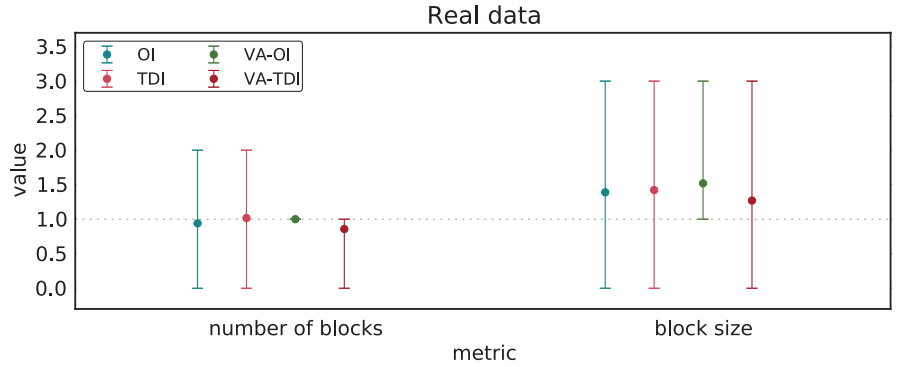

Fig. 9. Average number of vertical blocks per vertical and the vertical block size of each vertical; real data. Error bars correspond to 5th and 95th percentiles.

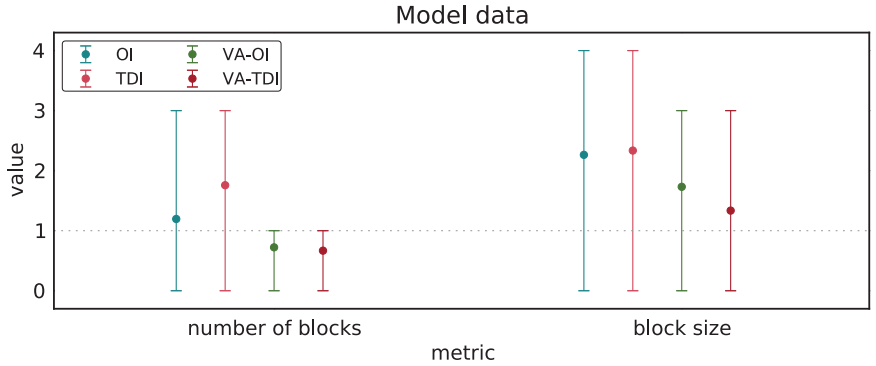

Fig. 10. Average number of vertical blocks per vertical and the vertical block size of each vertical; model data. Error bars correspond to 5th and 95th percentiles.

We conclude that VA-TDI and VA-OI keep vertical documents together, as designed. These methods produce up to one vertical block per result list, thus bounding the impact of interleaving on the user experience. When vertical blocks are placed independently, the impact of TDI and OI without vertical awareness is high. However, when blocks are placed at the same fixed positions, the impact is much lower, especially for OI (but still higher than for its vertical-aware extension, VA-OI).

6.2. Offline Quality Measures of the Interleaved Page

6.2.1. Setup. Our second and third sets of experiments involve comparing an interleaved system's performance to that of the systems A and B that are being interleaved. Ideally, we want an interleaving method to produce ranked lists that are not worse than those of A and B. We see it as one of the main contributions of our work and claim that every new interleaving method should be tested for that.

In this section, concerning offline quality measures, we only use model rankings (Section 5.1.2) since these rankers, unlike synthetic rankers, have topical relevance and vertical orientation labels that we can use for quality comparison.[11] We do not use real rankings, since only a small fraction of these documents have relevance judgments.

[11]More details on the topical relevance and vertical orientation labels available for the model rankings can be found in Zhou et al. [2012a].

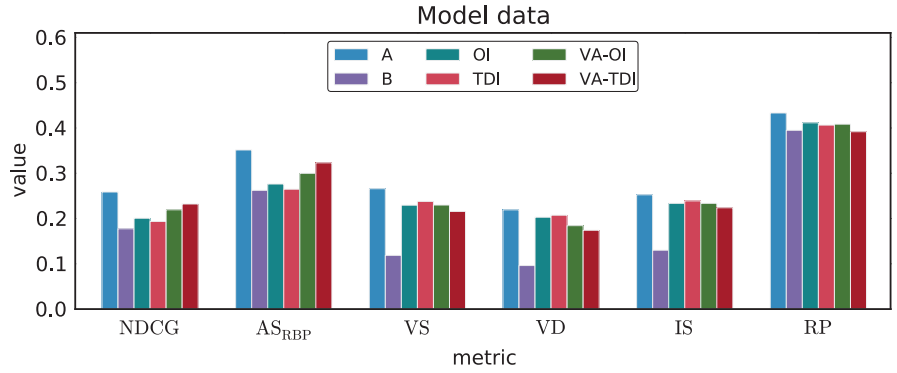

Fig. 11. Average value of offline quality measures for the interleaved lists and original rankings A (better system) and B (worse system).

Given two rankings A and B, we can compute a relevance-based quality measure for A, B and for the interleaved lists similar to He et al. [2009]. As our quality metrics, we use a classic evaluation metric, NDCG [Järvelin and Kekäläinen 2002], as well as a metric specific to aggregated search, AS_{RBP} [Zhou et al. 2012a]. Separately, we also analyze simple one-component aggregated search metrics [Zhou et al. 2013b] to verify that our interleaving procedure does not lead to degradation in any of the aspects of aggregated search (VS, VD, IS, RP). The choice of AS_{RBP} is motivated by the fact that this metric is the best at capturing all four aggregated search aspects (provided that they are all treated equally important) and at discriminating different aggregated search systems [Zhou et al. 2013b]. The simple single-component aggregated search metrics that we use are adopted from the aggregated search meta-evaluation study by Zhou et al. [2013b]; they independently reflect basic aggregated search properties and are as simple as possible to be agnostic to potential differences in the interleaving methods (e.g., the credit function choice):

(1) *VS metric*—selected vertical precision on relevant verticals with high vertical orientation (orientation values more than 0.5)
(2) *VD metric*—selected vertical recall on all available verticals
(3) *IS metric*—mean precision of retrieved vertical documents
(4) *RP metric*—Spearman's rank correlation with a "perfect" aggregated search reference page.

For each offline metric M and each pair of 36 model rankers (Table I), we first determine which ranker in the pair has a higher average value of M. This ranker is named A and the other one B. Ideally, our interleaved system should have an average value of quality metric M that lies in between those of A and B, respectively. We also perform a paired t-test (two-tailed 95%) to see how many interleaving experiments (out of $C_{36}^2 = 630$ experiments corresponding to pairs of model rankers) result in an interleaved systems being statistically significantly worse than B (i.e., the worst out of two systems). We do not perform multiple testing corrections, as we are not interested in the absolute values here. Instead, we compare the number of detected significant differences for different interleaving algorithms: if the null hypothesis is indeed true (i.e., two systems have the same quality), the odds of erroneously reporting N significant differences decreases with N (exponentially if we assume that the tests are independent).

6.2.2. Results. Figure 11 and Table IV summarize our results. All interleaving methods, on average, perform better than B and worse than A. It is also worth noting

Table IV. Percentage of Pairwise Comparisons (out of 630) Where the Interleaved Ranking Scores Are Statistically Significantly Lower Than *B* as Measured by the Offline Quality Measures

	NDCG	AS_{RBP}	VS	VD	IS	RP
OI	9.5%	13.2%	0.0%	0.8%	0.0%	0.5%
TDI	11.6%	21.4%	0.0%	3.3%	0.0%	0.2%
VA-OI	0.2%	0.8%	0.5%	0.3%	0.2%	0.5%
VA-TDI	0.2%	0.2%	0.0%	0.6%	0.2%	3.0%

that VA-TDI and VA-OI perform substantially better than TDI and OI if measured by the classic NDCG metric. We believe this to be an artifact of the model dataset (Section 5.1.2) that we are using. We noticed that in general, there are more relevant organic Web documents than there are relevant vertical documents [Demeester et al. 2013], and since vertical-aware methods produce smaller vertical blocks (Figure 10), and those blocks tend to be lower in the ranking than they are for TDI and OI, we conclude that this leads to higher NDCG scores.

