Cluster Based Personalized Search*

Hyun Chul Lee
University of Toronto
Toronto,ON, Canada
leehyun@cs.toronto.edu

Allan Borodin
University of Toronto
Toronto,ON, Canada
bor@cs.toronto.edu

ABSTRACT

We study personalized web ranking algorithms based on the existence of document clusterings. Motivated by the topic sensitive page ranking of Haveliwala [19], we develop and implement an efficient "local-cluster" algorithm by extending the web search algorithm of Achlioptas et al. [10]. We propose some formal criteria for evaluating such personalized ranking algorithms and provide some preliminary experiments in support of our analysis.

1. INTRODUCTION

Due to the size of the current Web and the diversity of user groups using it, the current algorithmic search engines are not completely ideal for dealing with queries generated by a large number of users with different interests and preferences. For instance, it is possible that some users might input the query "Star Wars" with their main topic of interest being "movie" and therefore expecting pages about the popular movie as results of their query. On the other hand, others might input the query "Star Wars" with their main topic of interest being "politics" and therefore expecting pages about proposals for deployment of a missle defense system. Of course, in this example, the user could easily disambiguate the query by adding say "movie" or "missle" to the query terms. But a more curious user might want to understand the process by which fictional movie scripts have an impact on current political debates. Therefore, to both expedite simple searches as well as to try to accommodate more complex searches, *web search personalization* has recently gained significant attention for handling queries produced by diverse users with very different search intentions. The goal of web search personalization is to allow the user to perform and expedite web search according to ones personal search preference or context.

There is no general consensus on exactly what web search personalization means, and moreover, there has been no general criteria for evaluating personalized search algorithms. The goal of this paper is to propose a framework, which is general enough to cover many real application scenarios, and yet is amenable to analysis with respect to correctness in the spirit of Achlioptas et al [10] and with respect to stability properties in the spirit of Ng et al. [25] and Lee and Borodin [23] (see also [12, 16]). We achieve this goal by assuming that the targeted web service has an underlying cluster structure. Given a set of clusters over the intended documents in which we want to perform personalized search, our framework assumes that a user's preference is represented as a preference vector over these clusters. A user's preference over clusters can be collected either on-line or off-line using various techniques [26, 14, 28, 18]. We do not address how to collect the user's search preference but we simply assume that the user's search preference (possibly with respect to various search features) is already available and can be translated into his/her search preference(s) over given cluster structures of targeted documents. We define a class of personalized search algorithms called "local-cluster" algorithms that compute each page's ranking with respect to each cluster containing the page rather than with respect to every cluster. In particular, we propose a specific local-cluster algorithm by extending the approach taken by Achlioptas et al. [10]. Our proposed local-cluster algorithm considers linkage structure and content generation of cluster structures to produce a ranking of the underlying clusters with respect to a user's given search query and preference. The rank of each document is then obtained through the relation of the given document with respect to its relevant clusters and the respective preference of these clusters. The ranking of documents obtained using our model can be combined with other IR or link analysis techniques used for traditional web search. Therefore, our algorithm is particularly suitable for equipping already existing web services with a personalized search capability without affecting their original ranking system.

Our framework allows us to propose a set of evaluation criteria for personalized search algorithms. We prove that the Topic-Sensitive PageRank algorithm [19], which is probably the best known personalized search algorithm in the literature, does not satisfy some properties that we propose for a "good" personalized search algorithm. In contrast, we show that our local-cluster algorithm satisfies the suggested properties.

Our main contributions are the following.

- We propose a new personalized search algorithm which shows the practicability of the web search model and

*Research supported by MITACS

algorithm proposed by Achlioptas et al [10].

- We propose some formal criteria for evaluating personalized search algorithms and then compare our proposed algorithm and the Topic-Sensitive PageRank algorithm based on such formal criteria.
- We experimentally evaluate the performance of our proposed algorithm against that of the Topic-Sensitive PageRank algorithm.

2. MOTIVATION

We believe that our assumption that the web service to be personalized admits cluster structures is well justified. Sometimes, cluster structures are not explicitly constructed but the web service classifies certain data items that share the same property and processes them in a special way so that such a grouping of data items can be viewed as representing cluster structures. In what follows, we discuss some existing examples having either explicit or implicit cluster structures, usually associated with a certain type of web data:

- *Human generated web directories*: In those web sites like Yahoo [7] and Open Directory Project[5], web pages are classified into human edited categories (possibly machine generated as well). Therefore, in order to personalize such systems, we can simply take any subset of category levels of the given taxonomy as our clusters, and our framework is able to model the personalization scenario in which such web directories are to be equipped with the personalized search capability upon the corresponding sub-taxonomy.
- *Articles classified according to topics*: Sites like To-pix.net [6], Google News [4], About.com [1] classify automatically collected or manually generated articles into different topics using various criteria. Normally, details about criteria used for such classification are not revealed. Once again, we can simply take each topic (e.g. sports) to represent a cluster and our framework properly models the personalization scenario of these sites.
- *Local search engines*: Sites like Yahoo Local [8], Google Local [3] and Citysearch [2] classify reviews, web pages and business information of local businesses into different categories and locations (e.g., city level). Therefore, in this particular case, a cluster would correspond to a set of data items or web pages related to the specific geographic location (e.g. web pages about restaurants in Houston, TX).

We note that the same corpus can admit several cluster structures using different features. For instance, web documents can be clustered according to features such as topic [7, 5, 6, 4, 1], whether commercial or educational oriented [9], domain type, language, etc. Our framework allows incorporating various search features into web search personalization as it works at the abstract level of clustering structures.

3. PRELIMINARY

Let G_N (or simply G) be a web page collection (with content and hyperlinks) of node size N, and let q denote a query string represented as a term-vector. Let $\mathcal{C}(G) = \{C_1, \ldots, C_m\}$ be a clustering (not necessarily a partition) for G (i.e. each $x \in G$ is in $C_{i_1} \cap \ldots \cap C_{i_r}$ for some i_1, …, i_r). If it is clear in the context, we will simply denote $\mathcal{C}(G)$ as $\mathcal{C}$. We define a **cluster-sensitive page ranking algorithm** μ as a function with values in $[0, 1]$ where $\mu(C_j, x, q)$ will denote the ranking value of page x relative to [1] cluster C_j with respect to query q. We define a user's preference as a $[0, 1]$ valued function P where $P(C_j, q)$ denotes the preference of the user for cluster C_j (with respect to query q). We call $(G, \mathcal{C}, \mu, P, q)$ an *instance of personalized search*; that is, a personalized search scenario where there exist a user having a search preference function P over a clustering $\mathcal{C}(G)$, a query q, and a cluster-sensitive page ranking function μ. Note that either μ or P can be query-independent.

DEFINITION 1. *Let $(G, \mathcal{C}, \mu, P, q)$ be an instance of personalized search. A **personalized search ranking** PSR is a function that maps G_N to an N-dimensional vector by composing μ and P through a function F; that is,*

$$PSR(x) = F(\mu(C_1, x, q), \ldots, \mu(C_m, x, q), P(C_1, q), \ldots, P(C_m, q))$$

For instance, F might be defined as a weighted sum of μ and P values. We will interchangeably use *personalized search ranking* and *personalized search function*.

4. PREVIOUS ALGORITHMS

4.1 Modifying the PageRank algorithm

Due to the popularity of the PageRank algorithm [13], the first generation of personalized web search algorithms are based on the original PageRank algorithm by manipulating the *teleportation factor* of the PageRank algorithm. Recall that within the random walk model of the original PageRank algorithm, the user jumps uniformly at random to a node in the collection. More precisely, let U be a $n \times n$ rank-one row-stochastic matrix such that $U = ev^T$, where e is the n-vector whose elements are all $e_i = 1$ and v is an n-vector whose elements are all non-negative and sum to 1. Transition probability of the original PageRank is given by $\epsilon \cdot U + (1 - \epsilon) \cdot A^t_{row}$ where matrix A^t_{row} represents the transition probability from i to j. Since in terms of the random walk, the destination of the random surfer that performs a random jump is chosen according to the probability distribution given in v, the U is referred to as *teleportation*. Moreover, the v is referred as the *personalization* vector as it controls which pages should be preferred than others. The first generation of personalized web search algorithms introduce some bias, reflecting the user's search preference, over certain kinds of pages by adding artificial transitions with non-uniform probabilities on the teleportation factor (i.e. controlling v). Among these, we have Topic-Sensitive PageRank [19], Modular PageRank [20] and BlockRank [21]. In this paper, we restrict our analysis to the Topic-Sensitive PageRank algorithm leaving the study of other PageRank based personalized algorithms for future research.

