Optimizing Two-Dimensional Search Results Presentation

Flavio Chierichetti*
Dept. of Computer Science
Cornell University
Ithaca, NY 14853.
flavio@cs.cornell.edu

Ravi Kumar
Yahoo! Research
701 First Avenue
Sunnyvale, CA 94089.
ravikumar@yahoo-inc.com

Prabhakar Raghavan
Yahoo! Research
701 First Avenue
Sunnyvale, CA 94089.
pragh@yahoo-inc.com

ABSTRACT

Classic search engine results are presented as an ordered list of documents and the problem of presentation trivially reduces to ordering documents by their scores. This is because users scan a list presentation from top to bottom. This leads to natural list optimization measures such as the discounted cumulative gain (DCG) and the rank-biased precision (RBP).

Increasingly, search engines are using two-dimensional results presentations; image and shopping search results are long-standing examples. The simplistic heuristic used in practice is to place images by row-major order in the matrix presentation. However, a variety of evidence suggests that users' scan of pages is not in this matrix order. In this paper we (1) view users' scan of a results page as a Markov chain, which yields DCG and RBP as special cases for linear lists; (2) formulate, study, and develop solutions for the problem of inferring the Markov chain from click logs; (3) from these inferred Markov chains, empirically validate folklore phenomena (e.g., the "golden triangle" of user scans in two dimensions); and (4) develop and experimentally compare algorithms for optimizing user utility in matrix presentations. The theory and algorithms extend naturally beyond matrix presentations.

Categories and Subject Descriptors. H.3.m [**Information Storage and Retrieval**]: Miscellaneous

General Terms. Algorithms, Experimentation, Theory

Keywords. Page layout, Image search, Markov chain, User scan model

1. INTRODUCTION

Classically, search engine results are presented as an ordered list of documents; the problem of presentation trivially reduces to ordering documents by their scores. This is because users overwhelmingly scan a list presentation from top to bottom. This leads to natural list quality metrics such as the discounted cumulative gain (DCG) [13] and the rank-biased precision (RBP) [18], which is a geometrically weighted sum of the scores of the top 10 results. Increasingly, search engines are beginning to use two-dimensional results presentations; indeed, image and shopping search results are long-standing examples where the objects (images, or products) are presented to the user in a two-dimensional matrix. For image/product search, the commonest heuristic used is to place images (thumbnails) in a matrix presentation with the highest scoring object at the top left, then proceeding by decreasing score in row-major order on the matrix. However, a variety of evidence suggests that users' scan of pages is not in this row-major order. Rather, their eyes tend to traverse the page in a triangular trajectory, with some randomness [20, 19, 9]; see Figure 1. Such "non-linear" eye traversals are common even in page layouts other than the rectangular matrix [8, 3]. More generally, we wish to address settings such as Google's Universal search, Microsoft's Bing and Yahoo's direct displays, where results pages have a two-dimensional placement of objects including documents, photos, maps, fares — not necessarily in a grid of slots.

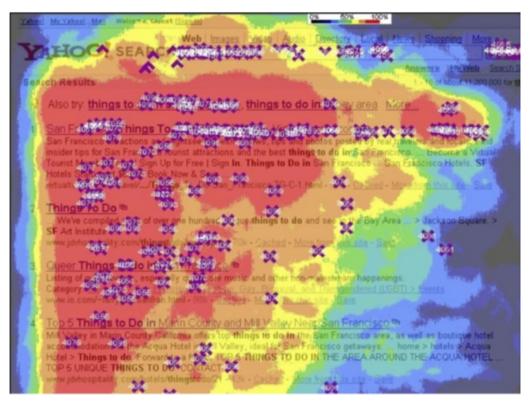


Figure 1: Golden triangle on a SERP.

The model. We view user eye-tracks as a Markov chain M on N states [15], with a state for each slot on the results page where we can place an image/product (thus the graph underlying M — henceforth denoted G_M — is a grid graph). The user's scan follows M, stopping with some probability at each step and occasionally clicking on an object and accumulating its utility; the transition probabilities of M govern

*Supported in part by NSF grant CCF-0910940. Part of this work was done while the author was at Yahoo! Research and at the Sapienza University of Rome.

WSDM'11, February 9–12, 2011, Hong Kong, China.

these various events. Given a Markov chain and a set of objects to be placed at its states, each object having a utility (its score for the query), the *placement problem* seeks to find an assignment of objects to states that maximizes the expected total utility of the user. For the case when the results are presented as a linear list, this utility can be shown to reduce to DCG or RBP for appropriate choices of transition probabilities; thus our model is a natural generalization of current metrics for linear lists. Further, this Markovian formulation extends to any two-dimensional arrangement of slots on a results page — not just the grid. Because the results in this paper focus on the grid, we make no further mention of the generalization; however our model, the optimization formulation, and the metrics we develop (generalizing precision and recall) are completely general for two-dimensional results presentation à la Google/Bing/Yahoo, and not restricted to the matrix presentation of image results.

Our contributions. In this paper we focus on the concrete application of placing images/products on a grid layout. Thus G_M is a grid, the transition probabilities come from actual user traces, the utilities are the scores of objects, and the algorithms must be relatively simple and fast. In Section 6 we compare several natural heuristics for image placement on data from web image search. The algorithms considered are naturally generalized to the case when G_M is not a grid, thus catering to general two-dimensional tilings in the results page; our experiments however are only for the grid. Along the way, we give new definitions of relevance metrics that naturally generalize precision and recall from the list setting to the general placement problem. We believe this generalization is of interest in its own right for other results presentation settings.

En route to these experiments we solve a technically challenging problem that may be of independent interest. In order to infer M from large-scale click data, we face the following: we are given a sequence of user clicks on grid points (corresponding to the clicks on an image search page). These clicks are seldom contiguous; thus, we do not know exactly how the user's eyes passed over the grid from one click to the next. Consequently, we must infer M from such click sequences. In Section 4 we formulate this inference problem, show that it is NP-hard even in the grid, and give heuristic solutions to it. We validate these solutions on large-scale click logs and show that our solutions converge rapidly to the underlying set of transition probabilities. From this "chain inference", we note several interesting phenomena in the inferred Markov chains. First, there is a conspicuous *golden triangle* in user eye movements, validating at large scale prior (smaller scale) eye-tracking studies [8, 3]: the transition probabilities are concentrated on a triangle anchored at one of the corners of the grid. Second, there is a shadow of a *silver triangle* typically rooted at the opposite corner.

Critique of our model. Our model has some shortcomings. While it can be enhanced to address some of these concerns, the resulting model becomes complex and even more computationally challenging; we note though that some of these shortcomings have existed even in one-dimensional list presentations.

(i) Markovian eye tracking: Our assumption of Markovian eye tracking with transitions independent of the actual objects being placed is not strictly correct. For instance, after seeing an image, the user is arguably likely to proceed to another image that appears visually different in the thumbnails. This suggests that a good placement algorithm should try and place diverse images in parts of the grid where the user is likely to begin scanning. This diversity problem is already an issue with results presented in linear lists, where we have promising approximate solutions [23]. While our model can be generalized to capture the formulation of [23], going beyond one dimension leads to optimization problems that are considerably harder. Thus, our model and results can be viewed as the two-dimensional analog of traditional list results, without twist of diversity added in.