Table IV, in particular, tells us that only a small fraction of the model systems, once interleaved using VA-TDI, would suffer from quality degradation. The same holds for VA-OI, although degradation in terms of AS_{RBP} is slightly higher. On the other hand, we will much more likely have a quality degradation for a big portion of system pairs if we use OI, and even more likely if we use TDI. As to the one-component metrics, VS, VD, IS, and RP, we only notice a small degradation of VD of the TDI rankings (possibly because some verticals are pushed out of the top 10 documents presented to the user) and RP of the VA-TDI rankings.

In summary, when evaluating offline, we conclude that our VA-TDI and VA-OI methods outperform TDI and OI in preserving user experience.

6.3. Online Quality Measures of the Interleaved Page

6.3.1. Setup. Next we repeat the experiments from the previous section, but instead of computing offline relevance-based metrics, we compute online click-based metrics. This type of analysis is less precise but can be used in situations when we do not possess relevance labels. This time we use the real and model datasets. For the real dataset (Section 5.1.1), we compute click metrics for all impressions of all interleaved lists L that appear in our click log (Section 5.2.1). For the model dataset (Section 5.1.2), we simulate user clicks using the mFCM click model (Section 5.2.2), repeated 50 times. The real dataset allows us to work with real user clicks, whereas the model dataset gives us variability. Since the amount of data is not an issue here, we do not need to use synthetic data in this experiment (cf. Figure 3).

As absolute metrics we use the metrics that are often used in A/B-testing experiments. We decided to use metrics that only require clicks and no additional information (like relevance judgments, user information, timestamps, or session information): Clicks@1, MaxRR, MeanRR, MinRR, PLC. These metrics were also used in the interleaving study by Chapelle et al. [2012]:

(1) *MaxRR, MinRR, MeanRR*—maximum, minimum, and mean reciprocal rank ($1/r$) of clicks.
(2) *PLC (precision at lowest click)*—the number of clicks divided by the rank of the lowest click. This is the opposite of *pSkip* used in Chapelle et al. [2012].
(3) *Clicks@1*—equals 1 if there was a click on the top 1 document, 0 otherwise.

We also add one vertical-specific metric, VertClick, which equals 1 if there was a click on a vertical document and 0 otherwise. Unlike previous metrics, this one measures less of the quality of the result page, but rather how much attention is being drawn to the vertical documents.

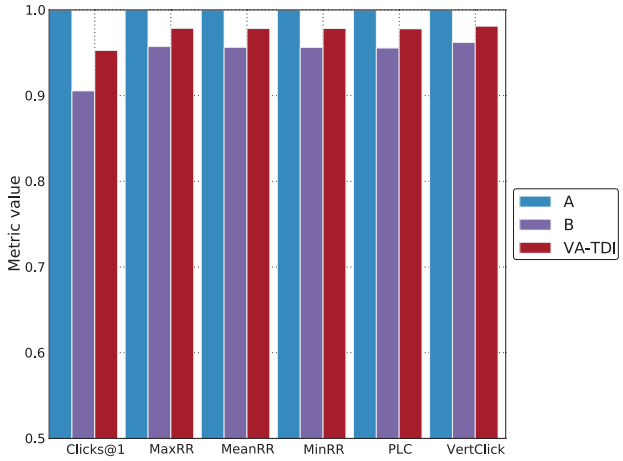

Fig. 12. Normalized scores for online click metrics for the rankings A and B and for the interleaved list obtained using VA-TDI (real dataset).

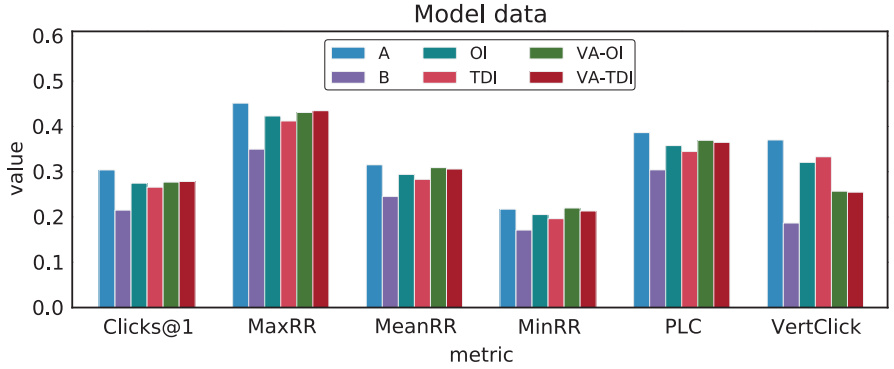

Fig. 13. Average online click metric scores for the rankings A and B and for the interleaved list obtained using different interleaving methods (model rankings, simulated clicks).

6.3.2. Results.

Real rankings. The results for the real dataset are summarized in Figure 12. As we are not interested in the absolute values of the metrics (and are not able to disclose them due to the proprietary nature of such information), we normalize all metrics to the corresponding values of the system A, which also happens to perform better than B according to all of those metrics.

We can see from the plot that the click metric values for the interleaved list L are between those of A and B. The differences between them are not statistically significant when using a 95% paired two-tailed t-test. We conclude that we do not have any degradation in user experience compared to the worst of two systems (B in this case). We should mention that we do have a degradation (not significantly) compared to the best system (A); however, this cannot be avoided, as we do not know which system is superior beforehand since finding this is the purpose of performing the evaluation in the first place.

Model rankings. Figure 13 shows that, on average, every interleaving method produces a system that scores between A and B according to all online metrics. We can see

Table V. Percentage of Pairwise Comparisons (out of 630 Model Ranking Pairs) Where the Interleaved Ranking Scores Are Statistically Significantly Lower Than *B* as Measured by Online Quality Measures

	Clicks@1	MaxRR	MeanRR	MinRR	PLC	VertClick
OI	7.1%	1.7%	4.9%	5.7%	4.3%	4.0%
TDI	7.6%	3.2%	6.8%	10.0%	8.1%	4.0%
VA-OI	4.9%	1.1%	0.8%	0.3%	0.5%	20.0%
VA-TDI	4.6%	0.5%	0.8%	1.6%	0.8%	27.9%

that VA-TDI and VA-OI are slightly better than TDI and OI according to all metrics except VertClick, where VA-TDI and VA-OI have substantially lower scores. This is also supported by Table V; like Table IV, it shows how many comparisons result in statistically significantly lower metric values for the interleaved system using a paired two-tailed 95% *t*-test. One explanation for the low VertClick values for VA-TDI and VA-OI is that they tend to produce smaller blocks (see Figure 10), so we have fewer vertical documents, and therefore they have a lower probability of being clicked.

In this section, we presented a thorough analysis of the influence that an interleaving method has on the user experience, on both visual and quality aspects. Such an analysis should be at the core of evaluating any new interleaving method, even though it was largely ignored until now. In answer to RQ1 about the influence of interleaving on the user experience, we demonstrated that the newly introduced vertical-aware interleaving methods not only preserve the user experience but also preserve the quality of the ranking systems being interleaved.