4.1.1 Topic-Sensitive PageRank

One of the first proposed personalized search ranking algorithms is ***Topic-Sensitive*** PageRank[19]. Based on the

[1]Our definition allows and even assumes a ranking value for a page x relative to C_j even if $x \notin C_j$. Most content based ranking algorithms provide such a ranking and if not, we can then assume x has rank value 0.

original PageRank algorithm, it computes a topic-sensitive ranking (i.e. cluster-sensitive in our terminology) by constraining the uniform jumping factor of a random surfer to each cluster. More precisely, let T_j be the set of pages in a cluster C_j. Then, when computing the PageRank vector for cluster C_j, in place of the uniform damping vector $h = [\frac{1}{N}]_{N\times q}$, we use the vector $h = v_j$ where

$$v_{ji} = \begin{cases} \frac{1}{|T_j|} & i \in T_j \\ 0 & i \notin T_j \end{cases}$$

Topic-Sensitive PageRank is computed as the solution to

$$TR(C_j) = (1-\epsilon)\cdot A^T \cdot TR(C_j) + \epsilon \cdot v_j$$

During query time, the cluster-sensitive ranking (Topic-Sensitive PageRank) is combined with a user's search preference (inferred from query terms provided by the user or obtained through some other advanced techniques like query-log analysis) to produce the final ranking. Given query q, using a multinomial naive-Bayes classifier or other more advanced classifier, we compute the class probabilities for each of the clusters, conditioned on q. Let $q(i)$ be the ith term in the query (or query context) q. Then, given the query q, we compute for each C_j the following:

$$Pr(C_j|q) = \frac{Pr(C_j)\cdot Pr(q|C_j)}{Pr(q)} \propto Pr(C_j)\cdot \prod_i Pr(q_i|C_j) \quad (1)$$

$Pr(q_i|C_j)$ is easily computed from the class term-vector D_j. The quantity $Pr(C_j)$ is not as straightforward. In the original Topic-Sensitive PageRank, is $Pr(C_j)$ chosen to be uniform. Certainly, more advanced techniques can be used to better estimate $Pr(C_j)$.

To compute the final rank, we retrieve all documents containing all of query terms using a text index. The final query-sensitive ranking of each of these pages is given as follows. Let $TR(x, C_j)$ be the cluster-sensitive rank of document x given by the rank vector $TR(C_j)$. For page x, we compute the final importance score $TSPR(x, q)$ as

$$TSPR(x,q) = \sum_{C_j \in C} Pr(C_j|q)\cdot TR(x, C_j)$$

One can easily check that the Topic-Sensitive PageRank algorithm is a personalized search ranking algorithm with $\mu(C_j, x, q) = TR(x, C_j)$, $P(C_j, q) = Pr(C_j|q)$ and F given by

$$\begin{aligned} & F(TR(x,C_1),\ldots,TR(x,C_m),Pr(C_1|q),\ldots Pr(C_m|q)) \\ = \;& TSPR(x,q) \end{aligned}$$

4.2 Other Personalized Systems

Aktas et al. [11] employ the Topic-Sensitive PageRank algorithm at the level of URL features such as Internet domain names. Chirita et al. [14], on the other hand, extend the Modular PageRank algorithm [20]. In [14], rather than using the arduous process for collecting the user profile as in Modular PageRank[20], the user's bookmarks are used to derive the user profile. Furthermore, they augment the set of pages obtained in this way by finding their related pages. Modified PageRank and the HITS algorithms are employed to find such related pages.

Most content based web search personalization methods are based on the idea of re-ranking the returned pages in the collection using the content of pages (represented as snippet, title, full content, etc) with respect to the user profile. In contrast to methods exploring linkage structure where the issue of automatizing the construction of the user profiles is not fully considered, some content analysis based personalization methods consider how to collect user profiles as part of its personalization framework. Liu et al.[24] propose a technique to map a user query to a set of categories, which represent the user's search intention for the web search personalization. A user profile and a general profile are learned from the user's search history and a category hierarchy respectively. Later, these two profiles are combined to map a user query into a set of categories. Chirita et al. [14] propose a way of performing web search using the ODP (open directory project) metadata. First, the user has to specify his/her search preference by selecting the set of topics (hierarchical) that he/she is interested from the ODP. Then, at run-time, the web pages returned by the ordinary search engine can be re-sorted according to the distance between the URL of a page and the user profile. Sun et al. [27] proposed an approach called CubeSVD which focuses on utilizing the click-through data to personalize the web search. Note that the click-through data is highly sparse data containing relations among user, query, and clicked web page. The analysis over this data is performed using an approach called CubeSVD which is motivated by HOSVD (High-Order Singular Value Decomposition).

5. OUR ALGORITHM

In this section, we propose a personalized search algorithm for computing cluster-sensitive page ranking based on a linear model capturing correlations between cluster content, cluster linkage, and user preference. Our model borrows heavily from the Latent Semantic Analysis (LSA) of Deerwester et al. [15], which captures term-usage information based on a simple (low-dimensional) linear model, and the SP algorithm of Achlioptas et al.[10], which captures correlations between 3 components (i.e. links, page content, user query) of web search in terms of proximity in a shared latent semantic space. We first define the notion of "local-cluster" algorithms which linearly combine cluster-sensitive page rankings. For a given clustering $\mathcal{C}$, let $CS(x) = \{C_j \in \mathcal{C} | x \in C_j\}$. Given an instance $(G, \mathcal{C}, \mu, P, q)$ of personalized search, a **local-cluster algorithm** is a personalized search ranking such that F is given by

$$\begin{aligned} & F(\mu(C_1,x,q),\ldots,\mu(C_m,i,q),P(C_1,q),\ldots P(C_m,q)) \\ = \;& \sum_{C_j \in CS(x)} P(C_j,q)\cdot\mu(C_j,x,q) \end{aligned}$$

Our algorithm personalizes existing web services utilizing existing ranking algorithms. Our model assumes that there is a generic page ranking $R(x, q)$ for ranking page x given query q. Using an algorithm to compute the ranking for clusters (described in the next section), we compute the cluster-sensitive ranking $\mu(C_i, x, q)$ as

$$\mu(C_i,x,q) = \begin{cases} R(x,q)\cdot CR(C_i,q) & \text{if } x \in C_i \\ 0 & Otherwise \end{cases}$$

where $CR(C_i, q)$ refers to the ranking of cluster C_i with respect to query q. Finally $PSR(x, q)$ will be computed as

$$PSR(x,q) = \sum_{C_j \in CS(x)} P(C_j,q)\cdot\mu(C_j,x,q)$$

We call our algorithm PSP (for Personalized SP algorithm) and note that it is a local-cluster algorithm.

5.1 Ranking Clusters

The algorithm for ranking clusters is the direct analogy of the SP algorithm [10] where now clusters play the role of pages. That is, we will be interested in the aggregation of links between clusters and the term content of clusters. We also modify the generative model of [10], so as to apply to clusters. This generative model motivates the algorithm and also allows us to formulate a correctness result for the PSP algorithm analogous to the correctness result of [10]. We note that like the SP algorithm, PSP is defined without any reference to the generative model.

Let $\{C_1, \ldots, C_m\}$ be a clustering for the targeted corpus. We assume that there is an $n \times m$ matrix Z whose (p, j) entry indicates the probability that page p is part of cluster j. Now following [15] and [10], we assume that there exists a set of k unknown (latent) basic concepts whose combinations represent every topic of the web. Given such a set of k concepts, a *topic* is a k-dimensional vector λ, describing the contribution of each of the basic concepts to this topic.

