IEEE International Conference on Data Engineering

ApproxRank: Estimating rank for a subgraph

Yao Wu [1], Louiqa Raschid [2]

University of Maryland Institute for Advanced Computer Studies (UMIACS)
University of Maryland, College Park
College Park, Maryland, 20742, U.S.A.
[1]yaowu@cs.umd.edu
[2]louiqa@umiacs.umd.edu

***Abstract*— Customized semantic query answering, personalized search, focused crawlers and localized search engines frequently focus on ranking the pages contained within a subgraph of the global Web graph. The challenge for these applications is to compute PageRank-style scores efficiently on the subgraph, i.e., the ranking must reflect the global link structure of the Web graph but it must do so without paying the high overhead associated with a global computation. We propose a framework of an exact solution and an approximate solution for computing ranking on a subgraph. The IdealRank algorithm is an exact solution with the assumption that the scores of external pages are known. We prove that the IdealRank scores for pages in the subgraph converge. Since the PageRank-style scores of external pages may not typically be available, we propose the ApproxRank algorithm to estimate scores for the subgraph. Both IdealRank and ApproxRank represent the set of external pages with an external node Λ and extend the subgraph with links to Λ. They also modify the PageRank-style transition matrix with respect to Λ. We analyze the L_1 distance between IdealRank scores and ApproxRank scores of the subgraph and show that it is within a constant factor of the L_1 distance of the external pages (e.g., the true PageRank scores and uniform scores assumed by ApproxRank). We compare ApproxRank and a stochastic complementation approach (SC) [1], a current best solution for this problem, on different types of subgraphs. ApproxRank has similar or superior performance to SC and typically improves on the runtime performance of SC by an order of magnitude or better. We demonstrate that ApproxRank provides a good approximation to PageRank for a variety of subgraphs.**

I. Introduction

The explosion of information available on the Web has made the ranking of web pages an expensive but unavoidable component of query answering. Since hyperlinks from one page to another usually implies an "endorsement" or "recommendation", link analysis plays a critical role in determining the importance of web pages. PageRank[2] and HITS[3] are two seminal approaches in the area. PageRank iteratively computes the score of a page based on the scores of its parent pages. HITS separates the role of each web page into a *hub* or *authority*. The hub score estimates the value of its links to other pages and the authority score estimates the importance of the page. These algorithms are expensive because of the number of web pages/objects involved in the computation.

In January 2005, the indexable Web for search engines was estimated to be more than 11.5 billion pages [4]. According to [5], the Web is growing at a rate of 25% per year. To make ranking manageable, and to reflect the diversity of clients' information needs, web applications such as semantic search, focused crawlers, localized search engines, and personalized search have emerged. They all have a common objective to rank a subgraph.

The first intriguing application is a *focused crawler* [6], [7], also called a thematic crawler. A focused crawler is interested in collecting a subset of the Web pages that are related to a specific topic. Compared to a standard crawler which can easily get lost and waste resources, a focused crawler acquires relevant pages using a Best First Search; it selects links based on their scores [7]. In contrast to focused crawlers which are topic specific, a *localized search engine* indexes a subset of web pages that are within a specific domain. The web fragment retrieved by the focused crawler (or localized search engine) is a subgraph of the global web graph. Only PageRank scores for local pages in the subgraph are of interest to users. Figure 1 shows the typical infrastructure of a focused crawler (or a localized search engine). Users submit queries to the subgraph collected by a focused crawler and local query answers are returned to the user. The ranking on this local graph, however, should reflect the link structure of all web pages.

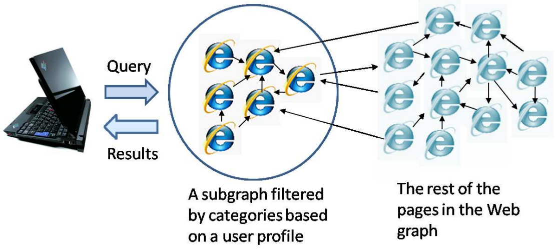
Query
Results
A subgraph filtered by categories based on a user profile
The rest of the pages in the Web graph

Fig. 1. The infrastructure of a focused crawler or a localized search engine.

Another interesting scenario is semantic ranking. ObjectRank [8] creates a schema graph to model the semantic connections between entity sets, e.g., authors or conferences. The semantic connections are associated with an authority transfer assignment which can be arbitrarily set by a domain expert based on her interpretation of the domain. Figure 2 [8] shows an authority transfer schema graph for DBLP.

While ObjectRank is flexible and allows the tuning of ObjectRank scores by a domain expert, it leads to computational challenges if a search engine has to consider all possible combinations of keywords and authority transfer assignments.

1084-4627/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDE.2009.108

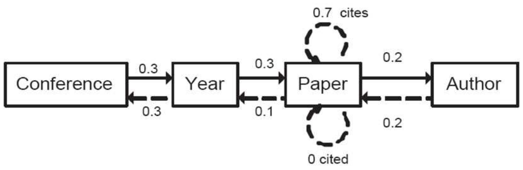

Fig. 2. The DBLP authority transfer schema graph in ObjectRank ([8]).

Recent research on reformulating ObjectRank scores based on individual user feedback [9] and a graph exploration framework for the biological Web [10] highlights the optimization challenges of query answering and ranking for the semantic Web.

If we can model a subgraph to contain the subset of pages associated with the entity sets of interest to some domain expert, we can then define the ObjectRank problem as a problem of ranking a subgraph. This problem, too, is to exploit existing PageRank scores for other regions of the graph that are not of interest to the domain expert, and whose scores may also remain largely unchanged. Figure 3 shows an example where a subgraph is associated with an authority transfer assignment and the external pages are beyond the focus of the domain expert.

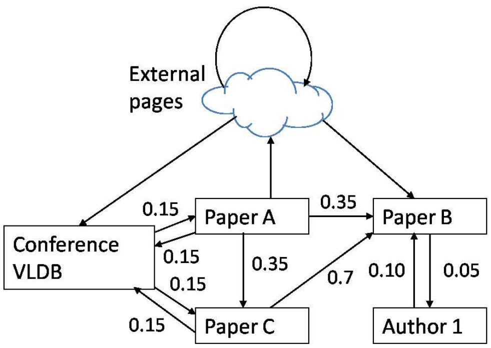

Fig. 3. An example of subgraph semantic ranking.