7. CORRECTNESS AND SENSITIVITY (RQ2)

Our second research question (RQ2) concerns the amount of data we need to be able to draw conclusions about the system quality (sensitivity) and how accurate this conclusion is (correctness). Typically, before a new ranking algorithm in a commercial search engine setting is launched to the public, it is compared to the previous version of the algorithm using interleaving or some other method. It is therefore crucial that we can quickly and accurately decide if the new ranking algorithm is better or worse than the current one.

In Sections 7.1, 7.2, and 7.3, we analyze how the fraction of *correctly* identified preference pairs depends on the number of user impressions. We first look at the strong cases where one ranker Pareto dominates the other (Section 7.1) and then analyze more relaxed settings where instead of Pareto dominance we use offline quality metrics like NDCG or AS_{RBP} (Section 7.2), or online metrics like MeanRR or Clicks@1 (Section 7.3). In Section 7.4, we consider typical 10-day experiments and see how often interleaving methods can come to a statistically significant decision in this time frame and how they compare to A/B-testing in terms of sensitivity.

7.1. Finding Strong Differences: Pareto Dominance

7.1.1. Setup. Our simulation approach is based on the experimental setup proposed by Hofmann et al. [2011, 2013a]. We first take two synthetic (Section 5.1.3) or model (Section 5.1.2) ranked lists, apply an interleaving method, and subsequently offer the interleaved list to a *simulated* user that produces clicks (Section 5.2.2). Then it is up to the interleaving method to select a winning ranker. Our experiment measures to what degree an interleaving method can detect differences in the quality of result lists. Specifically, we look at how the confidence about the preferred ranker depends on the number of user impressions (sensitivity) and what the final correctness level is. The fewer impressions are needed for an interleaving method to determine the winner, the more sensitive the method is.

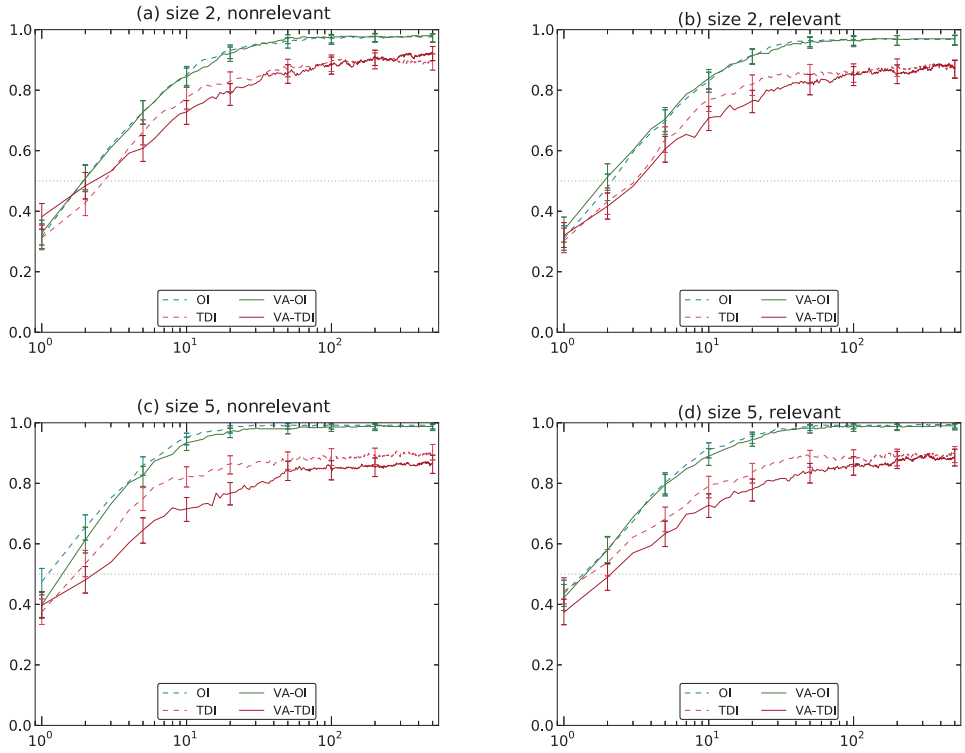

Fig. 14. Portion of correctly identified ranker preferences (vertical axis) by different interleaving methods after 1 to 500 user impressions (horizontal axis, log scale). The dashed horizontal line at 0.5 denotes random preference. All figures have independent block placement. Error bars correspond to 95% binomial confidence intervals. All rankings have one or zero verticals.

We repeat the process of generating two synthetic rankings (or randomly drawing a pair of model rankers and a query) until one ranking Pareto dominates[12] the other in terms of how it ranks relevant documents. Because the rankings are chosen in such a way that one dominates the other, we know which ranking should be preferred by an interleaving method.

We generate 500 pairs of rankings, with one ranking dominating the other, as described earlier. These pairs are each interleaved 500 times by different interleaving methods. For synthetic rankings, we repeat this process for several combinations of the conditions described in Section 5.1.3. We observe the portion of correctly identified ranking preferences (i.e., the accuracy) by each interleaving method. We calculate the mean and 95% binomial confidence bounds.

7.1.2. Results.

Synthetic rankings. We compare our vertical-aware interleaving methods (VA-TDI and VA-OI) to the non–vertical-aware baselines (TDI and OI) in terms of the accuracy of the identified ranker preferences. Figures 14 and 15 show the portion of correctly

[12]As in Hofmann et al. [2013a], we re-rank documents by examination probability $P(E_i = 1)$. In Hofmann et al. [2013a], $P(E_i = 1)$ is implicitly defined by the *cascade click model*. In our case, it is dictated by the mFCM model (Section 5.2.2); we marginalize over attention bias, using (5.1) and (5.4). We say that ranking *A dominates B*, if and only if re-ranked with $P(E_i = 1)$, *A* ranks all relevant documents at least as high as *B* and at least one relevant document higher than *B*.

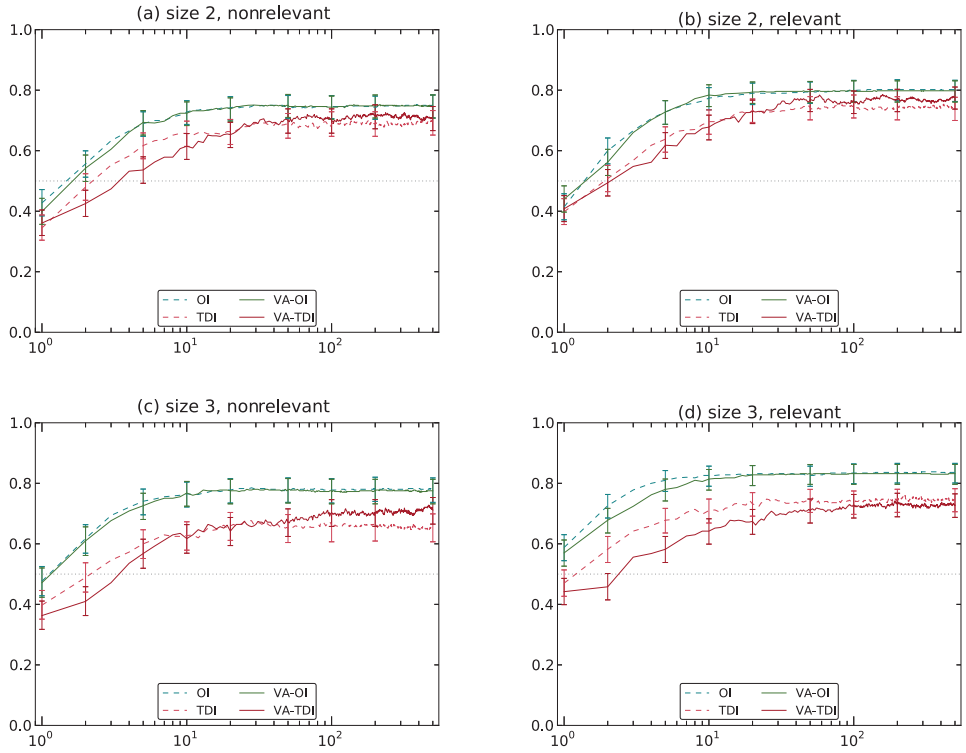

Fig. 15. Portion of correctly identified ranker preferences (vertical axis) by different interleaving methods after 1 to 500 user impressions (horizontal axis, log scale). The dashed horizontal line at 0.5 denotes random preference. All figures have independent block placement. Error bars correspond to 95% binomial confidence intervals. All rankings have up to three different verticals.

identified ranker preferences by TDI, OI, VA-TDI, and VA-OI after 1 to 500 user impressions modeled by the federated click model, mFCM (see Section 5.2.2).[13]

In Figure 14, we only consider cases where rankings are allowed to have verticals of one type (e.g., News), whereas in Figure 15, up to three different types of vertical are allowed.