5.1.1 Authority and Hub values for clusters

We first review the notion of a page's authority and hub values as introduced in Kleinberg [22] and utilized in [10] before introducing the concept of authority and hub values for clusters. Two vectors are associated with each web page p:

- The first vector associated with p is a k-tuple $A(p) \in [0, 1]^k$ reflecting the topic on which p is an authority. The i-th entry in $A(p)$ expresses the degree to which p concerns the concept associated with the i-th entry in $A(p)$. This topic vector captures the content on which this page is an authority.
- The second vector associated with p is a k-tuple $H(p) \in [0, 1]^k$ reflecting the topic on which p is a hub. This vector is defined by the set of links from p to other pages.

Based on this notion of page hub and authority values, we introduce the concept of cluster hub and authority values. With each cluster $C_j \in C$, we associate two vectors:

- The first vector associated with C_p is a k-tuple $\widetilde{A}^{(j)}$ which represents the expected authority value that is accumulated in cluster C_j with respect to each concept. We define $\widetilde{A}^{(j)}$ as $\widetilde{A}^{(j)}(c) = \sum_{p \in C_j} Z(p, j)A(p, c)$ where $A(p, c)$ is document p's authority value with respect to the concept c and $Z(p, j)$ is the probability of document p being in cluster C_j.
- The second vector associated with C_j is a k-tuple $\widetilde{H}^{(j)}$ which represents the expected hub value that is accumulated in cluster C_j with respect to each concept. We define $\widetilde{H}^{(j)}$ as $\widetilde{H}^{(j)}(c) = \sum_{p \in C_j} Z(p, j)H(p, c)$ where $H(p, c)$ is document p's hub value with respect to the concept c.

5.1.2 Link Generation over clusters

In what follows, we assume all random variables have bounded range. Given clusters C_p and $C_r \in \mathcal{C}$, our model assumes that the total number of links from pages in C_p to pages in C_r is a random variable with expected value equal to $<\widetilde{H}^{(p)}, \widetilde{A}^{(r)}>$. Note that the intuition is the same as in the link generation model for two arbitrary documents [10]. The more closely aligned the hub topic of the pages in C_p is with the authority topic of the pages in C_r, the more likely it is that there will be a link from a document in C_p to a document in C_r. Therefore, the link generation model among different clusters is described in terms of a $m \times m$ matrix $\widetilde{W} = \widetilde{H} \cdot \widetilde{A}^T$ where the p-th row of $\widetilde{H}$ is $(H^{(p)})^T$ and the r-th row of A is $(A^{(r)})^T$. Each entry (p, r) of $\widetilde{W}$ represents the expected number of links from C_p to C_r. Let $\widehat{W}$ be the actual link structure of documents for the targeted corpus. The assumption is that $\widehat{W}$ is an instantiation of the link generation model for documents and then $\overline{W} = Z^T \widehat{W} Z$ is an instantiation of the link generation model for clusters.

5.1.3 Term Content Generation over Clusters

Once again, our term-content generation model heavily borrows from that introduced in [10]. We assume that there are l terms and the term distributions over clusters are given by the following two distributions:

- The first distribution expresses the expected number of occurrences of terms as authoritative terms within all documents. More precisely, we assume the existence of a k-tuple $\widetilde{S}_A^{(u)}$ whose c-th entry describes the expected number of occurrences of the term u in the set of *all pure authority documents* in the concept c which are not hubs on anything.
- The second distribution expresses the expected number of occurrences of terms as hub terms within all documents. More precisely, we assume the existence of a k-tuple $\widetilde{S}_H^{(u)}$ whose c-th entry describes the expected number of occurrences of the term u in the set of *all pure hub documents* in the concept c which are not authorities on anything.

The above distributions can be expressed in terms of two matrices, namely $\widetilde{S}_A$, the $l \times k$ matrix whose rows are indexed by terms, where row u is the vector $(\widetilde{S}_A^{(u)})^T$, and $\widetilde{S}_H$, the $l \times k$ matrix, whose rows are indexed by terms, where row u is the vector $(\widetilde{S}_H^{(u)})^T$. Our model assumes that terms within cluster C_p having authority value $\widetilde{A}^{(p)}$ and hub value $\widetilde{H}^{(p)}$ are generated from a distribution of bounded range where the expected number of occurrences of term u is

$$< \widetilde{A}^{(p)}, \widetilde{S}_A^{(u)} > + < \widetilde{H}^{(p)}, \widetilde{S}_H^{(u)} >$$

We describe the term generation model of clusters with a m by l matrix $\widetilde{S}$, where again m is the number of underlying clusters and l is the total number of possible terms,

$$\widetilde{S} = \widetilde{H} \cdot \widetilde{S}_H^T + \widetilde{A} \cdot \widetilde{S}_A^T$$

The (j, i) entry in $\widetilde{S}$ represents the expected number of occurrences of term i within all documents in cluster j. Let $\widehat{S}$ be the actual term-document matrix of all documents in the targeted corpus. Analogous to the previous link generation model of clusters, we instantiate our term generation model of clusters described by $\widetilde{S}$ through $\overline{S} = Z^T \widehat{S}$.

5.1.4 User Query

The user has in mind some topic on which he wants to find the most authoritative cluster of documents on the topic when he performs the search. The terms that the user

presents to the search engine should be the terms that a perfect hub on this topic would use, and then these terms would potentially lead to the discovery of the most authoritative cluster of documents on the set of topics closely related to these terms. The query generation process in our model is given as follows:

- The user chooses the k-tuple v describing the topic he wishes to search for in terms of the underlying k concepts.
- The user computes the vector $\tilde{q}^T = \tilde{v}^T \widetilde{S}_H^T$ where the u-th entry of $\tilde{q}$ is the expected number of occurrences of the term u in a cluster.
- The user then decides whether or not to include term u among his search terms by sampling from a distribution with expectation $\tilde{q}[u]$. We denote the instantiation of the random process by $\overline{q}[u]$.

The input to the search engine consists of the terms with non-zero coordinates in the vector $\overline{q}$.

5.1.5 Algorithm Description

Given this generative model that incorporates link structure, content generation, user preference, and query, we can rank clusters of documents using a spectral method. While the basic idea and analysis for our algorithm follows from [10], our PSP algorithm is different from the original SP algorithm in one substantial aspect. *In contrast to the original SP algorithm which works at the document level, our algorithm works at the cluster level making our algorithm computationally more attractive and consequently more practical*[2]. For our algorithm, in addition to the SVD computation of $\widetilde{M}$ and $\widetilde{W}$ matrices, the SVD computation of $\widetilde{S}$ is also required. This additional computation is not very expensive because of the size of matrix $\widetilde{S}$.

We need some additional notation. For two matrices A and B with an equal number of rows, let $[A|B]$ denote the matrix whose rows are the concatenations of the rows of A and B. Let $\sigma_i(A)$ denote the i-th largest singular value of a matrix A. Let $r_i(B) \geq 1$ denote the ratio between the primary singular value and the i-th singular value of B: $r_i(B) = \sigma_1(B)/\sigma_i(B)$. Let $[0^n]$ denote a row vector with n zeros, and let $[0^{i\times j}]$ denote an all zero matrix of dimensions $i \times j$. We use a standard notation for the singular value decomposition (SVD) of a matrix. More precisely, given a matrix $B \in R^{n\times m}$, let the singular value decomposition (SVD) of B be $U\Sigma V^T$ where U is a matrix of dimensions $n \times rank(B)$ whose columns are orthonormal, Σ is a diagonal matrix of dimensions $rank(B) \times rank(B)$, and V^T is a matrix of dimensions $rank(B) \times m$ whose rows are orthonormal. The (i, i) entry of Σ is $\sigma_i(B)$.

The cluster ranking algorithm performs the following preprocessing of the entire corpus of documents independent of the query.