(ii) Query independence: Our model posits a single Markov chain for user scans on all queries. Arguably the objects comprising the results of a query will influence the transition probabilities. As an extreme case, suppose that a query retrieves only a single object; then all transition probabilities out of the one slot containing this object are zero. Nevertheless, for the ensemble measurements we compute (utility, generalized precision/recall), as a first cut we assume that M is query-independent.

(iii) Browser geometry: In image and product search where the results are placed in a grid, the grid size depends on the browser shape at he time of the query. Thus a server that optimizes (say) placement on a 6×3 grid may do the wrong thing if the user sizes the browser so the displayed grid is 5×4. There are several ways of getting around this. Users tend by and large to leave their browsers at set full-screen sizes, so there is relatively little variation in grid size in practice. Moreover, as optimal placement algorithms are deployed, the image results can be sent to the browser with either a simple script to re-optimize based on the browser geometry, or alternatively the server could send the browser placements for the commonest grid sizes.

2. RELATED WORK

The related work falls into three main categories: the interplay between eye-tracking and SERPs, the body of work on Markov chain methods for user modeling on SERPs, and the algorithmic work on the placement problem.

Granka et al. [11] study the method of eye-tracking in online search and its use in augment standard IR techniques; see also the work by Cutrell and Guan [7]. Aula et al. [2] use eye-tracking to study how users evaluate SERPs. Kammerer and Gerjets [14] perform a similar study on results presented in a grid-like fashion. For a comprehensive account of eye tracking and online search, see the recent article by Lorigo et al. [17]. Rodden et al. [24] study the eye-mouse coordination patterns on SERPs, finding a relationship between eye and mouse movements; this line of research has been very active recently [12], including applying it to to infer relevance. To the best of our knowledge, there has been no work so far to model two-dimensional search results presentation in the context of eye-tracking.

Markov chain methods have been used in modeling visual search [22], inferring intent in visual interfaces [25], and in print media [21]. The work closest to ours it that of Wang, Gloy, and Li [27] who propose a partially observable Markov (POM) model to infer user actions (such as skipping certain results) that are missing from click logs. They propose a Viterbi-like algorithm and apply it to infer latent user actions on a SERP. Their segmental decoding method, however, crucially uses the one-dimensional aspect of the problem and hence does not seem applicable to our grid setting. In addition, they do not consider the placement problem. Bahl et al. [4] and Yu and Kobayashi [28] study the Markov chain inference problem in the setting of hidden Markov models with missing observations; however, in their work, the missing observations arise from rather complicated processes that do not apply here. Terwijn [26] shows that even a simpler problem very similar to our inference problem is hard, under cryptographic assumptions, on the the complete graph.

Chierichetti et al. [5] study the abstract computational complexity of the placement problem, showing that even if G_M is a directed acyclic graph (DAG) with only a single self-loop and each object has unit utility, the placement problem is inapproximable to within a factor better than exponential in N. They also show that the placement problem on general graphs can be approximated to within a factor $O(\log N)$ if the algorithm is allowed to leave some empty slots; these cases are mainly of theoretical interest. Thus the worst-case computational complexity of the placement problem is daunting. Aggarwal et al. [1] as well as Kempe and Mahdian [16] study special cases of linear lists, in the context of sponsored search advertisements. When G_M is a line with all inter-state transition probabilities being equal, they give an exact solution to the placement problem. Craswell et al. [6] give some empirical evidence (from click logs) in support of the linear model for list presentations. There appears to be no prior work on the placement problem beyond the one-dimensional list.

3. MODEL AND PROBLEMS

Let U be a universe of objects. Each *object* $u \in U$ has three attributes c_u, σ_u, and ν_u, where c_u is the *clicking* probability, σ_u is the *stopping* probability, and ν_u is the *utility*.

We assume a rectangular grid of height n and width m, where each grid point represents a *slot*; let $N = nm$. The top-left slot of the grid is labeled $(0, 0)$ and the bottom-right slot is labeled $(m-1, n-1)$. Each non-boundary slot (i, j) in the grid has directed edges to four of its neighboring slots and each edge has a probability associated with it. Let $u_{i,j}$, $d_{i,j}$, $\ell_{i,j}$, $r_{i,j}$ be the edge probabilities such that $u_{i,j} + d_{i,j} + \ell_{i,j} + r_{i,j}) \leq 1$; the remaining is the self-loop probability. We also assume that an edge probability is zero if the neighboring slot is outside the grid. Thus, the grid together with the edge probabilities forms a Markov chain M whose states are the slots. Let ξ be an *initial* probability distribution over the mn slots in the grid. Unless otherwise specified, we will assume that the distribution ξ is concentrated on the top-left corner (i.e., the slot $(0, 0)$).

Suppose each slot in the grid is filled with an object. Our model of *user behavior* is as follows: the user scans through the grid, where the starting slot of a scan is drawn from ξ. When looking at an object u in slot (i, j), the user will click on u with probability c_u, accumulating utility ν_u; she will stop scanning with probability σ_u. In case she decides not to stop, her scan moves either one slot up with probability $u_{i,j}$, or one down with probability $d_{i,j}$, or one left with probability $\ell_{i,j}$, or one right with probability $r_{i,j}$, or stays in the slot (i, j) with probability $1 - (u_{i,j} + d_{i,j} + \ell_{i,j} + r_{i,j})$.

Given this user behavior model, we define the placement problem.

Definition 1 (Placement problem). *Given a grid with the transition probabilities and a universe U of objects, assign an object from U to each slot in the grid so as to maximize the expected total accumulated utility for the user before stopping.*

Note that we can define the placement problem for any Markov chain that is not necessarily a grid; we focus on the grid in this paper. Also, note that in general $|U| \gg N$ and since each object has both a stopping probability and a utility, we cannot simply restrict our attention to the N objects in U of the highest utility.

In our empirical comparisons of placement algorithms, we need the underlying Markov chain M. We estimate the transition probabilities of M from observed trails of user clicks in image search logs. We are given a set of *click trails*, each consisting of a time stamp and the sequence of clicked grid slots, for example: ((0,4), (2,3), (3,3), (4,4)). From these trails, the goal is to infer the transition probabilities of M.

Definition 2 (Model estimation problem). *Given a set of click trails on a grid, estimate the transition and object click/stop probabilities to maximize the likelihood of generating the given set of click trails.*

This is an interesting variant of classic Hidden Markov model estimation and is non-trivial since the observed clicks may not be adjacent on the grid. We thus have to *interpolate* between successive clicks to derive the (probabilistic) trajectory of the scan. The difficulty of course is that we do not have the transition probabilities of M to begin with, so cannot compute a probability distribution over trajectories between successive clicks.

Thus, our scenario consists of solving two problems: the model estimation problem and the placement problem. We address the former in Section 4 and the latter in Section 6.