Another application that involves ranking a subgraph is peer-to-peer networks. The advent of peer-to-peer(P2P) technology has further boosted web information retrieval by leveraging distributed computing power, storage, and connectivity[11], [12], [13]. A distributed or decentralized system has multiple peers or servers, each of which stores its own subgraph of the Web. A user may ask queries on one peer and ranked query answers that are available locally are presented to the user. The ranking depends on the context of the query.

A final scenario is a reflection of the constant change of the Web. The ranking of pages needs to be updated frequently, especially for the subgraph of the Web that experiences the most change. This subgraph can be either a set of dangling pages that crawlers have not as yet crawled, referred to as the web ”frontier” [14], or the set of pages that are most affected by updates [15]. It is desirable that any strategy to update the ranking of this subgraph exploits existing PageRank scores for other regions of the graph which may remain largely unchanged.

In response to these many motivating applications, we address the problem of computing ranking scores for a subgraph. For ease of presentation, and to compare with existing approaches, we use the PageRank metric for explanation and experiments. However, our general approaches can be applied to estimate ObjectRank scores as well.

We note that current ranking techniques (to be discussed in the next section) either pay the cost of a global computation to get an accurate ranking [16], or they have to solve another potentially difficult problem: to determine a *relevant supergraph of web pages* that impact the rank of the subgraph [1], [17]. Our challenge is to obtain an accurate ranking that reflects the global link structure of the Web graph and to do so without paying the high overhead associated with a global PageRank computation or having to solve the difficult problem of identifying a relevant supergraph. We would also like to exploit pre-computed PageRank scores for external pages if and when they are available and appropriate for use.

We propose a framework based on an exact and an approximate solution to compute PageRank on a subgraph. The IdealRank algorithm is an exact solution. It assumes that the PageRank scores of external pages are known. We prove that the IdealRank scores for pages in the subgraph converge to the true PageRank scores. Since the PageRank scores of external pages may not typically be available, we propose the ApproxRank algorithm to estimate PageRank scores for the subgraph. Both IdealRank and ApproxRank represent the set of external pages with an external node Λ and extend the subgraph with links to Λ. They also modify the PageRank transition matrix with respect to (the links to) Λ.

The IdealRank and ApproxRank framework formalizes the problem of ranking a subgraph. It allows us to model multiple scenarios where ranking a subgraph is important. IdealRank can be used to model scenarios where PageRank scores of the global graph are known a priori and can potentially be re-used. This includes the case where the subgraph represents the pages that have been updated, or the subgraph represents the pages that contain all the semantic types of interest to a domain expert for a personalized or semantic ranking such as ObjectRank [8]. ApproxRank can be applied in general to all these problems, when we do not know the PageRank scores of external pages.

We compare our approach with the stochastic complementation (SC) approach [1]. SC builds a supergraph by carefully examining candidate external pages and adding them into the supergraph if adding this page has a significant influence on the PageRank scores of the subgraph. Our approach in contrast models the external pages using a node Λ, and it can be used in situations when a supergraph cannot be obtained easily. Our approach also avoids the cost of a global computation [16]. The ApproxRank computation is also much cheaper than SC [1] since SC pays the high cost of constructing the supergraph.

We experimentally study the effect of size and type of the subgraphs on the accuracy of ApproxRank. We study several types of subgraphs including domain specific subgraphs, topic specific subgraphs, and subgraphs gathered by a Breadth First Search crawler. We compare our results with SC and two baseline ranking algorithms; one was discussed in [18], and the other is local PageRank on the subgraph (ignoring the external pages). We show that ApproxRank has similar or superior ranking accuracy to SC and typically its runtime performance is an order of magnitude better than SC. ApproxRank also outperforms the two baseline algorithms on ranking accuracy.

Our contributions are as follows:

1) We define an efficient algorithm, IdealRank, to compute PageRank scores for a subgraph when PageRank scores of the external pages are known. The random walk defined by IdealRank utilizes these scores.
2) We prove that the IdealRank scores converge to the true PageRank scores for all local pages in the subgraph, and the IdealRank score for the external node Λ converges to the sum of true PageRank scores for all external pages.
3) When PageRank scores of external pages are not known, we define an efficient algorithm ApproxRank. We provide important properties of ApproxRank scores.
4) We show through empirical results that the ApproxRank ranking accuracy is similar (sometimes superior) to the best competitor SC, and it overwhelmingly outperforms the runtime efficiency of SC.

The rest of the paper is organized as follows. Section II briefly reviews the PageRank algorithm and discusses related work. Section III defines the IdealRank algorithm. Section IV presents the ApproxRank algorithm. Experimental results are described in V and VI concludes.

II. Related Work

We briefly describe the PageRank algorithm and summarize research on PageRank. We refer to surveys [19], [20] and book [21] for a more complete description of related work.

A. PageRank Review

PageRank was introduced in [2] to compute the stationary distribution of a Markov chain from the link structure of a web graph. The underlying assumption is that links between pages confer authority. A link from page i to page j is evidence that i is suggesting that j is important. The importance contributed to page j from i is inversely proportional to the outdegree of i. Let D_i be the outdegree of page i. The corresponding random walk on the directed web graph can be expressed by a transition matrix A as follows:

$$A[i,j] = \begin{cases} \frac{1}{D_i} & \text{if there is an edge from } i \text{ to } j, \\ 0 & \text{otherwise.} \end{cases}$$

Let R be the PageRank vector to be computed over the web pages. Initially R can be an arbitrary vector representing the probability of visiting web pages. Let *damping factor* ϵ be the probability that a web surfer follows the hyperlinks and let $(1-\epsilon)$ be the probability of a surfer making a random jump to a page, where ϵ is usually set to be 0.85. The *personalization vector* P can be used to bias PageRank to prefer certain pages. In standard PageRank, P is a uniform distribution to indicate the equi-probability of randomly jumping to any page; it is as follows:

$$P = [\frac{1}{n}]_{n\times 1}$$

Let A^T be the transpose of A. The PageRank vector R is recursively defined as follows:

$$R = \epsilon A^T \cdot R + (1-\epsilon)P$$

According to the Ergodic Theorem [22], [23] for Markov chains, if the graph is aperiodic and irreducible, i.e., the Web graph is strongly connected, then a unique steady state distribution exists. Since the Web graph is generally aperiodic and irreducible by adding damping factor, R converges to the stationary distribution for the Web graph.