Pre-processing Step

1. Let $\overline{M} = [\overline{W}^T|\overline{S}]$. Recall that $\overline{M} \in R^{m\times(m+l)}$ (m is the number of clusters and l is the number of terms). Compute the SVD of the matrix as $\overline{M}^* = U_{\overline{M}}\Sigma_{\overline{M}}V_{\overline{M}}^T$
2. Choose the largest index r such that the difference $|\sigma_r(\overline{M}^*) - \sigma_{r+1}(\overline{M}^*)|$ is sufficiently large (we require $\omega(\sqrt{(m+l)})$). Let $\overline{M}_r^* = (U_{\overline{M}})_r(\Sigma_{\overline{M}})_r(V_{\overline{M}}^T)_r$ be the rank r-SVD approximation to $\overline{M}$.
3. Compute the SVD of the matrix $\overline{W}$ as $\overline{W}^* = U_{\overline{W}}\Sigma_{\overline{W}}V_{\overline{W}}^T$
4. Choose the largest index t such that the difference $|\sigma_t(\overline{W}^*) - \sigma_{t+1}(\overline{W}^*)|$ is sufficiently large (we require $\omega(\sqrt{(t)})$). Let $\overline{W}_t^* = (U_{\overline{W}})_t(\Sigma_{\overline{W}})_t(V_{\overline{W}}^T)_t$ be the rank t-SVD approximation to $\overline{W}$.
5. Compute the SVD of the matrix $\overline{S}$ as $\overline{S}^* = U_{\overline{S}}\Sigma_{\overline{S}}V_{\overline{S}}^t$
6. Choose the largest index o such that the difference $|\sigma_o(\overline{S}^*) - \sigma_{o+1}(\overline{S}^*)|$ is sufficiently large (we require $\omega(\sqrt{(o)})$). Let $\overline{S}_o^* = (U_{\overline{S}})_o(\Sigma_{\overline{S}})_o(V_{\overline{S}}^T)_o$ be the rank o-SVD approximation to $\overline{S}$.

Query Step

Once a query vector $\overline{q}^T \in R^l$ is presented, let $\overline{q'}^T$=$[0^m|\overline{q}^T]$ $\in R^{m+l}$. Then, we compute the vector

$$w^T = \overline{q'}^T \overline{M^*}_r^{-1} \overline{W}_t^*$$

where $\overline{M^*}_r^{-1} = (V_{\overline{M}}^T)_r(\Sigma_{\overline{M}})_r^{-1}(U_{\overline{M}})_r$ is the pseudo-inverse of $\overline{M}_r$.

The next theorem formalizes the correctness of the algorithm with respect to the generative model.

THEOREM 2. *Assume that the link structure for clusters, term content for clusters and search query are generated as described in our model: $\overline{W}$ is an instantiation of $\widetilde{W} = \widetilde{H}\widetilde{A}^T$, $\overline{S}$ is an instantiation of $\widetilde{S} = \widetilde{A}\widetilde{S}_A^T + \widetilde{H}\widetilde{S}_H^T$, $\overline{q}$ is an instantiation of $\tilde{q} = v^T\widetilde{S}_H^T$, the user's preference is provided by p^T. Additionally, we have*

1. *$\overline{q}$ has $\omega(k \cdot r_k(\overline{W})^2 r_{2k}(\overline{M})^2 r_k(G^T))$ terms.*
2. *$\sigma_k(\overline{W}) \in \omega(r_{2k}(\overline{M})r_k(G^T)\sqrt{m})$ and $\sigma_{2k}(\overline{M}) \in \omega(r_k(\overline{W})$ $r_{2k}(\overline{M})r_k(G^T)\sqrt{m})$,*
3. *$\overline{W}$, $\overline{H}\overline{S}_A^T$ and $\overline{S}_H^T$ are rank k, $\overline{M} = [\overline{W}^T|\overline{S}]$ is rank $2k$, $l = O(m)$, and $m = O(k)$.*

then the algorithm computes a vector of authorities that is very close to the correct ranking. More precisely, we have

$$\frac{||\overline{q'}^T \overline{M^*}_r^{-1} \overline{W}_t^* \cdot p^T \cdot \overline{S}_o^{*T} - v^T\widetilde{A}^T p^T \widetilde{S}^T||_2}{||v^T\widetilde{A}^T p^T \widetilde{S}^T||_2} \in O(1)$$

The proof of this theorem is similar to that of Achlioptas et al. [10]. We present the proof in the full version.

5.2 Final Ranking

Once we have computed the ranking for clusters, we proceed with the actual computation of cluster-sensitive page ranking. Let $w^T(j)$ denote the authority value of cluster C_j as computed in the previous section. The cluster-sensitive page rank for page x with respect to cluster C_j is computed as

$$\mu(x, C_j, q) = \begin{cases} R(x,q) \cdot w(j) & \text{if } x \in C_j \\ 0 & \text{Otherwise} \end{cases}$$

where again $R(x, q)$ is the generic rank of page x with respect to query q.

[2]To the best of our knowledge, the SP algorithm was never implemented in practice

As discussed in Section 1, we assume that the user provides his search preference having in mind certain clusters (types of documents that he/she is interested). If the user exactly knows what the given clusters are, then he might directly express his search preference over these clusters. However, such explicit preferences will not generally be available. Instead, we consider a more general scenario in which the user expresses his search interests through a set of keywords (terms). More precisely, our user search preference is given by:

- The user expresses his search preference by providing a vector $\tilde{p}^T$ over terms whose i-th entry indicates his/her degree of preference over the term i.
- Given the vector $\tilde{p}^T$, the preference vector over clusters is obtained as $\tilde{p}^T \cdot \widetilde{S}^T$.

Let $\tilde{p}^T \widetilde{S}^T(j) = P^T(C_j)$ denote this preference for cluster C_j. The final personalized rank for page x is computed as

$$PSP(x,q) = \sum_{C_i \in CS(x)} R(x,q) \cdot w(i) \cdot P^T(C_i)$$

or in a matrix form as $R^T I_n Z P^T I_m w$

6. PERSONALIZED SEARCH CRITERIA

We present a series of results comparing Topic-Sensitive PageRank algorithm and our PSP algorithm with respect to a set of personalized search algorithm criteria that we propose. Our criteria are all of the form "small changes in the input imply small changes in the computed ranking". We believe such criteria are a practical necessity as well as of theoretical interest. All proofs are given in the Appendix. Since our ranking of documents produces real authority values in $[0, 1]$, one natural approach is to study the effect of small continuous changes in the input information as in the rank stability studies of [12, 16, 23, 25].

One basic property shared by both Topic-Sensitive PageRank and our PSP algorithm is continuity.

THEOREM 3. *Both TSPR and our PSP ranking algorithms are continuous; i.e. small changes in any μ value or preference value will result in a small change in the ranking value of all pages.*

Our first distinguishing criteria is a rather minimal *monotonicity property* that we claim any personalized search should satisfy. Namely, since a (cluster based) personalized ranking function depends on the ranking of pages within their relevant clusters as well as the preference of clusters, when these rankings for a page and cluster preferences are increased, we expect the personalized rating can only improve. More precisely, we have the following definition:

DEFINITION 4. *Let $(G, \mathcal{C}, \mu, P, q)$ and $(G, \mathcal{C}, \mu, \tilde{P}, q)$ be two instances of personalized search. Let χ and ψ be the set of ranked pages produced by $(G, \mathcal{C}, \mu, P, q)$ and $(G, \mathcal{C}, \mu, \tilde{P}, q)$ respectively. Suppose that $x \in \chi$, $y \in \psi$ share the same set of clusters (i.e. $CS(x) = CS(y)$), and suppose that $\mu(C_j, x, q) \leq \mu(C_j, y, q)$ and $P(C_j, q) \leq \tilde{P}(C_j, q)$ hold for every C_j that they share. We say that a personalized ranking algorithm is **monotone** if $PSR(x) \leq \tilde{PSR}(y)$ for every such $x \in \chi$ and $y \in \psi$.*

We now introduce the idea of "locality". The idea behind locality is that (small) discrete changes in the cluster preferences should have only a minimal impact on the ranking of pages. The notion of locality justifies our use of the terminology "local-cluster algorithm". A **perturbation ∂_α of size** α changes a cluster preference vector P to a new preference vector $\tilde{P} = \partial_\alpha(P)$ such that P and $\tilde{P}$ differ in at most α components. Let $\tilde{PSR}$ denote the new personalized ranking vector produced under the new search preference vector $\tilde{P}$.