4. MODEL ESTIMATION PROBLEM

In this section we consider the model estimation problem. We begin by showing in Section 4.1 that even a weaker version of this problem is computationally hard. We then describe three heuristic methods for the estimation of the model parameters in Section 4.2. We provide an experimental evaluation of the estimation methods in Section 5.

Let $\mathrm{nyc}((i,j),(i',j')) = |i-i'|+|j-j'|$ denote the *Manhattan* distance between grid slots (i, j) and (i', j'). A directed path π is said to be *monotone* if for any two consecutive slots (i_1, j_1) and (i_2, j_2) in π, we have

$$\mathrm{nyc}((i_1, j_1), (i', j')) = \mathrm{nyc}((i_2, j_2), (i', j')) + 1.$$

The length of a monotone directed path π from slot (i, j) to slot (i', j') is $|\pi| = \mathrm{nyc}((i, j), (i', j'))$. It is easy to see that the total number of monotone paths is

$$\binom{|i'-i| + |j'-j|}{|i'-i|}.$$

4.1 Hardness

We first show that the model estimation problem on grids is NP-hard. The result is somewhat surprising (and disappointing) because even though our setting is more specialized than the general Hidden Markov model setting, the regularity of the grid structure does not seem to offer any computational relief. To show the NP-hardness, we actually work with a simpler version of the problem where we fix the stopping and the clicking probability of each object to be the same; let $p_s, p_c \in (0,1)$ be these values.

THEOREM 3. *The model estimation problem is NP-hard even if the underlying graph of the Markov chain M is an $n \times m$ grid.*

PROOF. We reduce from the *k-pairwise node disjoint shortest paths* problem on the grid, which was shown to be NP-hard [10]. In this problem, we are given an $n \times m$ grid, and t pairs of (non-necessarily distinct) grid nodes,

$$C_1 = \big((i_1, j_1), (i'_1, j'_1)\big), \ldots, C_p = \big((i_t, j_t), (i'_t, j'_t)\big).$$

The question is: are there t shortest disjoint paths $\pi_1, \ldots, \pi_t$ that connect the respective pairs? I.e., can one find a path π_k for each pair $C_k = ((i_k, j_k), (i'_k, j'_k))$ such that $|\pi_k| = \text{nyc}((i_k, j_k), (i'_k, j'_k))$ and no two paths share a node?

First observe that if there are two different pairs with non-empty intersection, then the problem has a trivial negative answer. We therefore assume that no node is shared by two or more pairs.

We build an instance of the model estimation problem given an instance of the k-pairwise node disjoint shortest paths problem. The grid will always be filled with the same object at each slot for each of the traces. For each pair $C_i = ((i,j),(i',j'))$ we construct a trail consisting of at most three clicks. Specifically,

(1) if $i = i'$ and $j = j'$, then we construct a trail consisting of two consecutive clicks in the slot (i,j);

(2) if $i \neq i'$ or $j \neq j'$, then we construct a trail consisting of the first click in slot (i,j) followed by two clicks in slot (i', j').

Suppose that there exist disjoint shortest paths $\pi_1, \ldots, \pi_t$ for the original instance. We now create a solution to the model estimation problem.

Let the initial distribution ξ be chosen to start in the first slot of C_k, for $k = 1, \ldots, t$, with probability $1/t$. Now, we consider C_k and $\pi_k = (z_{k,0}, \ldots, z_{k,|\pi_k|})$, where $z_{k,\ell}$ is the ℓth grid node in the path π_k. We drop the subscript k whenever it is obvious from the context. For $C = C_k$ and $\pi = \pi_k = (z_0, \ldots, z_{|\pi|})$,

(a) we assign probability 1 to the self-loop at slot $z_{|\pi|}$;

(b) if $|\pi| \geq 1$, then we assign probability

$$1 - s^* = \min\left(1, \frac{1}{(1-p_c)(1-p_s)} - 1\right)$$

to the edge going from slot z_0 to slot z_1, and we assign

$$s^* = \max\left(0, 2 - \frac{1}{(1-p_c)(1-p_s)}\right)$$

to the self-loop at the slot z_0;

(c) if $|\pi| \geq 2$, then we assign probability 1 to the edge going from slot z_ℓ to slot $z_{\ell+1}$, for each $\ell \in \{1, \ldots, |\pi|-1\}$.

The probability that a walk starting in slot $z_{|\pi|}$ produces a trail consisting of two copies of $z_{|\pi|}$ is exactly

$$\begin{aligned} A &= \sum_{i=2}^{\infty} \left(\binom{i}{2} p_c^2 (1-p_c)^{i-2} (1-p_s)^{i-1} p_s \right) \\ &= p_c^2 (1-p_s) p_s \sum_{i=2}^{\infty} \left(\binom{i}{2} ((1-p_c)(1-p_s))^{i-2} \right) \\ &= \frac{p_c^2 (1-p_s) p_s}{(p_c + p_s - p_c p_s)^3}, \end{aligned}$$

where the third equality follows from the identity $\sum_{i=r}^{\infty} \left(\binom{i}{r} a^{i-r}\right) = (1-a)^{-(r+1)}$, which holds for each $a \in [0,1)$.

Now consider the probability that a walk starting at the generic slot z, having self-loop probability s, produces a click in z without moving to other slots, and then moves to some other slot is

$$\begin{aligned} B(s) &= \sum_{i=1}^{\infty} \left(\binom{i}{1} p_c (1-p_c)^{i-1} (1-p_s)^i s^{i-1} (1-s) \right) \\ &= p_c (1-p_s)(1-s) \sum_{i=1}^{\infty} \left(\binom{i}{1} ((1-p_c)(1-p_s)s)^{i-1} \right) \\ &= \frac{p_c (1-p_s)(1-s)}{(1 - s(1-p_c)(1-p_s))^2}. \end{aligned}$$

Observe that the derivative of $B(s)$ is

$$\frac{d}{ds} B(s) = -p_c (1-p_s) \frac{(s-2)(1-p_c)(1-p_s) + 1}{(1 - s(1-p_c)(1-p_s))^3},$$

which is non-zero for $s < 2 - \frac{1}{(1-p_c)(1-p_s)} = S < 1$, zero if $s = S$, and negative if $S < s \leq 1$. Since s has to be chosen in $[0,1]$, one has that the $s = s^*$ maximizing $B(s)$ is given by $s^* = \max\left(0, 2 - \frac{1}{(1-p_c)(1-p_s)}\right)$.

Let $\pi = \pi_k$. If $|\pi_k| = 0$, then the probability that a walk starting in slot z_0 produces the trail $(z_0, z_0) = (z_{|\pi|}, z_{|\pi|})$ is exactly $P' = P'_k = A$. If $|\pi| \geq 1$, then the probability P''_k that a walk starting in slot z_0 produces the trail $(z_0, z_{|\pi|}, z_{|\pi|})$ is

$$P'' = P''_k = B(s^*)(1-p_c)^{|\pi|-1}(1-p_s)^{|\pi|-1} A.$$

Given any $C = C_k$, the probability that a walk starting in its first slot produces the trail constructed from C is then

$$P = P_k = A\,((1-p_c)(1-p_s))^{\max(|\pi|-1,0)}\, B(s^*)^{\min(|\pi|,1)}.$$

By the choice of ξ, we then have that the probability of observing the input set of traces is equal to $\mathcal{P} = t^{-t} \prod_{k=1}^{t} P_k$.