Figure 14(a) shows results averaged over all possible positions of a block of size 2 (nonfixed block placement) under the assumption that none of the vertical documents are relevant. We see that TDI and VA-TDI converge to correctly identify about 90% of the true preferences correctly, whereas OI and VA-OI converge to about 98%. There is no significant difference in the number of impressions between conventional and vertical-aware methods.

Figure 14(b) shows results in the setting with relevant vertical documents. Results are almost identical to the case of nonrelevant vertical documents with only subtle differences. The difference is more visible when allowing more verticals (Figure 15(a) vs. (b))—the final converged accuracy level is higher when there are relevant vertical documents.

VA-TDI initially requires more sample data than TDI: TDI is significantly more accurate when we have a very small number of impressions (fewer than 10). We believe that the reason for this is the noise added by vertical documents dropping out—the

[13]In all of our experiments, all methods plateaued after several hundred user impressions.

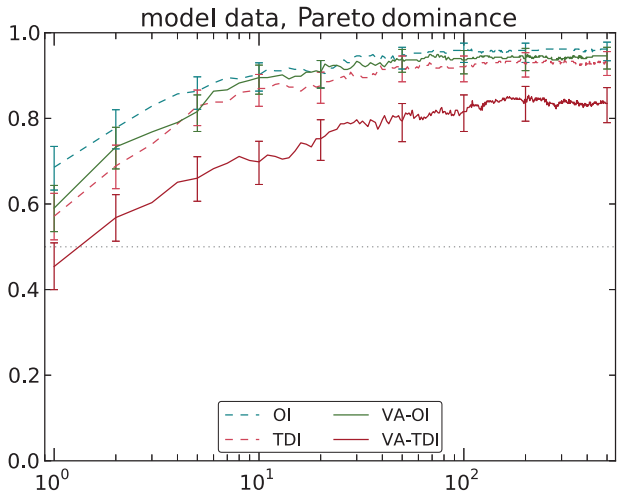

Fig. 16. Portion of correctly identified ranker preferences (vertical axis) by different interleaving methods after 1 to 500 user impressions (horizontal axis, log-scale). The dashed horizontal line at 0.5 denotes random preference. Error bars correspond to binomial confidence intervals.

problem discussed at the end of Section 4.1. Since this is noise and not bias toward either ranking, this levels out as the number of observed impressions increases. However, this need for more samples by VA-TDI is a small loss in efficiency for a method that preserves the original user experience as much as possible compared to TDI.

Figure 14(c) and (d) show results for the same conditions as (a) and (b), respectively, but with a block size of 5 instead of 2. The only difference with Figure 14(a) and (b) is the increased gap between TDI and OI families, which suggests that the latter should be preferred, especially in situations when we have many vertical documents. The same holds for Figure 15, where up to three different verticals are allowed.

Model rankings. Figure 16 shows the same experiment for the model data. This is a more realistic dataset than the synthetic one studied earlier. It contains many different vertical blocks of different types (see Table I). From the figure, we see that VA-TDI performs significantly worse than the other interleaving methods. As we explained at the end of Section 4.1, this is because vertical documents often drop out of the interleaved list, thereby limiting the sensitivity of VA-TDI. This is especially true if the difference between lists being interleaved is large, which is often the case for the model data.

7.2. Finding Weak Differences: Offline Metrics

7.2.1. Setup. In this section, we use a setup similar to the previous section, but instead of requiring that one ranker Pareto dominates another, we only require a nonzero difference in terms of an offline quality metric. This resembles the so-called live data simulations by Hofmann et al. [2013b], where interleaving methods are compared in terms of how well they can predict the direction of NDCG preference. On the one hand, offline metrics are less reliable as reference metrics than Pareto dominance. On the other hand, by using offline metrics, we evaluate our interleaving methods' ability to find weak differences between rankings as opposed to strong cases of Pareto dominance.

We picked NDCG and AS_{RBP} to use in our experiments. We also experimented with the one-component aggregated search metrics (VS, VD, IS, RP) but found that

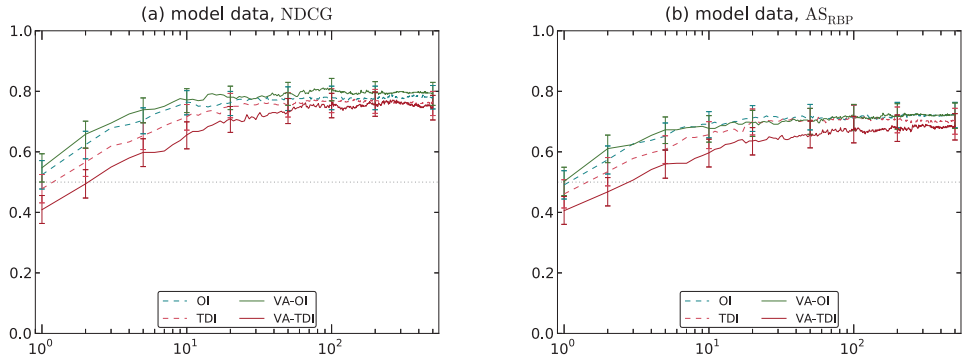

Fig. 17. Portion of ranker preferences that agree with the offline metric (vertical axis) by different interleaving methods after 1 to 500 user impressions (horizontal axis, log scale). The dashed horizontal line at 0.5 denotes random preference. Error bars correspond to binomial confidence intervals.

only IS differences can be identified by the interleaving methods; only for this one-component reference metric is the agreement level more than 0.5. This is because the one-component metrics are too simple to be used as reference metrics for system preference. One system can be better according to a one-component metric, but worse overall. Interleaving, in turn, considers the ranked list as a whole and is not required to agree with one-component metrics.

Here we only use model rankings (Section 5.1.2), as this is the only dataset that has meaningful relevance and vertical orientation labels.

7.2.2. Results. Results are presented in Figure 17. The error bars are big, and it is hard to state that some method is more accurate than another, with the only exception of VA-TDI, which converges much slower than VA-OI. In fact, for the first 10 impressions with the NDCG reference metric (Figure 17(a)), the error bars do not overlap, indicating that VA-OI identifies significantly more correct ranker preferences than VA-TDI does. A similar picture is observed for AS_{RBP} (Figure 17(b)).

We also observe that all of the interleaving methods agree better with the NDCG reference metric than the with AS_{RBP}. This demonstrates that it is more challenging to use interleaving methods to derive the preference of aggregated search pages. The loss of agreeing with AS_{RBP} might be because the assumptions made in assigning credits of documents in interleaving methods align better with the NDCG position-based discount than with the more complex assumptions of AS_{RBP}. It may also stem from the fact that the mFCM click model we utilize still assigns more value to topical relevance than to vertical orientation: Equations (5.2) and (5.3) guarantee that only topically relevant documents are clicked, whereas vertical orientation simply increases the chances of examining some documents.