DEFINITION 5. *Let $(G, \mathcal{C}, \mu, P, q)$ and $(G, \mathcal{C}, \mu, \tilde{P}, q)$ be the original personalized search instance and its perturbed personalized search instance respectively. Let $AC(\partial_\alpha)$, the active clusters, be the set of clusters that are affected by the perturbation ∂_α (i.e., $P(C_j, q) \neq \tilde{P}(C_j, q)$ for every cluster C_j in $AC(\partial_\alpha)$). We say that a personalized ranking algorithm is **local** if for every $x, y \notin AC(\partial_\alpha)$, $PSR(x, q) \leq PSR(y, q)$ $\Leftrightarrow \tilde{PSR}(x, q) \leq \tilde{PSR}(y, q)$ where PSR refers to the original personalized ranking vector while $\tilde{PSR}$ refers to the personalized ranking vector after the perturbation.*

THEOREM 6. *Topic-Sensitive PageRank algorithm is not monotone and not local*

In contrast we show that our PSP algorithm does enjoy the monotone and local properties.

THEOREM 7. *Any linear local-cluster algorithm (and hence PSP) is monotone and local.*

We next consider a notion of stability (with respect to cluster movement) in the spirit of [25, 23]. Our definition reflects the extent to which small changes in the clustering can change the resulting rankings. We consider the following page movement changes to the clusters:

- A **migration** $migr(x, C_i, C_j)$ moves page x from cluster C_i to cluster C_j.
- A **replication** $repl(x, C_i, C_j)$ adds page x to cluster C_j (assuming x was not already in C_j) while keeping x in C_i.
- A **deletion** $del(x, C_j)$ is the deletion of page x from cluster C_j (assuming there exists a cluster C_i in which x is still present).

We define the *size* of these three page movement operations to be $\mu(C_i, x, q) + \mu(C_j, x, q)$ for migration/replication, and $\mu(C_j, x, q)$ for deletion. We measure the size of a collection M of page movements to be the sum of the individual page movement costs. Our definition of stability then is that the resulting ranking does not change significantly when the clustering is changed by page movements of small size.

We recall that each cluster is a set of pages and its induced subgraph, induced from the graph on all pages. We will assume that the μ ranking algorithm is a stable algorithm in the sense of [25, 23]. Roughly speaking, locality of a μ ranking algorithm means that there will be a relatively small change in the ranking vector if we add or delete links to a web graph. Namely, the change in the ranking vector will be proportional the ranking values of the pages adjacent to the new or removed edges.

Query Used	Relevant Categories
middle east	[Society/Issues] [News/Current Events] [Recreation/Travel]
long distance	[Business/Telecommunications] [Sports/Walking] [Society/Relationships]
integration	[Computers/Software] [Health/Alternative] [Society/Issues]
proverb	[Society/Folklore] [Reference/Quotations] [Home/Homemaking]
fishing expedition	[Recreation/Camps] [Sports/Adventure Racing] [Recreation/Outdoors]
northern lights	[Science/Astronomy] [Kids and Teens/School Time] [Science/Software]
star wars	[Arts/Movies] [Games/Video Games] [Recreation/Models]]
strong man	[Sports/Strength Sports] [World/Deutsch] [Recreation/Drugs]
conservative	[Society/Politics] [News/Analysis and Opinion] [Society/Religion and Spirituality]
liberal	[Society/Politics] [News/Analysis and Opinion] [Society/Religion and Spirituality]
popular blog	[Arts/Weblogs] [Arts/Chats and Forums] [News/Weblogs]
common tricks	[Arts/Writers Resources] [Games/Video Games] [Home/Do It Yourself]
chaos	[Science/Math] [Society/Religion and Spirituality] [Games/Video Games]
english	[Arts/Education] [Kids and Teens/School Time] [Society/Ethnicity]
war	[Society/History] [Games/Board Games] [Reference/Museums]
jaguar	[Recreation/Autos] [Sports/Football] [Science/Biology]
technique	[Science/Methods and Techniques] [Arts/Visual Arts] [Shopping/Crafts]
vision	[Health/Senses] [Computers/Artificial Intelligence] [Business/Consumer Goods and Services]
graphic design	[Business/Publishing and Printing] [Computers/Graphics] [Arts/Graphic Design]
environment	[Business/Energy and Environment] [Science/Environment] [Arts/Genres]

Table 1: Sample queries and the preferred categories for search used in our experiments

DEFINITION 8. *Let $(G, \mathcal{C}, \mu, P, q)$ and $(G, \mathcal{C}, \mu, \tilde{P}, q)$ be a personalized search instance. A personalized ranking function PSR is* ***cluster movement stable*** *if for every set of page movements M there is a β, independent of G, such that*

$$||PSR - P\tilde{S}R||_2 \leq \beta \cdot size(M)$$

where PSR refers to the original personalized ranking vector while $P\tilde{S}R$ refers to the personalized ranking vector produced when the set of page movements M has been applied to a given personalized search instance.

THEOREM 9. *Topic-Sensitive PageRank algorithm is not cluster movement stable.*

THEOREM 10. *The PSP algorithm is cluster movement stable.*

7. EXPERIMENTS

As a proof of concept, we implemented both the PSP algorithm and the Topic-Sensitive PageRank algorithm for comparison. In section 7.1, we investigate the retrieval effectiveness of our PSP algorithm versus that of the Topic-Sensitive PageRank algorithm. In section 7.2, we study the sensitivity of algorithms when the user's search preference is perturbed.

As a source of data, we used the Open Directory Project (ODP) [3] data, which is the largest and most comprehensive human-edited directory in the Web. We first obtained a list of pages and their respective categories from the ODP site. Next, we fetched all pages in the list , and parsed each downloaded page to extract its pure text and links (without nepotistic links). We treat the set of categories in the ODP that are at distance two from the root category (i.e. the "Top" category) as the cluster set for our algorithms. In this way, we constructed 549 categories (or clusters) in total. The categorization of pages using these categories did not constitute a partition as some pages (5.4% of ODP data) belong to more than one category.

Query	PSP	Topic-Sensitive PageRank
middle east	0.76	0.8
planning	0.96	0.56
integration	0.6	0.16
proverb	0.9	0.83
fishing expedition	0.86	0.66
northern lights	0.7	0.8
star wars	0.6	0.66
strong man	0.9	0.86
conservative	0.86	0.76
liberal	0.76	0.73
popular blog	0.93	0.7
common tricks	0.66	0.9
chaos	0.56	0.56
english	0.8	0.26
war	0.83	0.16
jaguar	0.96	0.46
technique	0.96	0.7
vision	0.43	0
graphic design	1	0.73
environment	0.93	0.5
Average	**0.80**	0.59

Table 2: Performance of PSP and Topic-Sensitive PageRank

7.1 Comparison of Algorithms

To produce rankings, we first retrieved all the pages that contained all terms in a query, and then computed rankings taking into account the specified categories (as explained below). The PSP algorithm assumes that there is already an underlying page ranking for the given web service. Since we were not aware of the ranking used by the ODP search, we simply used the pure PageRank as the generic page ranking for our PSP algorithm. The Topic-Sensitive PageRank was implemented as described in Section 4.1.1. We used the same $\alpha = 0.25$ value used in [19].