B. Efficiently Computing PageRank

The efficient computation of the PageRank algorithm has been studied in [22], [24], [25], [26], [27]. An adaptive method is exploited in [26] where pages whose scores have converged are not recomputed in a new iteration. An extrapolation method is proposed in [22] so the higher terms in ranking vector expansion are suppressed. [27] presents a 3-stage algorithm to speed up the PageRank computation. The first step is to compute local PageRank scores for each host. Then a block graph is constructed in the second step, where every node represents a block and every edge represents a set of hyperlinks from a block to another block (or itself). The importance of hosts is computed on this block graph. Finally the standard PageRank is run on the global graph using as its starting vector the weighted aggregation of the local PageRank score. In [24], [25], other graph aggregation approaches are presented to approximate the PageRank computation.

C. Computing PageRank in a distributed system

Recent research efforts in distributed systems have addressed the case where the Web graph is partitioned into disjoint web sites or domains [18], [28]. In [18], the Web is modeled as numerous disjoint web servers. The hyperlinks in the Web are divided into two categories, *intra-sever* links and *inter-server* links. Intra-server links are links between pages within a server and these links are used to compute a local PageRank vector on each server. Inter-server links are links between pages in different servers, and they are used to compute ServerRank. ServerRank measures the relative importance of the different web servers. Finally results from multiple web servers are merged to generate a ranked hyperlink list on the submitting server. In [28], a ranking algebra is proposed to deal with rankings at different granularity levels, which can also be applied to aggregate local ranking and site ranking to get global ranking.

There has been a work [16] on PageRank approximation in a fully decentralized system, where each peer is autonomous and peers may overlap with each other. In the proposed JXP algorithm, each peer computes the local PageRank scores, then randomly meets other peers and gradually increases its knowledge about the global web graph by exchanging information, and then recomputes the PageRank scores on local peer. This meeting and recompute process is repeated until the peer gathers enough information. The JXP scores converge to the true global PageRank scores if peers eventually meet sufficient number of times to exchange information. The assumption is that the outdegree of each page in the global graph is known.

However, these work focus on providing an approximation for the global graph, in centralized systems or distributed systems.

D. Computing PageRank for a subgraph

The problem of estimating PageRank values for a small portion of the Web graph has received recent attention in the literature [1], [17], [29]. The goal of [17] is to estimate the PageRank value for one target node. [1] addresses the problem of estimating the score for a subgraph. [29] aims to do link based ranking on a small graph exploiting users' access patterns.

The common approach in all of these papers is to expand the subgraph to a supergraph and then run PageRank on this augmented graph. They differ in the procedure to augment the subgraph. The expansion in [17] proceeds backwards by following reverse hyperlinks. PageRank scores for boundary nodes in the augmented graph that have incoming edges from outside of the subgraph are estimated. Then standard PageRank is computed on this graph. [1] starts with following the outgoing links in a given local graph of size n. A set of k nodes are selected via stochastic complementation and the PageRank of this expanded graph is computed. This process is repeated for a number of iterations (e.g. 25 iterations). In [29], besides the existing hyperlinks that indicate recommendation, implicit recommendation links that can be determined by mining user access patterns are also added into the supergraph.

We compare our approach with SC in [1] which is also a solution to rank a subgraph. They introduce a new challenge to construct a good supergraph and that has some disadvantages. By following outgoing links it is possible that their approach could sometimes miss important pages that link into the subgraph. To decide if each candidate page should be included in the supergraph, SC has to estimate PageRank scores on the subgraph when added the candidate page. When the size of the constructed supergraph is large, SC paid the overhead of computing PageRank scores for pages that are not relevant to the user. They are also not able to utilize the PageRank scores of pages when they are known a priori and are not expected to change significantly. This corresponds to the scenario of customizing or personalizing the rank of a subgraph using a metric such as ObjectRank, or the scenario where the updates to the Web graph are confined to the subgraph. We take a different approach through state aggregation to reduce computation in each iteration. We will compare our approaches in Section V.

E. State Aggregation Approaches

Iterative Aggregation/Disaggregation (IAD) method [30] is applied to PageRank in [15] to update PageRank scores when the transition probabilities or states are changed. As discussed in Section II-B, [24] presents a graph aggregation method to approximate PageRank for the entire graph. In contrast to these aggregation approaches, the IdealRank and ApproxRank framework is more general since it addresses aggregation for a subgraph both when the PageRank scores are known and when they are not known a priori.

III. IdealRank Approach

We formally define the IdealRank algorithm to compute PageRank scores for a local graph. Our approach is inspired by research on collapsing matrices with the same eigenvector [31]. IdealRank performs a random walk on a modified local graph called the *extended local graph*, where an external node Λ is added to the local graph. Λ represents the set of pages that are not local. The transition matrix probabilities of IdealRank are derived from the transition matrix of PageRank for the global graph. IdealRank assumes that the PageRank scores of all external pages in Λ are known. This assumption will be relaxed in the next section where we present an approximate solution.

Consider two graphs; a global graph of size N, and a local graph of size n. The local graph is a subgraph of the global graph. The pages in the local graph are called *local pages* while pages in the global graph and that are not in the local graph are called *external pages*. The goal is to provide the true PageRank for the local graph without running PageRank on the global graph.

Table I lists the symbols used to define our algorithms.

Symbol	Meaning
Λ	External node, the artificial node representing all external pages.
G_l	A subgraph of the Web with n pages
G_g	The global Web graph with N pages.
G_e	The extended local graph with $n+1$ pages.

TABLE I

Symbols used by algorithms

A. The IdealRank algorithm

Recall that in [16], [18], an artificial node represents the external world. There are edges between the artificial node and local nodes based on the global Web graph. However, this solution cannot distinguish between the case of one link or multiple links between a local page and the external pages as seen in the following example:

Let Figure 4 be a global graph. Node A,B,C, and D are local pages, and node X, Y and Z in the cloud are external pages. Figure 5 provides an example of adding an artificial external node to represent the external pages. Edges are added from local pages to the external node without a strategy to modify the original PageRank transition matrix to reflect that each such edge may represent multiple edges in the global graph. When computing the standard Pagerank algorithm on this graph, the probability flow from a page is proportional to the inverse of its outdegree. Page C which has 3 incoming edges from the external pages is treated similarly to page D which has only 1 incoming edge from the external pages. Intuitively, however, we should expect a higher probability of following links from the external pages to page C. Similarly, the probability of following links from page A to Λ is $1/3$. This too is lower than the transition probability based on the global graph.