Now suppose that a solution to the model estimation problem exists with value at least $\mathcal{P}$. Observe that, for each trail (z_0, z_0), i.e., a trail for which $|\pi| = 0$, the probability of observing it, conditioned on the first visited slot to be $z_0 = z_{|\pi|}$, is at least P' only if the probability of the self-loop on $z_{|\pi|}$ is 1. (Indeed, if the probability of moving to another slot from z_1 was non-zero, then with at least that probability we would need to avoid the stopping event on the other slot before hoping to click on z_1. This would decrease the probability of the trail.)

On the other hand, the probability of observing the trail $(z_0, z_{|\pi|}, z_{|\pi|})$, for $\pi = \pi_k$, conditioned on the first visited slot to be z_0 is at most P_k with equality only if (i) the probability of the self-loop on $z_{|\pi|}$ is 1, (ii) the number of steps used to reach $z_{|\pi|}$, after having left z_0, is exactly $|\pi|$, and (iii) the self-loop probability of slot z_0 is exactly s^*. We established

(iii) by proving that s^* is the point at which $B(s)$ achieves the maximum. Now, (i) and (ii) follow directly from the the definition of A, and from the observation that a shorter path has a larger probability of being followed than a longer path.

Let $S_0 = S_{k,0} = \{z_0\}$, and $S_{i+1} = S_{k,i+1}$ be the set of slots that are reachable with non-zero probability by some slot in $S_i = S_{k,i}$. Then (ii) holds if and only if for each $i = 0, \ldots, |\pi|$ and for each $z \in S_i$, $\text{nyc}(z, z_{|\pi|}) = |\pi| - i$. (Indeed, for (ii) to hold, we have to make a step towards $z_{|\pi|}$ each time.)

Observe that, if a slot z is contained in two different $S = S_{k,i}$ and $S' = S_{k',i'}$ with $k \neq k'$ or $i \neq i'$, then either (i) or (ii) does not hold. Indeed, if $k = k'$, then we would have a contradiction: z cannot be both at distance i and a distance $i' \neq i$ from $z_{|\pi|}$. If $k \neq k'$, let $\pi' = \pi_{k'} = (z'_0, \ldots, z'_{|\pi'|})$. Since $z_{|\pi|} \neq z'_{|\pi'|}$ and since by (i) both $z_{|\pi|}$ and $z'_{|\pi'|}$ would have a self-loop with probability 1, it would be impossible to reach both $z_{|\pi|}$ and $z'_{|\pi'|}$ from z with probability 1.

Now, if no slot z is contained in two different $S_{k,i}$ and $S_{k',i'}$, then the probability of arriving at $z_{k,0}$ from $z_{k',0}$ is zero. Therefore, if p_k is the probability of visiting $z_{k,0}$, we have $\sum_{k=1}^{t} p_k \leq 1$. We will show that (iv) for the probability of the input traces to be at least $\mathcal{P}$, one must have $p_k = 1/t$ for $k = 1, \ldots, t$.

Observe that the probability of observing the input trail sequence is

$$\mathcal{Q} \leq \prod_{k=1}^{t} (p_k P_k) = \prod_{k=1}^{t} P_k \prod_{k=1}^{t} p_k.$$

By the arithmetic mean-geometric mean inequality we have $\prod_{k=1}^{t} p_k \leq \left(\frac{\sum_{k=1}^{t} p_k}{t}\right)^t$ where equality holds if and only if p_k's are all equal to t^{-1}. Therefore

$$\mathcal{Q} \leq \prod_{k=1}^{t} P_k \left(\frac{\sum_{k=1}^{t} p_k}{t}\right)^t,$$

with equality iff (i)-(iv) are all satisfied.

Now given that a solution to the model estimation problem exists with value at least $\mathcal{P}$, we know that (i)–(iv) hold. So take any pair $C = (z, z')$ with $z \neq z'$ and consider the S_i's, for $i = 1, \ldots, |\pi| + 1$. Create a path by choosing, z_1 from S_1, and given z_i chosen from S_i one of its neighbors in S_{i+1}, which will exist for each $i = 1, \ldots, |\pi|$ by (ii). As we have already argued the S_i's are pairwise disjoint so the paths will be pairwise disjoint, and will not hit any of the nodes of the length-zero paths so they will constitute a solution to the disjoint paths problem in the grid. □

4.2 Heuristic methods for estimation

Given the NP-hardness of the model estimation problem, we consider several heuristic methods to estimate the transition probabilities of the underlying grid.

The input to the model estimation problem is a set of trails, where a generic trail is a sequence of clicks of the form $z_1, \ldots, z_\ell$. Each z_i is a grid slot that was clicked by the user. Note that two consecutive slots z_i and z_{i+1} in this trail need not be adjacent slots in the grid; we assume that the user took some path to go from z_i to z_{i+1} according to the (unknown) Markov chain M. Short of eye-tracking on every one of millions of user click trails, it is not possible to infer the exact path the user took to go from z_i to z_{i+1}. Also, note that a trail need not end on a click; once again, this cannot be inferred from the data. We make the following two simplifying assumptions: (i) the user took a monotone path to go from z_i to z_{i+1} and (ii) the trail ends with the last click.

We now describe three heuristic methods for model estimation. Let $\mathcal{T}$ be the set of given trails.

1. *Naive method.* Fix a direction, say, down (↓). For each trail in $\mathcal{T}$, we count the number of times $n_{i,j,\downarrow}$ when a user clicked consecutively on slot (i,j) and slot $(i, j+1)$. Likewise, the other counts $n_{i,j,\uparrow}, n_{i,j,\rightarrow}, n_{i,j,\leftarrow}, n_{i,j,\circ}$ can be computed, where the last count is the number of consecutive clicks on the slot (i,j). The probability $d_{i,j}$ of downward transition is estimated to be the ratio of $n_{i,j,\downarrow}$ and the sum $n_{i,j,\uparrow} + n_{i,j,\downarrow} + n_{i,j,\leftarrow} + n_{i,j,\rightarrow} + n_{i,j,\circ}$. Likewise, the other probabilities $u_{i,j}, r_{i,j}, \ell_{i,j}$ can be estimated. Thus„ the naive method only takes into account consecutive clicks that are on neighboring slots.

2. *Uniform walk method.* In this method, we maintain nm quintuples of counters $(n_{i,j,\uparrow}, n_{i,j,\downarrow}, n_{i,j,\leftarrow}, n_{i,j,\rightarrow}, n_{i,j,\circ})$, one for each slot (i,j). Given these counter values, we can at any time estimate tentative values for all transition probabilities as in the Naive method. We go through the trails in $\mathcal{T}$, ordered by time. For the tth trail, if the user clicked on slots $(i_1, j_1), \ldots, (i_{n_t}, j_{n_t})$ in order, then for each $k = 1, \ldots, n_t - 1$, we consider the set $\mathcal{P}$ of monotone paths from (i_k, j_k) to (i_{k+1}, j_{k+1}). We take each monotone path in $\mathcal{P}$ and increase by $1/|\mathcal{P}|$ each of its counters (i.e., if some step of the path goes from slot (i,j) to $(i, j+1)$, then we increase the variable $n_{i,j,\downarrow}$). After all trails are thus processed, we output estimates for all transition probabilities. The method can be thought of as picking a monotone path uniformly at random from all Manhattan paths.