We devised 20 sample queries and their respective search preferences (in terms of categories) as shown in Table 1. These "preferred" categories were chosen, heuristically, after inspecting the ranking results returned by the ODP search for each query in Table 1. For the Topic-Sensitive PageRank algorithm, we did not use the approach for automatically discovering the search preference (See Eq. 1) from a given query since we found that the most probable categories discovered in this way were heavily biased toward "News" related categories. Instead, we computed both Topic-Sensitive

[3] http://www.dmoz.com

query	category	PSP	TSPR
middle east	Society/Issues	51.17	6.17
	News/Current Events	3.67	14.17
	Recreation/Travel	31.50	19.50
long distance	Business/Telecommunications	0.00	0.00
	Sports/Walking	3.00	3.00
	Society/Relationships	81.50	74.50
integration	Computers/Software	0.00	0.00
	Health/Alternative	0.00	0.00
	Society/Issues	88.92	54.08
proverb	Society/Folklore	70.00	59.00
	Reference/Quotations	26.00	36.00
	Home/Homemaking	2.00	2.00
fishing expedition	Recreation/Camps	2.50	2.50
	Sports/Adventure Racing	1.00	1.00
	Recreation/Outdoors	55.00	55.00
northern lights	Science/Astronomy	62.33	62.33
	Kids and Teens/School Time	22.83	22.83
	Science/Software	0.50	0.50
star wars	Arts/Movies	22.83	11.33
	Games/Video Games	69.83	32.33
	Recreation/Models	0.00	42.00
conservative	Society/Politics	53.17	20.33
	News/Analysis and Opinion	8.00	56.00
	News/Religion and Spirituality	30.00	1.00
liberal	Society/Politics	48.33	26.67
	News/Analysis and Opinion	4.50	49.50
	News/Religion and Spirituality	36.83	1.00
chaos	Science/Math	11.33	49.91
	Society/Religion and Spirituality	28	3
	Games/Video Games	57	30.00
english	Arts/Education	17.83	43.66
	Kids and Teens/School Time	35.16	8.33
	Society/Ethnicity	23.5	5.66
war	Society/History	65.75	15.41
	Games/Board Games	0.5	0.5
	Reference/Museums	8.25	9.08
jaguar	Recreation/Autos	53.83	68
	Sports/Football	15.5	15.5
	Science/Biology	26	0
technique	Science/Methods and Techniques	2	17
	Arts/Visual Arts	36.5	6.5
	Shopping/Crafts	55.5	42.5
graphic design	Business/Publishing and Printing	0.5	0.5
	Computers/Graphics	0	0
	Arts/Graphic Design	94	92.5
environment	Business/Energy and Environment	3.16	3.3
	Science/Environment	74.41	26.25
	Arts/Genres	1	17.5

Table 3: Distribution of the preferred categories in the top 100 pages

PageRank and PSP rankings by equally weighting all categories listed in Table 1.

The evaluation of ranking results was done by three individuals: two having CS degrees with extensive web search experience and the third person having an engineering degree, but also with extensive web search experience. We used the precision over the top-10 (p@10) as the evaluation measure. For each query, we merged the top 10 results returned by both algorithms into a single list. Without any prior knowledge about what algorithm was used to produce the corresponding result, each person was asked to carefully evaluate each page from the list as "relevant" if in their judgment the corresponding page should be treated as a relevant page with respect to the given query and one of the specified categories, or *non-relevant* otherwise. In Table 2, we summarize the evaluation results where the presented precision value is the average of all 3 precision values. These evaluation results suggest that our PSP algorithm outperforms the Topic-Sensitive PageRank algorithm.

To gain further insight, we analyzed the distribution of categories associated with each produced ranking. An ideal personalized search algorithm should retrieve pages in clusters representing the user's specified categories as the top ranked pages. Therefore, in the list of top 100 pages associated with each query, we computed how many pages were associated with those categories specified in each search preference. Each page p in the list of top 100 pages was counted as $1/|nc(p)|$ where $nc(p)$ is the total number of categories associated with page p. We report on these results in Table 3. The presented results are excluding those queries like "strong man" (only 27 pages retrieved), "popular blog" (only 26 pages retrieved), "common tricks" (only 74 pages retrieved), and "vision" (only 4 pages retrieved), which did not retrieve a sufficient number of relevant pages in their lists of top 100 pages. Note that the total sum of all three preferred categories for each query was always less than 100 since several pages pertain to more than one category. For several queries in Table 3, one can observe that each algorithm's favored category is substantially different. For instance, for the query "star wars", the PSP algorithm prefers "Games/Video Games" category while the Topic-Sensitive PageRank prefers "Recreation/Models" category. Furthermore, for the queries "liberal", "conservative", "technique" and "english" the PSP algorithm and the Topic-Sensitive PageRank algorithm share a very different view on what the most important context associated with "liberal", "conservative", "technique", and "english" is. One should also observe that when there is a highly dominant query context (e.g. "Society/ Relationships" category for "long distance", "Society/Issues" category for "integration, and "Arts/Graphic Design" for "graphic design") over other query contexts, then for both algorithms the rankings are dominated by this strongly dominant category with PSP being somewhat more focused on the dominant category. Finally, averaging over all queries, 86.38% of pages in the PSP list of top 100 pages were found to be in the specified preferred categories while for Topic-Sensitive PageRank, 69.05% of pages in the list of top 100 pages were found to be in the specified preferred categories.

We compared the PSP and TSPR rankings using a variant of the Kendall-Tau similarity measure[19, 17], so as to measure the probability that the two partial rankings (i.e. their rankings might not overlap) agree on the relative ordering of two distinct pages selected at random. Consider two partially ordered rankings σ_1 and σ_2, each of length n. Let U be the union of the elements in σ_1 and σ_2. If δ_1 is $U - \sigma_1$, then let σ_1' be the extension of σ_1, where σ_1' contains δ_1 appearing after all the URLs in σ_1. We do the analogous extension or σ_2 to obtain σ_2'. Then define

$$KTSim(\sigma_1, \sigma_2) = \frac{|\{(u,v) : \sigma_1', \sigma_2' \text{ agree on order of } (u,v), u \neq v\}|}{|U||U-1|}$$

Using this KTSim measure, we computed the pairwise similarity between the PSP and TSPR rankings with respect to each query. Averaging over all queries, the KTSim value for the top 100 pages is 0.58 while the average KTSim value for the top 20 pages is 0.43, indicating a substantial difference in the rankings.

7.2 Locality

We conducted a study on how sensitive the algorithms are to change in search preferences. We argued that such

sensitivity is theoretically captured by the notion of locality in Section 6, and theoretically showed that the PSP algorithm is robust to the change in search preferences while the Topic-Sensitive PageRank algorithm is not. Our experimental evidence indicates that the Topic-Sensitive PageRank algorithm is somewhat more sensitive to the change in search preferences. Let Δ_α^N refer to the set of all size α perturbations (over preference vectors) on a clustering of an N node graph. Given $\partial_i, \partial_j \in \Delta_\alpha^N$, let PSR_i and PSR_j denote the personalized ranking vectors computed under ∂_i and ∂_j respectively. To compare the personalized ranking vectors produced under different perturbations, we again use the KTSim measure [19, 17].

We studied the variation of $KTSim(PSR_i, PSR_j)$ for different ∂_i and $\partial_j \in \Delta_\alpha^N$. For each query, the original search preference consisted of 7 randomly selected categories, and we varied α as 1,3, and 5. For a fixed α and for 5 random $\partial_i \in \Delta_\alpha^N$, we computed the pairwise similarity (KTSim) considering the top 100 pages. In Table 4, we report on the average pairwise similarity across all queries for each fixed α.

α	PSP	Topic-Sensitive PageRank
1	0.91	0.92
3	0.77	0.69
5	0.79	0.66

Table 4: Average KTSim values of rankings under different perturbation sizes across all queries

The $KTSim$ values in Table 4 suggest that our PSP algorithm is less sensitive to the change in search preferences than the Topic-Sensitive PageRank algorithm.

8. CONCLUSION

We have developed and implemented a computationally efficient "local-cluster" algorithm (PSP) for personalized search. Following [10], we can prove the correctness of the PSP algorithm relative to a probabilistic generative model. We propose some formal criteria for evaluating personalized ranking algorithms, and demonstrate both theoretically and experimentally that our algorithm is a good alternative to the Topic-Sensitive PageRank algorithm.