IdealRank addresses this shortcoming with the following solution: The first step is to add an external node Λ to the subgraph to represent all external pages. The second step is to construct the *extended local graph* G_e, the Λ enriched graph of size $n+1$. There is an edge from Λ to a local page in G_e if there is an edge from an external page to that local page. The same hold for edges out of local pages. Similarly, there is an edge from Λ to Λ if there is an edge between external pages. The next step is to define a transition matrix A_{ideal} and a personalization vector P_{ideal}. The details will be discussed in Section III-B. Finally, a random walk is performed on G_e. The IdealRank vector R_{ideal} is defined as follows:

$$R_{ideal} = \epsilon A_{ideal}^T \cdot R_{ideal} + (1-\epsilon) P_{ideal} \tag{1}$$

```
Algorithm IdealRank($G_l$, $G_g$)
  1. Add external node $\Lambda$ to $G_l$.
  2. Create edges associated with $\Lambda$ and get $G_e$.
  3. Assign values to $P_{ideal}$ and $A_{ideal}$.
  4. Perform a random walk on the extended local graph
     according to Formula (1).
```

B. A_{ideal} and P_{ideal}

We define an $(n+1)\times(n+1)$ transition matrix A_{ideal} and a length $(n+1)$ personalization vector P_{ideal}. Let A represent the $N \times N$ transition matrix for PageRank on the global graph. Entry $A_{i,j}$ has the value of the inverse outdegree of page i, if there is an edge (i,j); the value is the probability of a random surfer following this edge from i. Without loss of generality, we consider the local pages to be the first contiguous n pages in A and the external pages are indexed from $n+1$ to N in A.

Assume that the PageRank scores for all external pages are known. The values are $\{R[n+1], R[n+2], \cdots, R[N]\}$, respectively. Let $EXTSum = \sum_{i=n+1}^{N} R[i]$. A_{ideal} is defined as follows, based on the entries in the original PageRank transition matrix A:

$$\left(\begin{array}{ccc|c} A_{1,1} & \cdots & A_{1,n} & \sum\limits_{i=n+1}^{N} A_{1,i} \\ \vdots & & \vdots & \vdots \\ A_{n,1} & \cdots & A_{n,n} & \sum\limits_{i=n+1}^{N} A_{n,i} \\ \hline \frac{\sum\limits_{j=n+1}^{N} R[j]A_{j,1}}{EXTSum} & \cdots & \frac{\sum\limits_{j=n+1}^{N} R[j]A_{j,n}}{EXTSum} & \frac{\sum\limits_{j=n+1}^{N} R[j] \sum\limits_{i=n+1}^{N} A_{j,i}}{EXTSum} \end{array}\right)$$

Next we explain the elements in A_{ideal}. These values are as follows:

1) The $n \times n$ submatrix at upper left is identical to the corresponding elements in transition matrix A for the global graph. They represent the probability of transition between edges in the local graph.
2) The $n \times 1$ submatrix at upper right represents the probability flow from a local page to the node Λ. We note that the probability of reaching Λ is the sum of the probability of reaching any external page from the local page. For local page k, the value is $\sum_{i=n+1}^{N} A_{k,i}$.
3) The $1 \times n$ submatrix at lower left corresponds to the probability flow from Λ to local pages. For local page k, the value is $\frac{\sum_{j=n+1}^{N} R[j]A_{j,k}}{EXTSum}$.
4) The entry at the lower right corner denotes the probability flow from Λ to Λ.

The last row has entries that are each a weighted sum of probabilities summed over all external pages. The weight is determined by the PageRank score of the external page. This is a key feature of A_{ideal} and will be discussed next.

We define A_{ideal} formally as follows: $A_{ideal} = Q_1 A Q_2$, where Q_1 is an $(n+1)\times N$ matrix and Q_2 is an $N \times (n+1)$ matrix. Let Q_2 be an $N \times (n+1)$ matrix as follows:

$$\begin{pmatrix} I_n & B \\ C & D \end{pmatrix} \tag{2}$$

where I_n is an $n \times n$ identity matrix, B is an $n \times 1$ 0-matrix, C is a $(N-n)\times n$ 0-matrix, and D is a $(N-n)\times 1$ matrix with all 1's. The effect of AQ_2 on the ranking vector is to aggregate the authority flow from local pages to all external pages, which indicates the authority goes to Λ.

Let Q_1 be the following $(n+1)\times N$ matrix:

$$\begin{pmatrix} I_n & C^T \\ B^T & E \end{pmatrix} \tag{3}$$

where I_n is an $n \times n$ identity matrix, C^T is an $n \times (N-n)$ 0-matrix and B^T is a $1 \times n$ 0-matrix.

The matrix of interest is E, a $1 \times (N-n)$ matrix. It considers the PageRank scores for all external pages. Recall that $EXTSum$ is the sum of PageRank scores for all external pages, $EXTSum = \sum_{i=n+1}^{N} R[i]$. Then, E can be expressed as follows:

$$E = \left(\frac{R[n+1]}{EXTSum}, \frac{R[n+2]}{EXTSum}, \cdots, \frac{R[N]}{EXTSum} \right) \tag{4}$$

The idea of multiplying the values of entries in A with the two matrices Q_1 and Q_2, where Q_1 derived from the ranking vector for external pages, is *key* to the approach of A_{ideal}. It has the effect of distributing the probability flow from the external nodes, in a manner that is proportional to the importance of each of the external pages in the original PageRank vector.

Recall that the personalization vector in the original PageRank is defined as a uniform vector $P = [\frac{1}{n}]_{n\times 1}$. Instead, for IdealRank we define the personalization vector P_{ideal} according to the number of external pages and total number of pages in the graph. More specifically, the i-th entry of P_{ideal}, $P_{ideal}[i]$ can be expressed as follows:

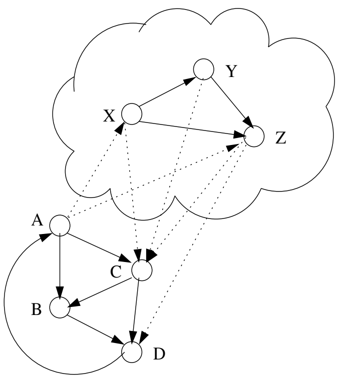

Fig. 4. A global graph of both local pages and external pages.