7.3. Finding Weak Differences: Online Metrics

7.3.1. Setup. In this section, we reuse the setup from the previous section but use online click metrics as reference metrics instead of offline metrics: we identify the better ranking by comparing average values of an online metric from 50 query sessions, simulated using the mFCM click model (Section 5.2.2). We picked two online metrics: the very simple Clicks@1 metric and position-aware MeanRR. These are also the metrics that, as we will see later, do not have statistically significant disagreement with the interleaving in the real A/B-testing settings (Section 7.4). Since the metrics values now vary more, we increased the number of comparisons to 10,000 to reduce the churn due

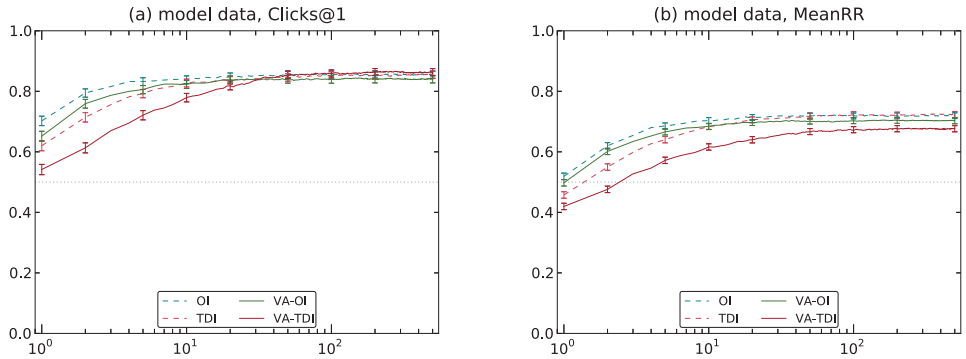

Fig. 18. Portion of ranker preferences that agree with the online metric (vertical axis) by different interleaving methods after 1 to 500 user impressions (horizontal axis, log scale). The dashed horizontal line at 0.5 denotes random preference. Error bars correspond to binomial confidence intervals.

to randomness present in the metrics. This also helped us overcome problems stemming from the fact that Clicks@1 is a binary metric and hence is often the same for A and B (we exclude such cases).

As in the previous section, we only use model rankings (Section 5.1.2) because the click models heavily rely on the relevance labels and it is crucial for these relevance labels to be meaningful. Our preliminary experiments with synthetic rankings showed that the click metrics are very noisy in that case, and all interleaving methods only marginally agree with them.

7.3.2. Results. Results are shown in Figure 18. We see that all interleaving methods converge to the same level of accuracy of about 0.85 as measured by the Clicks@1 reference metric. The agreement between the more complex MeanRR reference metric and the interleaving methods is less, but it confirms the previous finding (see Figure 16) that VA-TDI is significantly less discriminative than the other interleaving methods.

7.4. Time-Constrained Experiments and A/B-Testing

7.4.1. Setup. In a real-world setting of a search engine evaluation experiment, the experiment is run for a certain period of time within which our interleaving method needs to reach a conclusion. In this section, we reproduce such a time-constrained experiment setting using our real dataset (Sections 5.1.1 and 5.2.1).

One issue of live experiments is that we do not have ground-truth labeling of the better system. Usually, online click metrics are used to find the better system (e.g., in A/B-testing experiments), but these metrics may contradict each other or lack statistical significance. Therefore, one should not blindly use them as ground truth, but rather should examine the full picture of their agreement and disagreement with the interleaving outcomes and with each other.

Given the click log data to which we have access (Section 5.2.1), we compute the following values for each query:

(1) The average difference of the absolute click metrics for rankings A and B.
(2) The interleaving score[14] for each impression of each ranking implied by VA-TDI. As some configurations might have more impressions in our click log than the interleaving method suggests, we normalize the scores to the correct probabilities as implied by the interleaving method. Specifically, we compute the average score

[14]We assume that the score is 1 if ranking A wins a particular impression, -1 if B wins, and 0 if they tie.

Table VI. Agreement between *A*/*B*-Testing Measures and VA-TDI

Measure	t_1	t_2	t_3	t_4	t_5	t_6
Absolute Metrics						
Clicks@1	B$\diamond$	B	B	A	A	B
MaxRR	A	B	B	A	A	B
MeanRR	A$\diamond$	B	B$\diamond$	A	A	B
MinRR	A$\diamond$	B	A$\diamond$	A	A	B
PLC	A	B	B$\diamond$	A	A	B
VertClick	B	B	B	B	B	B
VA-TDI (different variants)						
Total	B	B	B	A	A	B
Organic only	B	B	B	A	A	B
Vertical only	A	A$\diamond$	A$\diamond$	B	B	B

Note: All changes are statistically significant except for ones marked by $\diamond$.

for each configuration, multiply it by the probability of such a configuration, and then average across all found configurations similar to Radlinski and Craswell [2013].

(3) The interleaving scores for vertical documents and nonvertical (organic) documents separately—that is, interleaving scores computed while only taking vertical (nonvertical) clicks into account.

To compare the direction of preference indicated by the absolute click metrics and the interleaving methods, we split the data into six buckets corresponding to six equal time periods of 10 days (t_1, t_2, t_3, t_4, t_5, t_6) and compute the weighted average of the absolute metrics and interleaving methods. The outcome for each impression (positive if *A* wins, negative if *B* wins, zero if it is a tie) is multiplied by the total frequency of the query[15] and summed up over all queries. We report the winning system according to each measure in Table VI. Note that we consider three ways of interpreting clicks in the interleaving method: *total*—all clicks are counted, *organic only*—only the clicks on nonvertical documents are taken into account, and *vertical only*—only the clicks on vertical documents are counted. For example, if we want to evaluate only changes in the organic ranking, we may want to look at the *organic only* results. On the other hand, we risk having an unbalanced team assignment if we skip the vertical block.

We also computed a per-query correlation between interleaving and A/B-testing metrics and performed a detailed analysis of the influence of the vertical block on this correlation (*total* vs. *organic only*). This analysis can be found in our previous work [Chuklin et al. 2013a, Section 4.2.2].

7.4.2. Results. Table VI shows that in most cases, VA-TDI (*total* and *organic only*) agrees with the majority of the absolute metrics. Similar to what was reported in Radlinski et al. [2008], the cases of disagreement between absolute click metrics and interleaving outcomes are always accompanied by the lack of statistical significance. We mark such cases where the winning system is *not* statistically significantly better with the $\diamond$ sign.[16] We can also note that the agreement between the *vertical-only* VA-TDI and the VertClick absolute metric is low. However, in two of the three cases with disagreement VA-TDI does not detect a statistically significant preference. This means that either VertClick is too simple to correctly capture the ranking changes or *vertical-only* is not the right way to interpret interleaving outcomes (or both).

[15]Remember that we previously normalized configurations to the correct probabilities.

[16]Here we use a bootstrap test with 1,000 bootstrap samples and a significance level of 99%.

In this section, we analyzed the sensitivity and correctness of conventional and vertical-aware interleaving methods under different settings and with different datasets. We showed that vertical-aware interleaving methods are in most cases as accurate as their conventional counterparts with the exception of VA-TDI, which in some rare cases may be less sensitive than TDI because of vertical documents dropping out. Some of our findings support the fact that OI and VA-OI are more sensitive and more accurate than TDI and VA-TDI, whereas others suggest that they are at least as sensitive as TDI and VA-TDI. We also confirmed that interleaving methods are more sensitive than A/B-testing and demonstrated that these two approaches often disagree with each other.