9. REFERENCES

[1] About.com. http://www.about.com.
[2] Citysearch. http://www.citysearch.com.
[3] Google local. http://local.google.com.
[4] Google news. http://news.google.com.
[5] Open directory project. http://www.dmoz.org.
[6] Topix. http://www.topix.net.
[7] Yahoo. http://www.yahoo.com.
[8] Yahoo local. http://local.yahoo.com.
[9] Yahoo! mindset. http://mindset.research.yahoo.com.
[10] D. Achilioptas, A. Fiat, A. R. Karlin, and F. McSherry. Web search via hub synthesis. In *FOCS*, pages 500–509. ACM, 2001.
[11] M. Aktas, M. Nacar, and F. Menczer. Personalizing pagerank based on domain profiles. In *WebKDD*. ACM, 2004.
[12] A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas. Link analysis ranking: algorithms, theory, and experiments. *ACM Trans. Internet Techn.*, 5(1):231–297, 2005.
[13] S. Brin and L. Page. The anatomy of a large-scale hypertextual search engine. In *Computer Networks*, pages 107–117. ACM, 1998.
[14] P. A. Chirita, W. Nejdl, R. Paiu, and C. Kohlschuetter. Using odp metadata to personalize search. In *SIGIR*. ACM, 2005.
[15] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 41(6):391–407, 1990.
[16] D. Donato, S. Leonardi, and P. Tsaparas. Stability and similarity of link analysis ranking algorithms. In *ICALP*, pages 717–729, 2005.
[17] R. Fagin, R. Kumar, and D. Sivakumar. Comparing top k lists. In *SODA*, pages 28–36, 2003.
[18] P. Ferragina and A. Gulli. A personalized search engine based on web-snippet hierarchical clustering. In *WWW (Special interest tracks and posters)*, pages 801–810, 2005.
[19] T. H. Haveliwala. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):784–796, 2003.
[20] G. Jeh and J. Widom. Scaling personalized web search. In *WWW*, pages 271–279, 2003.
[21] S. Kamvar, T. Haveliwala, C. Manning, and G. Golub. Exploiting the block structure of the web for computing pagerank. Technical Report Stanford University Technical Report, Stanford University, March 2003.
[22] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.
[23] H. C. Lee and A. Borodin. Perturbation of the hyper-linked environment. In *COCOON*, pages 272–283, 2003.
[24] F. Liu, C. T. Yu, and W. Meng. Personalized web search by mapping user queries to categories. In *CIKM*, pages 558–565, 2002.
[25] A. Y. Ng, A. X. Zheng, and M. I. Jordan. Link analysis, eigenvectors and stability. In *IJCAI*, pages 903–910, 2001.
[26] F. Qiu and J. Cho. Automatic identification of user interest for personalized search. In *WWW*, 2006.
[27] J.-T. Sun, H.-J. Zeng, H. Liu, Y. Lu, and Z. Chen. Cubesvd: a novel approach to personalized web search. In *WWW*, pages 382–390, 2005.
[28] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *SIGIR*, pages 449–456, 2005.

10. APPENDIX

Proof-Sketch for Theorem 2

The continuity of Topic-Sensitive PageRank and PSP easily follow from the way how these algorithms produce the final ranking. Both algorithms linearly combine μ and P to produce the final ranking. That is, for both algorithms the final rank vector $FR(q)$ with respect to query q can be written as $FR(q) = \Gamma(q) \cdot P(q)$ where $\Gamma(q)$ is a $n \times m$ matrix whose (i,j)th-entry denotes $\mu(C_j, x_i, q)$, and $P(q)$ denotes the cluster preference vector.

We first prove the continuity of algorithms with respect to cluster preference vector. Given $\epsilon > 0$, we have $||\Gamma(q) \cdot P(q) - \Gamma(q) \cdot \tilde{P}(q)||_2 \leq ||\Gamma(q)||_F ||P(q) - \tilde{P}(q)||_2 < \epsilon$. Therefore, $\delta = \frac{\epsilon}{m}$ would be sufficient for achieving the continuity of algorithms with respect to cluster preference vector. The continuity with respect to μ can be proved in a similar fashion.

Proof of Theorem 5

Monotonicity: We present a counter-example to show that topic-sensitive PageRank is not monotone. Suppose that G is a graph that consists of 4 points $\{x_1, x_2, x_3, x_4\}$. Let $C = \{C_1, C_2, C_3\}$ be a clustering of G such that x_1, $x_2 \in C_1$, $x_3 \in C_2$, and $x_4 \in C_3$. Let assume that $x_3 \rightarrow x_1$, $x_4 \rightarrow x_1$, and $x_4 \rightarrow x_2$. In addition, we assume $\epsilon >=$

0.25. We have $TR(x_1, C_1) = \frac{\epsilon}{2} + (1-\epsilon)$, $TR(x_2, C_1) = \frac{\epsilon}{2} + (1-\epsilon)$, $TR(x_1, C_2) = (1-\epsilon)\epsilon$, $TR(x_1, C_3) = (1-\epsilon)\frac{\epsilon}{2}$, and $TR(x_2, C_2) = 0$, $TR(x_2, C_3) = (1-\epsilon)\frac{\epsilon}{2}$. Moreover, we assume $P(C_1, q) = \frac{2}{5}$, $P(C_2, q) = 1$, $P(C_3, q) = 1$, $\tilde{P}(C_1, q) = \frac{3}{5}$, $\tilde{P}(C_2, q) = 1$ and $\tilde{P}(C_3, q) = 1$. Therefore, all conditions of monotonicity are satisfied. However, we have

$$
\begin{aligned}
& TSPR(x_1) = P(C_1, q)TR(x_1, C_1) + P(C_2, q)TR(x_1, C_2) \\
+ \quad & P(C_3, q) \cdot TR(x_1, C_3) = \frac{2}{5}(\frac{\epsilon}{2} + (1-\epsilon)) + (1-\epsilon)\epsilon + \\
& (1-\epsilon)\frac{\epsilon}{2} > \frac{3}{5}(\frac{\epsilon}{2} + (1-\epsilon)) + (1-\epsilon)\epsilon + (1-\epsilon)\frac{\epsilon}{2} \\
= \quad & \tilde{TSPR}(x_2) = \tilde{P}(C_1, q)TR(x_2, C_1) + \tilde{P}(C_2, q)TR(x_2, C_2) \\
+ \quad & \tilde{P}(C_3, q) \cdot TR(x_2, C_3)
\end{aligned}
$$

Non-locality: We present a counter-example to show that topic-sensitive PageRank is not local. In particular, we show that a small perturbation in preference values can have considerably large impact on the overall ranking. Let $G = C_1 \sqcup C_2 \sqcup C_3 \sqcup C_4$, $|C_1| = |C_2| = N - \beta$ and $|C_3| = |C_4| = \beta$ where β is a fixed constant. Every page in $C_3 \sqcup C_4$ points to every page in $C_1 \sqcup C_2$. One can verify that for each $x \in C_1$ and $y \in C_2$ we have $TSPR(x, C_1) = TSPR(y, C_2)$, and similarly we have $TSPR(x, C_2) = TSPR(y, C_1)$. Furthermore, $TSPR(x, C_3) = TSPR(x, C_4) = TSPR(y, C_3) = TSPR(y, C_4)$. Now, suppose that the original cluster preferences are altered from $P(C_1, q) = P(C_2, q)$, $P(C_3, q) < P(C_4, q)$ to $\tilde{P}(C_1, q) = \tilde{P}(C_2, q)$, $\tilde{P}(C_3, q) > \tilde{P}(C_4, q)$. From the original cluster preferences, we will have $TSPR(x) < TSPR(y)$ for $x \in C_1$, $y \in C_2$. On the other hand, from the modified cluster preferences, we will have $\tilde{TSPR}(x) > \tilde{TSPR}(y)$ for $x \in C_1$, $y \in C_2$. That is, we have shown non-locality. More precisely, $d_r(PSR, \tilde{PSR}) = (N-\beta)^2 \in o((2N)^2)$ as $2N \to \infty$.

Proof of Theorem 6

Monotonicity: Since by the assumption, for every $C_j \in CS(x) = CS(y)$, we have $P(C_j, q) \leq \tilde{P}(\tilde{C}_j, q)$, and $\mu(C_j, x, q) \leq \mu(C_j, y, q)$, we will have $PSR(x) = \sum_{C_j \in CS(x)} P(C_j, q)\ \mu(C_j, x, q) \leq \sum_{C_j \in CS(y)} P(C_j, q)\ \mu(C_j, y, q) = \tilde{PSR}(x)$.