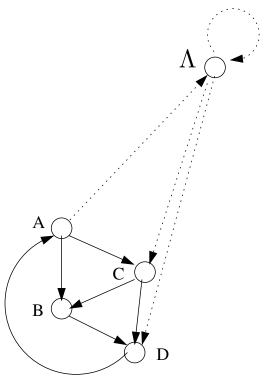

Fig. 5. An extended local graph without a strategy to adjust transition probabilities

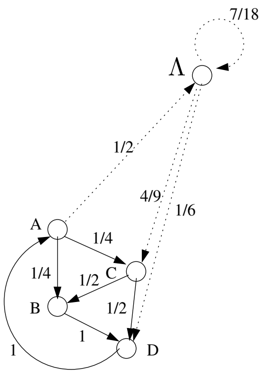

Fig. 6. An extended local graph marked with transition probabilities in ApproxRank

$$P_{ideal}[i] = \begin{cases} \frac{1}{N} & \text{if page } i \text{ is local,} \\ \frac{N-n}{N} & \text{if page } i \text{ is the external node } \Lambda. \end{cases} \quad (5)$$

C. Convergence of IdealRank

Let R_{ideal} be the final ranking vector of *IdealRank*, where the first n elements are scores for local pages and the $(n+1)$-th element is the score for the external node Λ. We show that the scores of first n elements are identical to the true PageRank scores.

Theorem 1: In R_{ideal}, scores for the first n pages converge to the true PageRank scores. The score for the $(n+1)$th element, Λ, converges to the sum of true PageRank scores for all external pages.

Proof: Let R be the true PageRank vector such that $R = \epsilon A^T \cdot R + (1-\epsilon)P$, i.e., R is the converged stationary distribution for A. Let $R' = Q_2^T R$ be a vector with $n+1$ entries. We also know that $R = Q_1^T R'$. It is obvious that $R'[i] = R[i]$ for first n elements and $R'[n+1] = \sum_{i=n+1}^{N} R[i]$. We will show that R' is the IdealRank vector.

We know that $\epsilon A^T R + (1-\epsilon)P = R$. Next consider a left multiply with Q_2^T to obtain the following:

$$\begin{array}{lcll} \epsilon A^T R + (1-\epsilon)P & = & R & \Rightarrow \\ Q_2^T \epsilon A^T R + Q_2^T (1-\epsilon)P & = & Q_2^T R & \Rightarrow \\ \epsilon Q_2^T A^T Q_1^T R' + (1-\epsilon) Q_2^T P & = & Q_2^T R & \Rightarrow \\ \epsilon (Q_1 A Q_2)^T R' + (1-\epsilon) P_{ideal} & = & R' & \Rightarrow \\ \epsilon A_{ideal} R' + (1-\epsilon) P_{ideal} & = & R' & \end{array}$$

Since A_{ideal} is stochastic and Markov Chain defined by IdealRank is irreducible and aperiodic, there is a unique stationary distribution for A_{ideal}. Therefore, $R' = R_{ideal}$. ■

The IdealRank algorithm addresses several applications. One is where some subgraph of the Web graph has been updated. A second case is when the personalized authority transfer is limited to the subgraph. In these cases, the knowledge of PageRank scores can be potentially relied on to estimate new ranking scores.

IV. The ApproxRank algorithm

Unlike the previous scenario where PageRank values for external pages are known, we now consider scenarios where the PageRank scores are not known a priori. To cover this situation, our framework has an approximate solution ApproxRank. The key difference is that for ApproxRank, the algorithm is not able to differentiate the (previously weighted) contribution of authority from each individual external page (since these PageRank scores are unknown). Instead, ApproxRank will consider the authority flow from external pages assuming they are equally important. We analyze the L_1 distance between IdealRank scores and ApproxRank scores of the subgraph and reveal that it is within a constant factor of L_1 distance between the true PageRank scores and uniform scores of the external pages. We will show through experiments that ApproxRank is a good approximation.

A. The ApproxRank algorithm

The ApproxRank vector R_{approx} is defined as follows:

$$R_{approx} = \epsilon A_{approx}^T \cdot R_{approx} + (1-\epsilon) P_{ideal} \quad (6)$$

ApproxRank adopts the same personalization vector as IdealRank. It however, defines its own transition matrix A_{approx}.

B. A_{approx} definition

A_{approx} is an $(n+1) \times (n+1)$ matrix. It is defined as follows:

$$\left(\begin{array}{ccc|c} A_{1,1} & \cdots & A_{1,n} & \sum\limits_{i=n+1}^{N} A_{1,i} \\ \vdots & & \vdots & \vdots \\ A_{n,1} & \cdots & A_{n,n} & \sum\limits_{i=n+1}^{N} A_{n,i} \\ \hline \frac{\sum\limits_{j=n+1}^{N} A_{j,1}}{N-n} & \cdots & \frac{\sum\limits_{j=n+1}^{N} A_{j,n}}{N-n} & \frac{\sum\limits_{i=n+1}^{N} \sum\limits_{j=n+1}^{N} A_{i,j}}{N-n} \end{array} \right)$$

A_{approx} is different from A_{ideal} in the last row (see Section III-B), since IdealRank does not utilize knowledge about PageRank scores of external pages in the first n rows.

For the first n entries in the last row, the value represents the (average) probability flow accumulated from $(N-n)$ external pages to each local page. The last entry in this n-th row of the matrix is the (average) probability flow from external pages to other external pages. Similar to $A_{ideal} = Q_1 A Q_2$, A_{approx} can be formally defined as $A_{approx} = Q_1' A Q_2$, where the vector E is replaced by a vector E_{approx} in Q_1':

$$E_{approx} = \left(\ \frac{1}{N-n}, \frac{1}{N-n}, \cdots, \frac{1}{N-n} \ \right) \tag{7}$$

In A_{approx}, the values at the last row are as follows:

1) For the first n values, $(1 <= k <= n)$, the probability from Λ to a local page k is assigned the summation of flow from all external pages to k, divided by the number of external pages. For local page k, it is $\frac{\sum_{j=n+1}^{N} A_{j,k}}{N-n}$.
2) For the $(n+1)$-th value, the probability for the self-loop edge is determined by the total authority flow among external pages, divided by the number of external pages.

Given the global graph example in Figure 4, the probabilities assigned by A_{approx} are shown in Figure 6. We provide some examples of edge weight calculation following these rules. According to rule 1, the authority flow on edge AB, AC, CB, BD, CD, DA are the outdegree inverse. Since A points to page X, Z, the authority flow on edge (A, Λ) is $1/2$. The authority flow on edge (Λ, C), $\frac{\frac{1}{D_X}+\frac{1}{D_Y}+\frac{1}{D_Z}}{3} = \frac{\frac{1}{3}+\frac{1}{2}+\frac{1}{2}}{3} = \frac{4}{9}$. The self-loop edge authority flow will be $\frac{\frac{2}{D_X}+\frac{1}{D_Y}}{3} = \frac{\frac{2}{3}+\frac{1}{2}}{3} = \frac{7}{18}$.