3. *MLE method.* In this iterative method, we once again maintain nm quintuple of counters $(n_{i,j,\uparrow}, n_{i,j,\downarrow}, n_{i,j,\leftarrow}, n_{i,j,\rightarrow}, n_{i,j,\circ})$, one for each slot (i,j). Given these counter values, we can at any time estimate tentative values for all transition probabilities as in the Naive method. Now, we go through the trails in $\mathcal{T}$, ordered by time. For the tth trail, if the user clicked on slots $(i_1, j_1), \ldots, (i_{n_t}, j_{n_t})$ in order then, for each $k = 1, \ldots, n_t - 1$, we consider the set $\mathcal{P}$ of monotone paths going from (i_k, j_k) to (i_{k+1}, j_{k+1}). We take the path in $\mathcal{P}$ that is most likely according to the current estimate of transition probabilities, where the probability of a path is the product of the current transition probabilities along it, and increase by 1 each of its counters, (i.e., if some step of the path goes from slot (i,j) to $(i+1, j)$, then we increase the variable $n_{i,j,\downarrow}$). After all trails are thus processed, we output new estimates for all transition probabilities and repeat the process.

We can actually show that the MLE method converges.

LEMMA 4. *Let $p(t) \in \{d_{i,j}(t), u_{i,j}(t), \ell_{i,j}(t), r_{i,j}(t)\}$ denote the probability estimate in the MLE method after processing the tth trail. Then,* $\lim_{t\to\infty} \frac{p(t)}{p(t+1)} = 1$.

PROOF. The variable $p(t)$ is the probability that a walk in slot (i,j) goes to slot (i', j'), with $\text{nyc}((i,j), (i', j')) \leq 1$. Suppose that the MLE method changes the probabilities of slot (i,j) finitely many times; then if t_0 is the index of the last trail that changes the probability of slot (i,j), then it holds that $\frac{p(t)}{p(t+1)} = 1$, for each $t \geq t_0$ and we are done. Otherwise, the sum of the counters $n_{i,j,\uparrow}(t), n_{i,j,\downarrow}(t), n_{i,j,\leftarrow}(t)$,

$n_{i,j,\rightarrow}(t)$, and $n_{i,j,\circ}(t)$ diverges. A trail can change this sum by at most a constant value (which is the sum of the Manhattan distances of consecutive clicked slots in the trails), therefore by choosing a large enough $t_0 = t_0(\epsilon)$, we can achieve $1 - \epsilon \leq p(t)/p(t+1) \leq 1 + \epsilon$, for each $\epsilon > 0$. $\square$

Let A_u be the number of times object u received the last click of a trail and let B_u denote the number of times that object u was clicked. We estimate the stopping probability σ_u by the ratio A_u/B_u.

5. MODEL ESTIMATION: EXPERIMENTS

In this section we outline the experimental evaluation of various model estimation methods. First, we describe the dataset used in the experiments, which consists of image click trails (Section 5.1). Next, we apply the various methods to infer the Markov chains; we present the results in Section 5.2. We then evaluate the methods by comparing how well each predicts observed click-through rates (Section 5.3). Finally, we study the robustness of the parameters and the validity of assumptions in our experiments (Section 5.4).

5.1 Data

The data consists of queries issued to the image corpus of Yahoo! search engine and all the user clicks for each of these queries. The data was based on a subset of US search query logs from Dec 1–10, 2009 and consists of the following fields: timestamp, bcookie, number of results shown (18, 20, or 21), slot clicked, page number (1, 2, or 3), and the identifier of the image that was clicked. There were more than 28.8M clicked images in the data. By aggregating the user bcookie, we obtain the click trail for each query and each user, i.e., the chronological sequence of one or more clicks on one or more pages on images made by a user for a query.

At the time of data collection, the results (i.e., images) for the Yahoo! image search query could be shown in a variety of grid configurations: 7×3, 6×3, 5×4, 4×5, and 3×7. Due to instrumentation issues, the configuration was not recorded in the logs. Thus, we have to choose the 6×3 configuration for our experiments since it was the only configuration that was unambiguous, given the number of results. Also, at the time of data collection, the first page of image results in Yahoo! had numerous search engine features such as search suggestions, direct display elements, images inserted from late-breaking news stories, and other sources that distort the user click behavior. For a cleaner interpretation of the results, we therefore focus on page 2 of the image search results for most of the paper.

With the above restrictions, there were about 1.34M click trails representing 402K queries. Recall that each click trail is of the form $(z_1, \ldots, z_\ell)$, where $\ell \geq 1$ and each z_i is slot on the 6×3 grid. And, recall that if z_i and z_{i+1} are non-neighboring slots on the grid, we assume that the user took a monotone path to go from z_i to z_{i+1}. Also, we assume that the last click at z_ℓ denotes the end of the trail.

5.2 Inferred Markov chains

Our first goal is to infer the Markov chain on the 6×3 grid (note that in our notation, this corresponds to 6 columns and 3 rows) using the model estimation methods. As mentioned, we assume that the user begins at the top-left slot on the grid; in Section 5.4 we examine this assumption in more detail. By this assumption, a trail $(z_1, \ldots, z_\ell)$ becomes $((0, 0), z_1, \ldots, z_\ell)$

The results of the Markov chain inferencing methods in Section 4.2 are shown in Figure 2. For ease of presentation, we omit the probabilities corresponding to the self-loops; these can be inferred from the other probabilities depicted and the stopping probability.

It is important to note that the self-loop probabilities of the random walk and the MLE methods are lower than that of the naive method, since we consider all slots in the trail as opposed to consecutive slots in the trail that are neighbors in the grid. Also, the MLE method converges very fast in practice: Figure 3 shows the variational distance between the inferred Markov chains for consecutive iterations of the MLE method. It seems clear that fewer than 10 iterations is sufficient for the MLE method to converge.