8. UNBIASEDNESS (RQ3)

Our third research question (RQ3) concerns unbiasedness of the interleaving methods. We investigate whether any of the interleaving methods identifies preferences for a ranker when there is no evidence in the data. For this purpose, we employ randomly clicking users to show that there is no systematic bias in the algorithm.[17]

8.1. Randomly Clicking User

8.1.1. Setup. Our final simulated experiment assesses the unbiasedness of different interleaving methods under random clicks (Section 5.2.2). It is important that accounting for vertical documents does not introduce bias, as it may otherwise lead to wrong interpretations of interleaving results. VA-TDI and VA-OI (Algorithms 3 and 4) were designed to be unbiased under many forms of noise; here, we validate that our implementation does indeed fulfill this requirement.

Under the random click model (Section 5.2.2), an unbiased interleaved comparison method should not systematically prefer either ranker—that is, the rankers should tie in expectation. We measure this following the methodology proposed by Hofmann et al. [2013a] by counting the number of comparisons for which a method detects a significant preference toward one of the rankers. For an unbiased method, this number should be close to the number expected due to noise. For example, a significance test with a p-value of 0.05 should detect statistically significant differences between rankers under random clicks in 5% of the comparisons. We repeat the test for different document datasets (synthetic, model, real) for different numbers of impressions (from 100 to 500).

8.1.2. Results. Tables VII, VIII, and IX show the results—the percentage of detected significant differences—for synthetic, model, and real datasets, respectively. For all methods and all datasets, we see that the number of significant differences detected is in line with the expected 5%.

Using a one-tailed binomial confidence test (also with $p = 0.05$), we confirm that this number is only once significantly higher than 5%, in the case of OI with synthetic rankings and up to three verticals (nonrelevant, fixed; see Table VII). We also see that the number of detected significant differences does not increase with the number of impressions, which confirms that all interleaving methods are indeed unbiased and can be relied on.

In this section, we verified that none of the interleaving methods exhibit a bias under the random click model. We performed an analysis with different datasets and confirmed that there is no bias under an assumption of randomly clicking user. There could be other types of biases that we do not analyze in the current work.

[17]One may also test unbiasedness under other user models that are blind to results, such as the pure position-based model.

Table VII. Percentage of Significant Differences between Rankers Detected under the Random Click Model for Synthetic Rankings

	Up to 1 Vertical															
	Nonrelevant								Relevant							
	Fixed				Nonfixed				Fixed				Nonfixed			
	OI	TDI	VA-OI	VA-TDI	OI	TDI	VA-OI	VA-TDI	OI	TDI	VA-OI	VA-TDI	OI	TDI	VA-OI	VA-TDI
100	5.2%	4.0%	4.6%	4.6%	4.8%	6.2%	4.4%	5.0%	5.8%	5.4%	4.6%	5.8%	5.2%	6.0%	4.6%	4.0%
200	6.6%	5.6%	3.6%	6.0%	4.2%	5.4%	4.6%	6.6%	4.6%	3.0%	3.6%	2.6%	5.0%	4.6%	5.8%	6.2%
300	5.8%	5.8%	4.6%	4.4%	4.8%	5.2%	4.4%	4.4%	5.0%	5.0%	3.6%	4.0%	3.8%	5.4%	4.2%	5.6%
400	6.6%	5.6%	3.6%	3.6%	5.2%	5.0%	3.8%	5.4%	3.6%	3.4%	3.8%	5.0%	4.8%	5.2%	4.6%	7.4%
500	5.6%	5.2%	4.6%	4.4%	4.6%	5.2%	4.0%	5.0%	4.6%	5.0%	4.2%	5.0%	5.4%	4.4%	4.8%	5.6%
	Up to 3 Verticals															
	Nonrelevant								Relevant							
	Fixed				Nonfixed				Fixed				Nonfixed			
	OI	TDI	VA-OI	VA-TDI	OI	TDI	VA-OI	VA-TDI	OI	TDI	VA-OI	VA-TDI	OI	TDI	VA-OI	VA-TDI
100	5.2%	5.0%	4.6%	4.8%	3.4%	5.4%	5.0%	6.2%	4.6%	5.6%	5.8%	5.4%	5.6%	5.6%	3.6%	5.6%
200	4.6%	4.2%	4.2%	5.6%	3.6%	4.8%	5.0%	4.4%	4.2%	5.4%	7.0%	5.4%	6.2%	3.8%	4.6%	5.8%
300	6.4%	4.0%	4.4%	4.6%	4.2%	5.0%	5.2%	4.8%	4.4%	4.0%	5.0%	5.4%	6.2%	4.6%	5.2%	5.6%
400	6.2%	4.2%	5.6%	4.8%	4.4%	5.6%	5.0%	6.0%	4.8%	4.4%	5.0%	4.2%	5.0%	5.0%	5.4%	4.8%
500	7.4%*	4.4%	6.8%	5.4%	4.0%	4.2%	4.8%	4.6%	5.4%	4.4%	3.8%	3.6%	6.2%	4.0%	5.4%	4.6%

Note: $p < 0.05$ on 500 ranker pairs after 100 to 500 user impressions (left-most column) under all combinations of conditions for block size 2. With $p < 0.05$, an interleaving method is expected to detect around 5% significant differences. An asterisk (*) marks outcomes that are statistically significantly higher than 5% using a one-tailed binomial confidence test (with significance level set to 0.05).

Table VIII. Percentage of Significant Differences between Rankers Detected under the Random Click Model for Model Rankings

	OI	TDI	VA-OI	VA-TDI
100	4.4%	6.2%	6.2%	5.8%
200	3.8%	6.4%	6.4%	5.2%
300	6.0%	6.2%	5.2%	5.4%
400	4.8%	4.6%	4.8%	6.6%
500	4.4%	5.2%	4.8%	4.4%

Note: $p < 0.05$ on 500 ranker pairs after 100 to 500 user impressions (left-most column). With $p < 0.05$, an interleaving method is expected to detect around 5% significant differences. None of the outcomes are statistically significantly higher than 5% using a one-tailed binomial confidence test (with significance level set to 0.05).

Table IX. Percentage of Significant Differences between Rankers Detected under the Random Click Model for Real Rankings

	OI	TDI	VA-OI	VA-TDI
100	4.4%	4.0%	5.2%	6.0%
200	5.6%	7.8%	5.8%	4.2%
300	5.6%	5.8%	7.2%	5.4%
400	5.8%	6.2%	5.6%	5.4%
500	5.0%	5.0%	5.8%	5.0%

Note: $p < 0.05$ on 500 ranker pairs after 100 to 500 user impressions (left column). With $p < 0.05$, an interleaving method is expected to detect around 5% significant differences. None of the outcomes are statistically significantly higher than 5% using a one-tailed binomial confidence test (with significance level set to 0.05).

9. CONCLUSION

To assess different interleaving methods under a wide range of conditions, we have addressed three main research questions about the influence of interleaving methods on the user experience, about the correctness and sensitivity of interleaving methods, and about their unbiasedness, all with a special focus on vertical-aware interleaving. We have done so using both simulations and real clicks. With simulated rankings, we have tested several vertical block sizes, several block placements, and different levels of relevance within the block.[18]

To summarize, we have shown the following:

—Vertical-aware interleaving methods do not degrade the user experience, as they do not break vertical blocks (Section 6.1) and do not degrade the quality of the result page (Sections 6.2 and 6.3).
—Vertical-aware interleaving methods can find differences between rankings as fast and as accurate as their conventional counterparts, with the OI family being substantially more sensitive (Sections 7.1, 7.2, and 7.3). We also demonstrated that in a 10-day experiment, the VA-TDI is able to reach a conclusion, whereas A/B-testing metrics often contradict each other (Section 7.4).
—Both vertical-aware and conventional interleaving methods are unbiased under random clicks (Section 8.1).