An advantageous quality about ApproxRank is that it is suitable to adopt precomputation for various subgraphs. With the same global graph, A_{approx} can be figured out easily from the difference between the local values and the global values. This is especially beneficial for applications where there are multiple subgraphs.

ApproxRank scores converge to a unique vector R_{approx}. There are two reasons. First, the transition matrix A_{approx}^T is a column stochastic matrix, as the sum of each column is 1. Second, since we complement the random walk with jumps from dangling pages, the Markov Chain we defined is irreducible and aperiodic. ApproxRank satisfies the two conditions of being irreducible and aperiodic of the Ergodic Theorem for Markov chains [22]. Next we will investigate how close is R_{approx} to R_{ideal}, which we have shown to be the true PageRank scores for local pages.

C. Error analysis of ApproxRank ranking vector R_{approx}

In this section we provide important properties of ApproxRank scores through iterations. We show that the L_1 distance between IdealRank scores and ApproxRank scores of the subgraph is within a constant factor of L_1 distance between the true PageRank scores and assumed scores of the external pages. This relationship can be utilized to improve ApproxRank algorithm, which will be our future work. Our experiments show that, however, even assume that the external pages are equally important, ApproxRank behaves well and produces comparable results to existing approach. To our best knowledge, similar analysis has not been conducted in previous work for PageRank estimation [1], [17], [18]. There are analysis results of the same flavor through different approaches in the area of stable analysis of PageRank [32] and in the area of updating PageRank scores [33].

Let R_{ideal} and R_{approx} be the ranking vectors from IdealRank and ApproxRank respectively, each with length $n+1$, where the $(n+1)th$ elements in vectors are scores for the external node Λ. We abuse notations and let R_{ideal} and R_{approx} be the subvector of the first n elements, as we are interested in accuracy of ApproxRank for the n local pages. Let R_{ideal}^m and R_{approx}^m be the ranking vectors after the m-th iteration from IdealRank and ApproxRank.

Let E and E_{approx} in Equation (4) and (7) be the vector used to define A_{ideal} and A_{approx}. Both E and E_{approx} are vectors of length $N-n$, where each element denotes the relative importance of $N-n$ external pages. We note that we index these elements with $n+1, \cdots, N$ to reference the scores for the corresponding external pages.

Theorem 2 states that after m iterations,

$$\parallel R_{ideal}^m - R_{approx}^m \parallel_1 \leq (\epsilon^m + \epsilon^{m-1} + \cdots + \epsilon) \parallel E - E_{approx} \parallel_1$$

When the number of iterations goes to infinity, this becomes

$$\parallel R_{ideal}^\infty - R_{approx}^\infty \parallel_1 \leq \frac{\epsilon}{1-\epsilon} \parallel E - E_{approx} \parallel_1$$

This shows that the accuracy of ApproxRank is dependent on the knowledge of relative importance of external pages. When ϵ is set to be 0.85, which is usually the case, the error of ApproxRank is bounded by a constant factor of 5.67 of the error of E_{approx}.

We first derive the base case and recurrence relation for the L_1 distance between ApproxRank ranking vector and IdealRank ranking vector. Then an error bound are obtained based on a priori error of the external pages in Theorem 2.

Lemma 1: After the first iteration, the ApproxRank ranking vector R_{approx}^1 satisfies:

$$\parallel R_{ideal}^1 - R_{approx}^1 \parallel_1 \leq \epsilon \parallel E - E_{approx} \parallel_1$$

Proof:

$$\begin{aligned}
& \parallel R_{ideal}^1 - R_{approx}^1 \parallel_1 \\
= \ & \textstyle\sum_{k=1}^{n} |R_{ideal}^1[k] - R_{approx}^1[k]| \\
= \ & \textstyle\sum_{k=1}^{n} |\epsilon \sum_{i=1}^{n} A_{ik} \cdot 1 + \epsilon \sum_{j=n+1}^{N} A_{jk} E[j] + (1-\epsilon)\frac{1}{N} \\
& \textstyle -\epsilon \sum_{i=1}^{n} A_{ik} \cdot 1 - \epsilon \sum_{j=n+1}^{N} A_{jk} E_{approx}[j] - (1-\epsilon)\frac{1}{N}| \\
= \ & \textstyle\epsilon \sum_{k=1}^{n} | \sum_{j=n+1}^{N} A_{jk}(E[j] - E_{approx}[j])| \\
\leq \ & \textstyle\epsilon \sum_{k=1}^{n} \sum_{j=n+1}^{N} A_{jk} |E[j] - E_{approx}[j]| \\
\leq \ & \textstyle\epsilon \sum_{j=n+1}^{N} \sum_{k=1}^{n} A_{jk} |E[j] - E_{approx}[j]| \\
\leq \ & \textstyle\epsilon \sum_{j=n+1}^{N} |E[j] - E_{approx}[j]| \\
\leq \ & \epsilon \parallel E - E_{approx} \parallel_1
\end{aligned}$$

To derive the inequality, we first express the L_1 distance based on its definition, then calculate $R_{ideal}^1[k]$ and $R_{approx}^1[k]$ assuming that the initial vectors for IdealRank and ApproxRank are the same (e.g. 1) for local pages. Because transition matrix A is row stochastic, $\sum_{k=1}^{n} A_{jk} \leq 1$. The definition of L_1 distance concludes the proof. ■

Next we explore a recurrence relation for the L_1 distance between R_{approx} and R_{ideal} after m iterations. Lemma 2 shows that after each iteration, the L_1 distance deviate not too much.