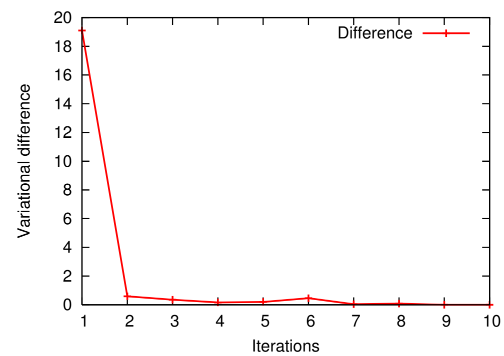


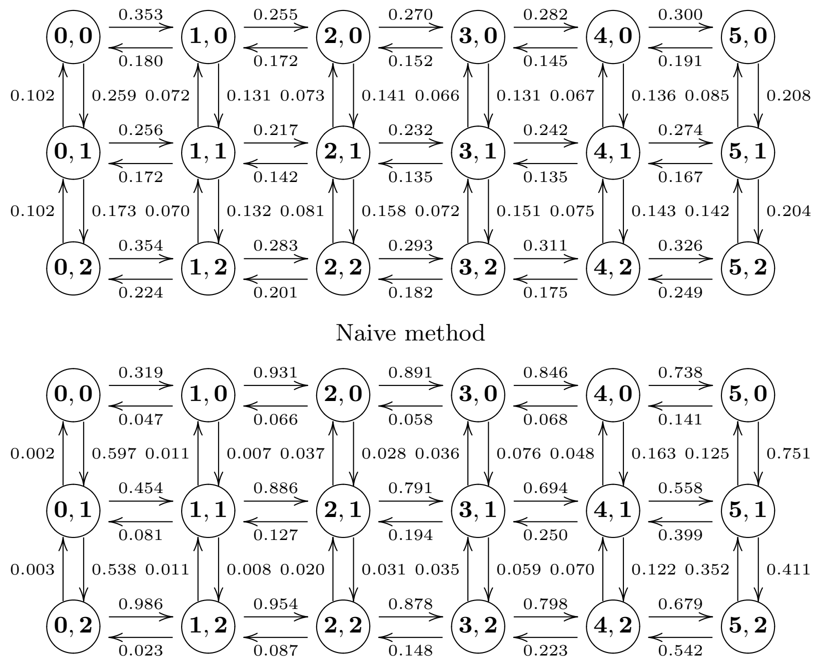
Figure 3: Convergence of the MLE method.

5.3 Evaluation of the methods

To evaluate the goodness of our two estimation methods, we perform the following experiment. First, we estimate from our data the probability p_s with which a trail stops at each step. This is the reciprocal of the average trail length, where the length of a trail $(z_0 = (0, 0), z_1, \ldots, z_\ell)$ is $\sum_{i=1}^{\ell} \text{nyc}(z_i, z_{i-1})$. For our data, we have $p_s = 0.20$.

Then, for the Markov chain obtained by each method, we compute its stationary distribution, where the random walk resets to ξ (in our case, jumps to (0, 0)) with probability p_s at each step. Intuitively, this corresponds to the ending of the current user's trail and the beginning of a new user according to ξ. To evaluate the performance of the two methods, we compute the variational distance between the stationary distribution of the inferred chain (modified with p_s) and the empirically obtained click fractions. The rationale: the stationary distribution of the Markov chain is a proxy for users' likelihoods of clicking at various grid points; thus if the stationary distribution of an estimated Markov chain is close to the observed click probabilities, it is a good user model. Table 1 shows the results. Clearly, the stationary distribution of the MLE and uniform walk methods is much closer to the empirical values than the naive method. Furthermore, the MLE method obtains about 8% less error than the uniform walk method. For the rest of the paper, we will assume the Markov chain is obtained using the MLE method.

We now examine the stationary distributions in more detail. To visualize better, we show in Figure 4 a heat map of



Naive method

MLE method

Figure 2: Inferred Markov chain; the self-loop probabilities are not shown. Note that many of the transition probabilities are very different in the two cases, even though the corresponding errors in Table 1 are not dramatically different.

Method	Error
Naive	0.447
Uniform walk	0.214
MLE	0.197

Table 1: Variational distance between the stationary distributions of the inferred chain and the empirical click fractions.

the empirical fraction of clicks at each slot on the grid and the stationary distribution for the inferred Markov chain using the naive method and the MLE method. Clearly, the golden triangle is prominent on the top-left corner in all three heat maps, reconfirming folklore. Less prominent yet noticeable is a *silver triangle* — a region of slightly higher click probabilities in the bottom-right corner. The silver triangle exists in the empirical data and the inferred Markov chain using the MLE method exhibits it as well. We believe this is caused by user's "last-ditch" attention, hoping to find something on the current page of results, before proceeding to the next page of results.

5.4 Robustness of choices

Effect of ξ. First, we discuss the effect of choosing ξ to be concentrated at the top-left slot (0, 0). We consider other obvious choices: bottom-left, top-right, bottom-right corners of the grid or the center of the grid. Notice that the choice of ξ impacts both the inference process of the Markov chain and the computation of the stationary probabilities (since the restart step uses ξ). Table 2 shows the variational distance between the stationary distribution using ξ concentrated in different slots and the empirical click fractions. Clearly we can see that the empirical click fractions are best matched by the top-left choice, justifying our assumption.

ξ	Error
top-left	0.197
top-right	0.381
bottom-left	0.289
bottom-right	0.381
center	0.347

Table 2: Variational error for different choices of the concentration of ξ.

Effect of choosing the results page. Next we study the difference in user behavior on pages 1, 2, and 3 of the image search results. To do this, we once again resort to the variational distance for all three methods. Table 3 shows the results. We see that pages 2 and 3 exhibit a great deal of similarity according to all the methods. Page 1, on the other hand, is markedly different, due to the artifacts on page 1 mentioned earlier (news images, etc). This justifies our choices of using page 2 as the basis of our experimental study in the next section. This is also confirmed by the difference in empirical click probability distributions, also shown in Table 3.

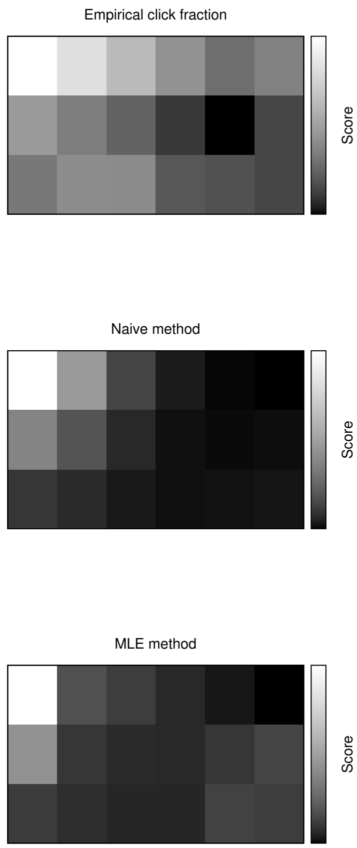


Figure 4: Heat map of the empirical fraction of clicks at each slot on the 6×3 grid and the stationary distribution for the inferred Markov chain obtained from the naive and the MLE methods, with $p_s = 0.20$.

6. PLACEMENT PROBLEM

In this section we develop and compare various heuristics for the placement problem. Recall that in the placement problem, we are given a grid Markov chain M and a universe U of objects and we wish to assign an object from U to each slot in M to maximize the expected accumulated utility. Our approach to this problem is to first obtain a static ordering of the nodes in M and then assign the objects in U in this order, after sorting the objects according to the better of a decreasing order of their utilities or an increasing order of their stopping probabilities.

Since our universe U of objects is large (every image in the index has a non-zero utility), we first introduce a "kernelization" trick that enables us to prune the number of objects considered for placement. An object u is said to be *preferable* to an object u' if $\sigma_u \leq \sigma_{u'}$ and $\nu_u \geq \nu_{u'}$; we write this as $u \geq u'$. If either of these inequalities is strict, we write $u > u'$.