Locality: It easily follows from the fact that the ranking produced local-cluster algorithms are only based on those clusters containing the point to be ranked. Therefore, the original ranking for points in UC is unaffected by the perturbation.

Proof of Theorem 8 We exhibit a counter-example to show that the Topic-Sensitive PageRank is not stable. Let G be a graph that consists of $n+1$ points and 3 clusters C_1, C_2 and C_3. C_1 contains x_0, C_2 contains $\{x_2, \ldots, x_n\}$ and C_3 contains all points. We have $x_0 \to x_1$, $x_n \to x_1$ and $x_k \to x_{k+1}$ for every $1 < k < (n-1)$. Furthermore, suppose that $P(C_1, q) = 1$, $P(C_2, q) = 0$, and $P(C_3, q) = 0$ [4]. One can verify that $TSPR(x_0) = TR(x_0, C_1) = \epsilon$, $TSPR(x_n) = TR(x_n, C_1) = \delta$ and $TSPR(x_m) = TR(x_m, C_1) = (1-\epsilon)^m(\epsilon + \delta)$ where $\delta = \frac{\epsilon(1-\epsilon)^n}{1-(1-\epsilon)^n}$ for every $1 \leq m < (n-1)$. On the other hand, one can easily see that $TR(x_i, C_2) = 1/n$ for every $1 \leq i \leq n$. Now, we delete x_1 from C_2. One can see that $\tilde{TSPR}(x_0, C_1) = \tilde{TR}(x_0, C_1) = 1$, and $\tilde{TSPR}(x_i, C_1) = \tilde{TR}(x_i, C_1) = 0$ for every $1 \leq i \leq n$. We have

$$
\begin{aligned}
& \frac{||TSPR - \tilde{TSPR}||_2}{size(del(x_1, C_2))} = n \cdot \sqrt{((\epsilon-1)^2 + \sum_{i=1}^{n-1}((1-\epsilon)^i(\epsilon+\delta))^2} \\
= \quad & n\sqrt{((\epsilon-1)^2 + (\frac{1-(1-\epsilon)^{2n}}{1-(1-\epsilon)^2} - 1)(\epsilon+\delta)^2)} \geq n|(\epsilon-1)|
\end{aligned}
$$

which is unbounded with respect to n.

Proof-Sketch for Theorem 9

We only consider replication and deletion as migration $migr(x_a, C_i, C_j)$ can be seen as a sequential application of $repl(x_a, C_i, C_j)$ followed by $del(x_a, C_i)$. To simplify our notation, we will simply use R^T to refer to $R^T I_n$ and P^T to refer to $P^T I_m$. Let $R^T Z P^T \omega$ be the ranking before the page movement. Let $\tilde{Z} = Z + E$ be the new matrix representing the page's membership in a cluster where E is given as $E_{a,j} = 1$ if it is $repl(x_a, C_i, C_j)$ and $E_{a,i} = -1$ if it is $del(x_a, C_i)$ while the rest of entries are all zero. Let $R^T \tilde{Z} P^T \tilde{\omega}$ be the ranking after the page movement. We will show that $\frac{||RZP^T\omega - R\tilde{Z}P^T\tilde{\omega}||_2}{||\lambda^R R\lambda^\omega \omega||_1}$ is bounded by a constant where λ^R is the projection of R over the affected page (e.g. $\lambda^R_{ij} = 1$ for $i, j = a$ and $\lambda^R_{ij} = 0$ otherwise for $repl(x_a, C_i, C_j)$) while P_ω is the projection of ω over the affected clusters (e.g. $\lambda^R_{fd} = 1$ for $f = a$, $d = i, j$ for $repl(x_a, C_i, C_j)$). We have $\frac{||RZP^T\omega - R\tilde{Z}P^T\tilde{\omega}||_2}{||\lambda^R R\lambda^\omega\omega||_1} \leq \frac{||RZP^T\omega - R\tilde{Z}P^T\omega + R\tilde{Z}P^T\omega - R\tilde{Z}P^T\tilde{\omega}||_2}{||\lambda^R R\lambda^\omega\omega||_2}$ $\leq \frac{||RZP^T\omega - R\tilde{Z}P^T\omega||_2}{||\lambda^R R\lambda^\omega\omega||_2} + \frac{||R\tilde{Z}P^T\omega - R\tilde{Z}P^T\tilde{\omega}||_2}{||\lambda^R R\lambda^\omega\omega||_2}$. The first term $\frac{||RZP^T\omega - R\tilde{Z}P^T\omega||_2}{||\lambda^R R\lambda^\omega\omega||_2} = \frac{||REP^T\omega||_2}{||\lambda^R R\lambda^\omega\omega||_2}$ is trivially bounded by $\frac{|R(x_a,q)P(C_i,q)\mu(C_i,a,q)|}{|R(x_a,q)\mu(C_i,a,q)|} \leq |P(C_i, q)| \leq 1$ for $del(x_a, C_i)$. For $repl(x_a, C_i, C_j)$ it requires some work. Note that we will have $||REP^T\omega||_2 = \sqrt{R(x_a)^2 P(C_i)^2 \omega_i^2} \leq \sqrt{R(x_a)^2\omega_i^2 + R(x_a)^2\omega_j^2} = ||\lambda^R R\lambda^\omega\omega||_2$ for $repl(x_a, C_i, C_j)$. Therefore, $\frac{||REP^T\omega||_2}{||\lambda^R R\lambda^\omega\omega||_2} \leq 1$. The second term, $\frac{||R\tilde{Z}P^T\omega - R\tilde{Z}P^T\tilde{\omega}||_2}{||\lambda^R R\lambda^\omega\omega||_2}$ is bounded as follows. One should note that $||\lambda^R R\lambda^\omega\omega||_2 \geq \frac{1}{\sqrt{2}}||\lambda^R R\lambda^\omega||_F||\tau||_2||\omega||_2$ where τ is the smallest possible cluster-ranking value (i.e. for a cluster having one page without no links). Therefore, we have

$$
\frac{||R\tilde{Z}P^T\omega - R\tilde{Z}P^T\tilde{\omega}||_2}{||\lambda^R R\lambda^\omega\omega||_2} \leq \sqrt{2}\frac{||R\tilde{Z}P^T||_F||\omega - \tilde{\omega}||_2}{||\lambda^R R\lambda^\omega||_F||\tau||_2||\omega||_2}
$$

But, one can observe that

$$
\begin{aligned}
& \frac{||R\tilde{Z}P^T||_F}{||\lambda^R R\lambda^\omega||_F} \leq \frac{\sqrt{\sum_{x_i}\sum_{x_i \in C_j} R(x_i, q)^2 P(C_j, x, q)^2}}{\sqrt{R(x_a, q)^2}} \\
\leq & \sqrt{\frac{\sum_{x_i} R^2(x_i, q)\sum_{x_i \in C_j} 1}{R^2(x_a, q)}} \leq \sqrt{\frac{\sum_{x_i} R^2(x_i, q)m}{R^2(x_a, q)}} \leq \sqrt{2m}
\end{aligned}
$$

Moreover, we have $\frac{||\omega - \tilde{\omega}||_2}{||\tau||_2||\omega||_2} \leq \frac{1}{\tau}(1 + \frac{||\tilde{\omega}||_2}{||\omega||_2}) \leq \frac{2}{\tau}$. Therefore, $\frac{||R\tilde{Z}P^T\omega - R\tilde{Z}P^T\tilde{\omega}||_2}{||\lambda^R R\lambda^\omega\omega||_2} \leq \frac{2\sqrt{2m}}{\tau}$.

Once that we have proved there is a constant bound for deletion and replication, it is easy to generalize the constant bound as the set of page movements are combinations of replications and deletions.

[4] Since C_3 contains x_0, it is not true that $P(C_3, q) = 0$ but when n is sufficiently large $P(C_3, q) \approx 0$. Therefore, we assume that $P(C_3, q) = 0$ for the sake of simplicity