Lemma 2: After an arbitrary positive integer $m > 1$ iterations, we have the following recurrence relation:

$$\begin{aligned} \| R_{ideal}^{m} - R_{approx}^{m} \|_1 \quad \leq \quad & \epsilon \| R_{ideal}^{m-1} - R_{approx}^{m-1} \|_1 + \\ & \epsilon \| E - E_{approx} \|_1 \end{aligned}$$

Proof: Albeit more terms involved, the proof follows the same vein with the base case in Lemma 1.

$$\begin{aligned} & \| R_{ideal}^{m} - R_{approx}^{m} \|_1 \\ = \ & \textstyle\sum_{k=1}^{n} |R_{ideal}^{m}[k] - R_{approx}^{m}[k]| \\ = \ & \textstyle\sum_{k=1}^{n} |\epsilon \sum_{i=1}^{n} A_{ik} \cdot R_{ideal}^{m-1}[i] + \epsilon \sum_{j=n+1}^{N} A_{jk} E[j] \\ & +(1-\epsilon)\frac{1}{N} - \epsilon \textstyle\sum_{i=1}^{n} A_{ik} \cdot R_{approx}^{m-1}[i] \\ & -\epsilon \textstyle\sum_{j=n+1}^{N} A_{jk} E_{approx}[j] - (1-\epsilon)\frac{1}{N}| \\ = \ & \epsilon \textstyle\sum_{k=1}^{n} | \sum_{i=1}^{n} A_{ik} \cdot (R_{ideal}^{m-1}[i] - R_{approx}^{m-1}[i]) \\ & + \textstyle\sum_{j=n+1}^{N} A_{jk}(E[j] - E_{approx}[j])| \\ \leq \ & \epsilon \textstyle\sum_{k=1}^{n} | \sum_{i=1}^{n} A_{ik} \cdot (R_{ideal}^{m-1}[i] - R_{approx}^{m-1}[i])| \\ & +\epsilon \textstyle\sum_{k=1}^{n} | \sum_{j=n+1}^{N} A_{jk}(E[j] - E_{approx}[j])| \\ \leq \ & \epsilon \textstyle\sum_{k=1}^{n} \sum_{i=1}^{n} A_{ik}|R_{ideal}^{m-1}[i] - R_{approx}^{m-1}[i]| \\ & +\epsilon \textstyle\sum_{k=1}^{n} \sum_{j=n+1}^{N} A_{jk}|E[j] - E_{approx}[j]| \\ \leq \ & \epsilon \textstyle\sum_{i=1}^{n} \sum_{k=1}^{n} A_{ik}|R_{ideal}^{m-1}[i] - R_{approx}^{m-1}[i]| \\ & +\epsilon \textstyle\sum_{j=n+1}^{N} \sum_{k=1}^{n} A_{jk}|E[j] - E_{approx}[j]| \\ \leq \ & \epsilon \textstyle\sum_{i=1}^{n} |R_{ideal}^{m-1}[i] - R_{approx}^{m-1}[i]| \\ & +\epsilon \textstyle\sum_{j=n+1}^{N} |E[j] - E_{approx}[j]| \\ \leq \ & \epsilon \| R_{ideal}^{m-1} - R_{approx}^{m-1} \|_1 + \epsilon \| E - E_{approx} \|_1 \end{aligned}$$

■

Theorem 2:

$$\| R_{ideal}^{m} - R_{approx}^{m} \|_1 \leq (\epsilon^m + \epsilon^{m-1} + \cdots + \epsilon) \| E - E_{approx} \|_1$$

Proof: The proof is straightforward by combining Lemma 1 and Lemma 2.

$$\begin{aligned} & \| R_{ideal}^{m} - R_{approx}^{m} \|_1 \\ \leq \ & \epsilon(\epsilon \| R_{ideal}^{m-2} - R_{approx}^{m-2} \|_1 + \epsilon \| E - E_{approx} \|_1) \\ & +\epsilon \| E - E_{approx} \|_1 \\ \leq \ & \epsilon^{m-1} \| R_{ideal}^{1} - R_{approx}^{1} \|_1 + \\ & (\epsilon^{m-1} + \epsilon^{m-2} + \cdots + \epsilon) \| E - E_{approx} \|_1 \\ \leq \ & (\epsilon^{m} + \epsilon^{m-1} + \cdots + \epsilon) \| E - E_{approx} \|_1 \end{aligned}$$

■

V. Evaluation

An experimental evaluation of both ApproxRank and IdealRank would be fairly extensive since we would have to consider a variety of scenarios where we would be able to apply one or both solutions. Due to space limitations, we limit our experimental evaluation in this paper to ApproxRank. We note that for the scenarios considered here, IdealRank is not applicable since the PageRank scores of the external pages are not known a priori.

A. Experiment Description

To evaluate our approach, we consider two goals in experiments. The first goal is to compare the ApproxRank with the stochastic complementation (SC) approach [1], which is the best existing approach for the problem. The second goal of experiments is to study the effect of size and type of the subgraphs on accuracy of the ApproxRank vector.

Ideally we would run experiments on the whole web graph, which is obviously infeasible. In choosing appropriate datasets, we first surveyed a few recent ranking papers and we list the key characteristics of their datasets in Table II. We will take a similar approach of crawling a relatively small portion of the Web, and let it reflect the whole Web.

Paper	data description	#pages (million)	#links (million)
[1]	"edu": crawl of 100 CS domains "politics": crawl under politics hierarchy	4.7 4.4	22.9 17.3
[34]	web objects including papers, authors etc	1.65	7
[16]	Amazaon.com data Web crawl	0.055 0.103	0.237 1.63
[18]	A breadth first search crawl within domain www.standford.edu	1.05	4.98

TABLE II

Dataset characteristics from recent ranking papers.

We consider the following three types of subgraph in our experiments:

- **TS subgraph**: The first type of subgraph is a topic specific subgraph.
- **DS subgraph**: This type of subgraph is a domain specific subgraph, where each subgraph contains *all* pages from the domain and hyperlinks between local pages within the local domain.
- **BFS subgraph**: This subgraph is constructed by a Breadth First Search (BFS) crawler which starts from a seeded URL. The crawler may follow hyperlinks and fetch Web pages across multiple domains.

For ApproxRank and PageRank implementation, we set the damping factor ϵ to be 0.85. The convergence of the algorithms is identified when the absolute value of the L_1 norm is less than 0.00001. For SC experiments for a subgraph of size n, we use the similar setting in [1] and expand the subgraph for 25 iterations to select another n external pages. The experiments were run on a Solaris machine with 12 GB RAM.

B. Evaluation Method

We compute the PageRank vector for the global graph. This ranking vector for the global graph is then limited to pages in the subgraph, denoted by ranking vector R_1. Let R_2 be the PageRank estimation on the local graph. We evaluate the difference between R_1 and R_2. Without considering the actual scores, these two ranking vectors produce two ranked list σ_1 and σ_2.

We use two ranking metrics in our experiments. The SC approach [1] reported on the L_1 distance. The L_1 distance is the absolute value of the differences between the PageRank estimation and the global PageRank scores, for the subgraph.

$$\| R_1 - R_2 \|_1 = \Sigma_{i=1}^{n} |R_1[i] - R_2[i]|$$

Other research [16], [35] use the Spearman's footrule distance to measure the success of their PageRank approximations. Thus, we also report on the Spearman's footrule distance between the ApproxRank vector σ_2 and the global PageRank vector σ_1.