We have shown that vertical-aware interleaving methods can accurately compare result lists while preserving vertical blocks and quality level. Accuracy is as high as for conventional interleaving methods, with only small losses in efficiency for small sample sizes. We also confirmed that methods based on optimized interleaving (OI, VA-OI) usually outperform methods based on team draft (TDI, VA-TDI). Based on an extensive analysis, we conclude that vertical-aware interleaving methods (VA-TDI, VA-OI) should be used for comparing two ranking systems in situations where vertical documents are present.

One big limitation of our work is that not all of the experiments were validated in a real search setting. We should note, however, that in cases when we did perform experiments with real click log data, we found them to be in line with the findings reported using synthetic or model data. Another limitation is that for unbiasedness (RQ3), we only considered the case where *the user* clicks randomly but did not analyze the case where *the ranking systems* are indistinguishable. The problem with this is that it is hard to define what *indistinguishable* systems are. One may even claim that real-world systems are always different; we just do not always have enough data or the right instruments to observe this.

As future work, it would be interesting to apply our analysis of the influence on the user experience as presented in Section 6 (RQ1) to the online learning to rank problem of *balancing exploration and exploitation* (e.g., see Hofmann et al. [2012]). Currently, in formulations of the dueling bandits problem for online learning to rank [Yue and Joachims 2009], the regret function (how much the users lose by using a suboptimal system) of an interleaving method is set to an average of the relative quality degradation by A and B. As we saw in Section 6, the quality of an interleaved list can be far from that average; moreover, it depends on the interleaving method used and the ranking systems being interleaved.

Another direction of future work concerns an analysis of A/B-testing methods for aggregated search and their comparison to the interleaving methods presented here. In our work, we considered only one very simple online A/B-testing metric (VertClick),

[18]The code of our simulation experiments, including a reference implementation of the vertical-aware interleaving methods, is publicly available at https://bitbucket.org/ilps/lerot [Schuth et al. 2013].

but one can go further and adapt conventional online metrics to aggregated search settings or present new click metrics or A/B-testing procedures.

A. SYNTHETIC RANKING PAIR SIMULATION PROCEDURE

First we describe how we simulate ranking pairs for organic Web documents with vertical documents inserted afterward (condition *fixed*).

ALGORITHM 5: Generate a pair of organic Web rankings.

```
1: function GenerateRankingPair(d, N_{R,max})
2:     documentPool ← GenerateDocumentPool(10 + d, N_{R,max})
3:     A ← GenerateRanking(documentPool, 10)
4:     B ← GenerateRanking(documentPool, 10)
5:     return (A, B)

6: function GenerateDocumentPool(N', N_{R,max})
7:     N_R ← RandomInteger(1, N_{R,max})
8:     return document list of length N' with N_R relevant documents

9: function GenerateRanking(X, N)
10:    initialize s_X(d) using (A.1)
11:    L ← []
12:    while |L| < N do
13:        d_next ← SampleWithoutReplacement(s_X(d))
14:        L ← L + d_next
15:    return L
```

The algorithm for `GenerateRankingPair` (Algorithm 5) consists of several steps. To generate two ranked lists of size 10, we first generate a document pool of slightly bigger size $(10+d)$ and then draw documents from this pool to produce rankings A and B. The reason is that rankings A and B can differ not only in the order of the documents but also one of them can have some additional documents that the other one does not have. To allow different document orders in A and B, we employ a softmax distribution s_X by Hofmann et al. [2011] in which the probability of selecting the next document is inversely proportional to a power of the rank $r_X(d)$ of a document d in a document pool X:

$$P_{s_X}(d) = \frac{\frac{1}{r_X(d)^\tau}}{\sum_{d' \in X} \frac{1}{r_X(d')^\tau}}. \tag{A.1}$$

In this distribution, the document from the bottom of the list, X, has lower probability of being selected at each step of generating a ranking (line 13, Algorithm 5) than one of the top documents. It leads to ranking lists A and B being quite similar to each other (which is usually the case where B is a small experimental ranking change), and we can control the degree of similarity by varying the τ parameter.

For our experiments, we set $N_{R,max} = 3$ as in Chuklin et al. [2013a]. We also set $d = 2$ and $\tau = 5$ to resemble the level of difference between A and B that we have in the real interleaving experiments.[19]

[19]97% of the real ranking pairs (Section 5.1.1) have no more than two distinct document pairs, so we choose $d = 2$. The parameter τ is chosen such that the percentage of cases where A and B have 0, 1, or 2 different document pairs resembles that of the real ranking pairs that we analyze.

ALGORITHM 6: Generate a pair of rankings with the fixed vertical blocks *vBlocks* (condition *fixed*).

function GENERATERANKINGPAIRWITHVERTICALSFIXED(d, $N_{R,max}$, *vBlocks*, *vPositions*)
  (A, B) ← GENERATERANKINGPAIR(d, $N_{R,max}$)
  insert *vBlocks* to A at positions *vPositions*
  insert *vBlocks* to B at positions *vPositions*
  return (A, B)

ALGORITHM 7: Generate a pair of rankings with vertical documents (condition *nonfixed*).

function GENERATERANKINGPAIRWITHVERTICALSNONFIXED(d, $N_{R,max}$, q)
  documentPool ← GENERATEDOCUMENTPOOLWITHVERTICALS($10 + d$, $N_{R,max}$, q)
  A ← GENERATERANKINGWITHVERTICALS(*documentPool*, 10)
  B ← GENERATERANKINGWITHVERTICALS(*documentPool*, 10)
  return (A, B)

function GENERATEDOCUMENTPOOLWITHVERTICALS(N', $N_{R,max}$, q)
  N_R ← RANDOMINTEGER(1, $N_{R,max}$)
  $L_{\text{pool}} \leftarrow []$
  while $|L_{\text{pool}}| < N'$ **do**
    d_{next} ← document of type t according tothe distribution q(A.2)–(A.3)
    $L_{\text{pool}} \leftarrow L_{\text{pool}} + d_{\text{next}}$
  assign exactly N_R documents in L_{pool} to be relevant
  return L_{pool}

function GENERATERANKINGWITHVERTICALS(X, N)
  initialize $s_X(d)$ using (A.1)
  $L \leftarrow []$
  while $|L| < N$ **do**
    d_{next} ← SAMPLEWITHOUTREPLACEMENT($s_X(d)$)
    $L \leftarrow L + d_{\text{next}}$
  for each vertical type t **do**
    reorder documents of type t directly belowthe highest document of type t
  return L

If the vertical blocks are fixed, then synthetic ranker is generated in two steps (Algorithm 6). First, organic rankings are generated using Algorithm 5; second, vertical block is inserted at fixed position. Note that if the organic list has 10 documents, then inserting a vertical block can only increase the total document count, so there will be more than 10 documents.

Now let us describe the procedure of generating pair of rankings that contain vertical documents (condition *nonfixed*). We assume that we have a list of vertical types $\{v_t \mid t \in \{1, \ldots, T\}\}$ and a probability distribution q such that for each document d and each vertical type t, document d has type v_t with probability q_t. We also require $\sum_{t=1}^{T} q_t < 1$ so that the probability of the fact that d is a Web (nonvertical document) can be set to $(1 - \sum_{t=1}^{T} q_t)$:

$$\forall t \; P(vert(d) = v_t) = q_t. \tag{A.2}$$

$$P(vert(d) = Web) = 1 - \sum_{t=1}^{T} q_t. \tag{A.3}$$

The procedure is presented in Algorithm 7.