LEMMA 5. *Suppose there exists $u, u_1, \ldots, u_N \in O$ such that $u_i \geq u$, for each $i = 1, \ldots, N$. Then any optimal placement for $U \setminus \{u\}$ is also optimal for U.*

Method	page 1 vs page 2	page 2 vs page 3
Naive	0.173	0.024
Uniform walk	0.129	0.019
MLE	0.277	0.024
Empirical	0.192	0.042

Table 3: Variational error of the stationary distribution of the Markov chain, across pages.

PROOF. We show that any placement that is optimal for U does not contain u. Suppose to the contrary there exists some $u_i \in \{u_1, \ldots, u_N\}$ that is not in the putative optimal placement (since the placement contains N objects one of which is u). Replace u with u_i. Then (i) each path in M has a traversal probability after the replacement no smaller than before (since $\sigma_{u_i} \leq \sigma_u$), and (ii) each such path has also utility that is no smaller than before (since $\nu_{u_i} \geq \nu_u$). Thus the new placement, containing u_i, is at least as good as the original one containing u. Thus we have an optimal placement for U that does not contain u. $\square$

Thanks to kernelization, we can remove all the objects $u \in U$ that have at least N preferable objects (recall that we cannot simply consider only the N objects of highest utility). This allows us to drastically reduce the number of objects. After this step, given a static ordering of the states of M, we assign the objects either by decreasing utility or by increasing stopping probability and select the object ordering that yields a better value.

We next describe our two new placement methods, which use the Markov chain inferred from one of the three methods outlined in Section 4.

1. EIGEN *placement.* This method computes the stationary distribution [15] of the Markov chain, with the random walk resetting to ξ (in our case, jumps to (0, 0)) with probability p_s at each step (this ensures ergodicity). The slots are then ordered by decreasing values of the stationary probabilities of this process and the objects (sorted by decreasing utility or increasing stopping probability) are assigned to the slots in this order. The intuition behind this method is that slots with large stationary probabilities should get objects of high utility.

2. HIT *placement.* This method computes the hitting time of the random walk starting according to ξ (in our case, from (0, 0)) and proceeds according the inferred Markov chain. Recall that the hitting time from i to j of a random walk is the expected time taken for the walk to go from i to j. It is given by the linear recurrence

$$h_i^j = 1 + \sum_{(i,k)} p_{ik} h_k^j, \quad \text{if } j \neq i,$$

and 0 otherwise. The slots are then ordered by increasing values of the hitting times and the objects (sorted by either decreasing utility or increasing stopping probability) are assigned to the slots in this order. The intuition behind this method is that slots likely to be visited first get objects of high utility.

Note that the performance of the above two methods can be compared against two natural baselines given by row-ordering (ROW) and column-ordering (COL) of the slots. In many scenarios, these baselines are the state-of-the-art.

Figure 5 shows the ordering of slots in the 6×3 obtained using EIGEN and HIT placement methods. The ordering produced by HIT seems intuitive and more reasonable than the ordering produced by EIGEN.

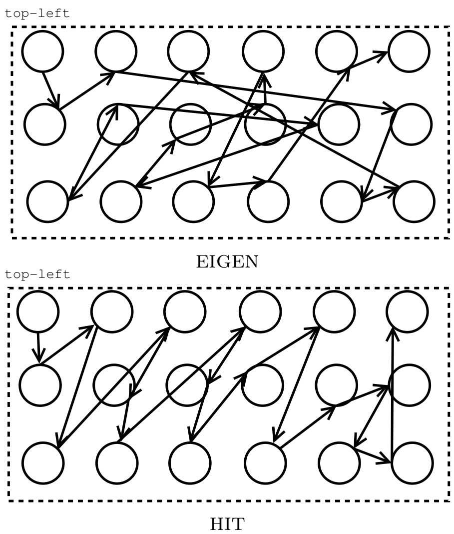


Figure 5: Slot ordering using EIGEN and HIT methods.

7. PLACEMENT: EXPERIMENTS

In this section, we analyze the performance of the placement methods and compare them with the ROW and COL baselines. First, we introduce a generalized notion of precision and recall that will be helpful in our Markovian setting (Section 7.1). We then discuss the datasets used (Section 7.2) and the performance of the methods (Section 7.3) on the datasets.

7.1 Generalized precision and recall

We have defined the quality (measured by total utility) of a complete placement. We now extend standard IR measures like precision and recall into our Markovian framework. Intuitively, *recall* in standard IR is a measure of the user's patience — how willing the user is to continue looking for relevant results. We capture this by introducing a probabilistic parameter in our Markovian model, allowing us to define generalized notions of recall and precision. Let r be the probability with which the user at each step continues to follow M, rather than stop immediately; we call r the *generalized recall (GR)*. Specifically at each step, the user with probability $1-r$ stops immediately at the current slot; with probability r, the user picks the next step from M (which might itself result in stopping). Thus at r close to 0 the user stops quickly, while at $r = 1$, the user simply follows M. For a given value of r, we can measure the utility from a run through the Markov chain; when divided by the number of objects clicked, we obtain the *generalized precision (GP)* for that run. Note that for simplicity, we allow the possibility of multiple clicks on the same object; for the relatively few object clicks in the trails in our data, this is not a material effect. Averaged over a large number of such runs, we obtain the generalized precision at r, which is analogous to the standard precision-recall curves. We show these for the four placement methods we have studied; the experiments were run as follows. Given M and a placement method $\mathcal{A}$, we draw the GP-GR curve for $\mathcal{A}$ as follows:

(1) Take a query and compute a placement of its results objects by $\mathcal{A}$.

(2) Run M on this placement for a given value of $r \in [0, 1]$.

(3) For a run, measure the total utility and divide by the number of objects that got clicked in that run; this is the GP for that run.

(4) Average over multiple runs and multiple queries, to get averaged GP values for $\mathcal{A}$, at each of given values of r.

7.2 Data

Our study is averaged over a query log consisting of the 10000 of the most popular queries to Yahoo! image search. For each query, we have the top 100 images, where each image has two attributes, namely, its utility and stopping probability. (We use the search engine's relevance score of the image for the query as a proxy for its utility, and compute the stopping probability as described earlier.) From these 100 candidates we select and place 18 images on a 6×3 grid.

7.3 Performance

We study the performance of the two placement methods EIGEN and HIT against the two baselines ROW and COL for the Markov chains inferred by the MLE method. Our measure of performance is the expected utility of placing the images according to the ordering, where the utility is computed according to our model *but* with object-specific stopping probabilities. The results are shown in Figure 6, where curves show the average utility when restricted to the top k queries. From the figure, we can see that the orderings

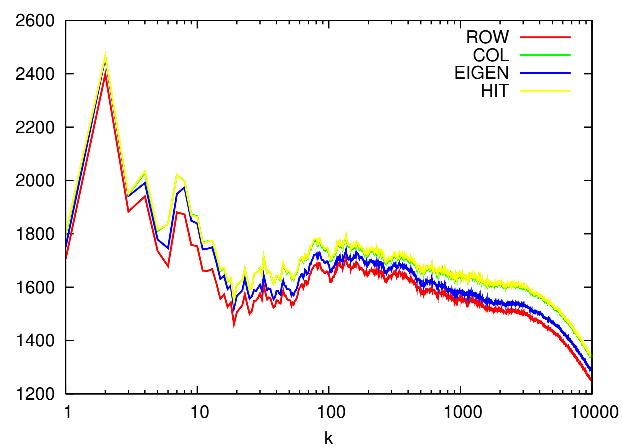


Figure 6: Instantaneous average utility of the placement methods for 10000 of the most popular queries.

given by COL and HIT produce better utilities than EIGEN and significantly better utilities than ROW.

In Figure 7, we observe the GP-GR curve for different object placement methods. As in a classical precision-recall plot, we notice how increasing the recall (x-axis) decreases the precision of the run. In particular we notice how the different curves (corresponding to different object placement's algorithms) look similar to one another. We observe that

HIT performs the best overall, followed by EIGEN and COL, which perform similar to each other.

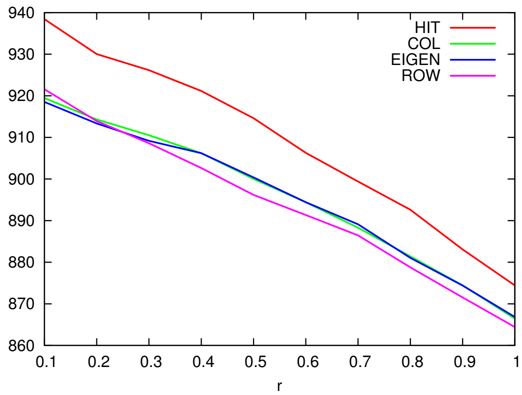


Figure 7: GP-GR curves of the placement methods.

8. CONCLUSIONS

We have formulated the presentation of results on a two-dimensional grid as an optimization problem, based on a view of user scans as following a Markov chain. The objective function for this optimization, the metrics of generalized precision and generalized recall, the Markov chain inference formulation and algorithms, and the placement algorithms are all completely general, beyond the grid. Thus the most significant future work would entail experimenting with non-grid two-dimensional placement. The added challenge here is that the layout of slots on the page and thus G_M may itself be a part of the design/optimization process. Thus, our framework and results are a first step in algorithmic two-dimensional results presentation and much work remains.

Acknowledgments

We thank Subhajit Sanyal, Malcolm Slaney, and Roelof van Zwoel for their help with the data.

9. REFERENCES

[1] G. Aggarwal, J. Feldman, S. Muthukrishnan, and M. Pal. Sponsored search auctions with Markovian users. In *Proc. 4th WINE*, pages 621–628, 2008.

[2] A. Aula, P. Majaranta, and K.-K. Raiha. Eye-tracking reveals the personal styles for search result evaluation. In *Proc. INTERACT*, pages 1058–1061, 2005.

[3] A. Aula and K. Rodden. Eye-tracking studies: More than meets the eye, 2009. `http://googleblog.blogspot.com/2009/02/eye-tracking-studies-more-than-meets.html`.

[4] L. R. Bahl, F. Jelinek, and R. L. Mercer. A maximum likelihood approach to continuous speech recognition. *PAMI*, 5:179–190, 1983.

[5] F. Chierichetti, R. Kumar, and P. Raghavan. A Markovian optimization problem with applications to page layout, 2010. Manuscript.

[6] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *Proc. 1st WSDM*, pages 87–94, 2008.

[7] E. Cutrell and Z. Guan. What are you looking for? An eye-tracking study of information usage in web search. In *Proc. CHI*, pages 407–416, 2007.

[8] A. Duchowski. A breadth-first survey of eye tracking applications. *Behavior Research Methods, Instruments, and Computers*, 34(4), 2002.

[9] Enquiro. Enquiro eye tracking report I: Google, 2005. `www.enquiro.com/research.asp`.

[10] T. F. Gonzalez and D. Serena. Complexity of pairwise shortest path routing in the grid. *TCS*, 326(1-3):155–185, 2004.

[11] L. Granka, M. Feusner, and L. Lorigo. Eye monitoring in online search. In R. Hammoud, editor, *Passive Eye Monitoring*, pages 283–304. Springer, 2008.

[12] Q. Guo and E. Agichtein. Towards predicting web searcher gaze position from mouse movements. In *Proc. CHI*, pages 3601–3606, 2010.

[13] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *TOIS*, 20(4):422–446, 2002.

[14] Y. Kammerer and P. Gerjets. How the interface design influences users' spontaneous trustworthiness evaluations of web search results: comparing a list and a grid interface. In *Proc. ETRA*, pages 299–306, 2010.

[15] J. G. Kemeny and J. L. Snell. *Finite Markov Chains*. Springer, 1976.

[16] D. Kempe and M. Mahdian. A cascade model for externalities in sponsored search. In *Proc. 4th WINE*, pages 585–596, 2008.

[17] L. Lorigo, M. Haridasan, H. Brynjarsdottir, L. Xia, T. Joachims, G. Gay, L. Granka, F. Pellacini, and B. Pan. Eye tracking and online search: Lessons learned and challenges ahead. *JASIST*, 59:1041–1052, 2008.

[18] A. Moffat and J. Zobel. Rank-biased precision for measurement of retrieval effetiveness. *TOIS*, 27(1), 2008.

[19] J. Nielsen. F-shaped pattern for reading web content, 2006. Alterbox.

[20] S. Outing and L. Ruel. The best of eyetrack III: What we saw when we looked through their eyes. `http://www.poynterextra.org/eyetrack2004/main.htm`.

[21] R. Pieters, E. Rosbergen, and M. Wedel. Visual attention to repeated print advertising: A test of scanpath theory. *J. Marketing Research*, 36:424–438, 1999.

[22] V. Ponsoda, D. Scott, and J. M. Findlay. A probability vector and transition matrix analysis of eye movements during visual search. *Acta Psychologica*, 88:167–185, 1995.

[23] F. Radlinski and S. Dumais. Improving personalized web search using result diversification. In *Proc. 29th SIGIR*, pages 691–692, 2006.

[24] K. Rodden, X. Fu, A. Aula, and I. Spiro. Eye-mouse coordination patterns on web search results pages. In *Proc. CHI*, pages 2997–3002, 2008.

[25] D. Salvucci. Inferring intent in eye-based interfaces: Tracing eye movements with process models. In *Proc. CHI*, pages 254–261, 1999.

[26] S. A. Terwijn. On the learnability of hidden Markov models. In *Proc. ICGI*, pages 261–268, 2002.

[27] K. Wang, N. Gloy, and X. Li. Inferring search behaviors using partially observable Markov (POM) model. In *Proc. 3rd WSDM*, pages 211–220, 2010.

[28] S.-Z. Yu and H. Kobayashi. A hidden semi-Markov model with missing data and multiple observation sequences for mobility tracking. *Signal Processing*, 83, 2003.