ACKNOWLEDGMENTS

We would like to thank Floor Sietsma for her help on an earlier version of this article and our anonymous reviewers for their comments and suggestions.

REFERENCES

Jaime Arguello, Fernando Diaz, Jamie Callan, and Jean-François Crespo. 2009. Sources of evidence for vertical selection. In *Proceedings of SIGIR*. ACM, New York, NY, 315–322.

Jaime Arguello, Fernando Diaz, and Jamie Callan. 2011a. Learning to aggregate vertical results into Web search results. In *Proceedings of CIKM*. ACM, New York, NY, 201–210.

Jaime Arguello, Fernando Diaz, Jamie Callan, and Ben Carterette. 2011b. A methodology for evaluating aggregated search results. In *Proceedings of ECIR*. 141–152.

Olivier Chapelle, Thorsten Joachims, Filip Radlinski, and Yisong Yue. 2012. Large-scale validation and analysis of interleaved search evaluation. *ACM Transactions on Information Systems* 30, 1, Article No. 6.

Danqi Chen, Weizhu Chen, Haixun Wang, Zheng Chen, and Qiang Yang. 2012. Beyond ten blue links: Enabling user click modeling in federated Web search. In *Proceedings of WSDM*. ACM, New York, NY, 463–472.

Aleksandr Chuklin, Anne Schuth, Katja Hofmann, Pavel Serdyukov, and Maarten de Rijke. 2013a. Evaluating aggregated search using interleaving. In *Proceedings of CIKM*. ACM, New York, NY, 669–678.

Aleksandr Chuklin, Pavel Serdyukov, and Maarten de Rijke. 2013b. Click model-based information retrieval metrics. In *Proceedings of SIGIR*. ACM, New York, NY, 493–502.

Aleksandr Chuklin, Pavel Serdyukov, and Maarten de Rijke. 2013c. Using intent information to model user behavior in diversified search. In *Advances in Information Retrieval*. Lecture Notes in Computer Science, Vol. 7814. Springer, 1–13.

Aleksandr Chuklin, Ke Zhou, Anne Schuth, Floor Sietsma, and Maarten de Rijke. 2014. Evaluating intuitiveness of vertical-aware click models. In *Proceedings of SIGIR*. ACM, New York, NY, 1075–1078.

Charles L. A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and diversity in information retrieval evaluation. In *Proceedings of SIGIR*. ACM, New York, NY, 659–666.

Cyril W. Cleverdon, Jack Mills, and Michael Keen. 1996. *Factors Determining the Performance of Indexing Systems*. Technical Report. ASLIB Cranfield project.

Thomas Demeester, Dolf Trieschnigg, Dong Nguen, and Djoerd Hiemstra. 2013. Overview of the TREC 2013 Federated Web Search track. In *Proceedings of TREC*.

Susan Dumais, Edward Cutrell, and Hao Chen. 2001. Optimizing search by showing results in context. In *Proceedings of CHI*. ACM, New York, NY, 277–284.

Jing He, Chengxiang Zhai, and Xiaoming Li. 2009. Evaluation of methods for relative comparison of retrieval systems based on clickthroughs. In *Proceedings of CIKM*. ACM, New York, NY, 2029–2032.

Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. 2011. A probabilistic method for inferring preferences from clicks. In *Proceedings of CIKM*. ACM, New York, NY, 249–258.

Katja Hofmann, Shimon Whiteson, and Maarten Rijke. 2012. Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval. *Information Retrieval* 16, 1, 63–90.

Katja Hofmann, Anne Schuth, Shimon Whiteson, and Maarten de Rijke. 2013a. Reusing historical interaction data for faster online learning to rank for IR. In *Proceedings of WSDM*. ACM, New York, NY, 183–192.

Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. 2013b. Fidelity, soundness, and efficiency of interleaved comparison methods. *ACM Transactions on Information Systems* 31, 3, Article No. 18.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems* 20, 4, 422–446.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of KDD*. ACM, New York, NY, 133–142.

Thorsten Joachims. 2003. Evaluating retrieval performance using clickthrough data. In *Text Mining*, J. Franke, G. Nakhaeizadeh, and I. Renz (Eds.). Physica/Springer-Verlag, 79–96.

Dong Nguyen, Thomas Demeester, Dolf Trieschnigg, and Djoerd Hiemstra. 2012. Federated search in the wild: The combined power of over a hundred search engines. In *Proceedings of CIKM*. ACM, New York, NY, 1874–1878.

Ashok Kumar Ponnuswami, Kumaresh Pattabiraman, Qiang Wu, Ran Gilad-Bachrach, and Tapas Kanungo. 2011. On composition of a federated Web search result page: Using online users to provide pairwise preference for heterogeneous verticals. In *Proceedings of WSDM*. ACM, New York, NY, 715–724.

Filip Radlinski and Nick Craswell. 2010. Comparing the sensitivity of information retrieval metrics. In *Proceedings of SIGIR*. ACM, New York, NY, 667–674.

Filip Radlinski and Nick Craswell. 2013. Optimized interleaving for online retrieval evaluation. In *Proceedings of WSDM*. ACM, New York, NY, 245–254.

Filip Radlinski, Madhu Kurup, and Thorsten Joachims. 2008. How does clickthrough data reflect retrieval quality? In *Proceedings of CIKM*. ACM, New York, NY, 43–52.

Anne Schuth, Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. 2013. Lerot: An online learning to rank framework. In *Proceedings of LivingLab*. ACM, New York, NY, 23–26.

Anne Schuth, Floor Sietsma, Shimon Whiteson, Damien Lefortier, and Maarten de Rijke. 2014. Multileaved comparisons for fast online evaluation. In *Proceedings of CIKM*. ACM, New York, NY, 71–80.

Jangwon Seo, W. Bruce Croft, Kwang Hyun Kim, and Joon Ho Lee. 2011. Smoothing click counts for aggregated vertical search. In *Advances in Information Retrieval*. Lecture Notes in Computer Science, Vol. 6611. Springer, 387–398.

Andrey Styskin. 2013. Aggregate and conquer: Finding the way in the diverse world of user intents. In *Proceedings of ECIR*.

Shanu Sushmita, Hideo Joho, Mounia Lalmas, and Robert Villa. 2010. Factors affecting click-through behavior in aggregated search interfaces. In *Proceedings of CIKM*. ACM, New York, NY, 519–528.

Yisong Yue and Thorsten Joachims. 2009. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of ICML*. ACM, New York, NY, 1201–1208.

Ke Zhou, Ronan Cummins, Mounia Lalmas, and Joemon M. Jose. 2012a. Evaluating aggregated search pages. In *Proceedings of SIGIR*. ACM, New York, NY,115–124.

Ke Zhou, Ronan Cummins, Mounia Lalmas, and Joemon M. Jose. 2012b. Evaluating reward and risk for vertical selection. In *Proceedings of CIKM*. ACM, New York, NY, 2631–2634.

Ke Zhou, Ronan Cummins, Mounia Lalmas, and Joemon M. Jose. 2013a. Which vertical search engines are relevant? In *WWW*, 1557–1568, ACM.

Ke Zhou, Mounia Lalmas, Tetsuya Sakai, Ronan Cummins, and Joemon M. Jose. 2013b. On the reliability and intuitiveness of aggregated search metrics. In *CIKM*, 689–698, ACM.

Ke Zhou, Thomas Demeester, Dong Nguyen, Djoerd Hiemstra, and Dolf Trieschnigg. 2014. Aligning vertical collection relevance with user intent. In *CIKM*, ACM.

Received June 2014; accepted September 2014