Note that there may be a substantial number of tied pages with the same score. A ranking with ties is referred to as a *partial ranking*. We consider an extension of the Spearman's footrule distance for ranking with ties [36].

The set of pages in ties is called a *bucket*. Each list σ_1, and σ_2 can be viewed as ranked buckets $B_1, B_2, \cdots, B_t$. The *bucket position* for bucket B_i, $pos(B_i)$, is defined as follows:

$$pos(B_i) = (\sum_{j<i} |B_j|) + \frac{|B_i| + 1}{2}$$

Intuitively, $pos(B_i)$ is the average location within the bucket. The position for a page x, $\sigma(x)$ in list σ is assigned the bucket position for B where x belongs to B.

Spearman's footrule distance for two partial rankings σ_1 and σ_2 is defined as follows:

$$F(\sigma_1, \sigma_2) = \frac{\Sigma_{i=1}^{n} |\sigma_1(i) - \sigma_2(i)|}{\lfloor |\sigma_1|^2/2 \rfloor}$$

.

We use the following symbols in our figures and tables:

- ApproxRank is labeled (▲).
- The first baseline algorithm, local PageRank, is labeled (■).
- The second baseline algorithm, LPR2, is labeled (●). The LPR2 algorithm is a component of the ServerRank algorithm [18]. For a subgraph of size n, an artificial page ξ is added to construct a local graph with $n+1$ pages. If there is an edge connecting local page i to an out-of-domain page, then page i and ξ are connected in the constructed graph. The standard PageRank is computed on this graph.
- SC is labeled (♦).

C. Performance on the TS Subgraphs

We conduct experiments on the same dataset used by the SC approach and compare the distance from the global PageRank for the two approaches. The dataset we consider is labeled **politics**. Starting from the set of pages under the "politics" hierarchy in the dmoz open directory project [37], the dataset is a crawl of pages up to four links away from the set of seeded pages. This dataset contains 4.4 million pages and 17.3 million links. Within the **politics** dataset, we consider the following three TS subgraphs, **liberalism, conservatism, socialism**. These subgraphs pages are identified by their corresponding dmoz categories, as well as by crawling to all pages within three links.

We report on the L_1 distance and the Spearman's footrule distance for SC and ApproxRank in Table III. We note that we have two values for the L_1 distance for SC. The values in column *SC (KDD)* were reported in [1] and *SC (Implemented)* was our implementation of SC. Since the SC approach expands subgraphs based on the influence scores of external pages, which may have ties, it is possible that a subgraph is expanded to different supergraphs. This explains our SC implementation may produce different L_1 distance compared to results in [1].

For the L_1 distance, ApproxRank has slightly superior behavior to SC for the subgraphs **liberalism** and **conservatism**. SC outperforms ApproxRank for **socialism**. For all the subgraphs reported in Table III, ApproxRank significantly outperforms SC for the Spearman's footrule distance value.

To summarize, ApproxRank shows similar (sometimes superior) behavior to SC for the L_1 distance and outperforms SC for the Spearman's footrule distance. We note that in many applications, e.g., Top-K query answering, the accuracy of the ordering (measured by Spearman's footrule distance) is more important than the accuracy of the scores (measured by L_1 distance).

D. Performance on the DS Subgraphs

Next, we present results of experiments on dataset **AU**. We report on the Spearman's footrule distance on each of the **DS** subgraphs from the **AU** dataset for ApproxRank, SC, and the two baseline algorithms, in Table IV. The performance of ApproxRank (▲), in the last column, is typically an order of magnitude better compared to local PageRank (■) and significantly outperforms the SC (♦) and LPR2 (●) – the distance values are at least 5 times smaller.

In **AU** dataset, the global graph consists of 38 domains and there are 3884199 pages and 23898513 links. Table IV lists 12 domains in ascending order of number of pages in **AU** dataset. The second column, *(%) of global graph*, reports on the size of the domain as a percentage of the global graph; the size ranges from 0.35% to 10.42%. We note that this is an independent variable, i.e., the domains are pre-defined.

First, we observe that as the size increases (as a percentage of the global graph), the distance decreases, for all algorithms. For example, the first row of Table IV is domain $acu.edu.au$ which is 0.35% of the global graph. The distance for local PageRank is as poor as 0.19171 whereas the distance for ApproxRank is 0.012112. The last row is domain $anu.edu.au$ which is 10.42% of the global graph. The distance for local PageRank has now improved to 0.04516 while the distance for ApproxRank is 0.004945.

The second and more interesting observation is that based on the Spearman's footrule distance, SC shows poor accuracy of ranking compared to ApproxRank The performance of SC lies between LPR2 and local PageRank in these domains. For example, the distance for SC ranges from 0.02048 to 0.15654; it is similar to the distance for LPR2 which ranges from 0.02022 to 0.10938. In contrast, the corresponding distances for ApproxRank is significantly better (distance is less) and ranges from 0.003934 to 0.013611.

subgraph	SC (KDD) L_1 distance	SC (Implemented) L_1 distance	ApproxRank L_1 distance	SC (Implemented) Spearman's footrule	ApproxRank Spearman's footrule
conservatism	0.0496	0.0476	0.0450	0.0632	0.0255
liberalism	0.0622	0.0733	0.0494	0.0917	0.0293
socialism	0.04318	0.0442	0.104	0.0316	0.0193

TABLE III

THE DISTANCE COMPARISON FOR TS SUBGRAPHS ON THE POLITICS DATASET.

Domain	(%) of global graph	Average outdegree	local PageRank (■)	SC (◆)	LPR2 (●)	ApproxRank (▲)
acu.edu.au	0.35	4.71	0.19171	0.15654	0.10938	0.012112
bond.edu.au	0.50	5.31	0.11049	0.09679	0.09102	0.013611
canberra.edu.au	0.66	5.92	0.10839	0.09197	0.07839	0.012554
cdu.edu.au	0.75	8.74	0.11999	0.09418	0.07898	0.012589
ballarat.edu.au	0.82	5.80	0.07317	0.06471	0.05762	0.006625
cqu.edu.au	0.95	3.80	0.11344	0.09033	0.06722	0.011167
csu.edu.au	2.58	4.26	0.07583	0.05745	0.04826	0.008273
adelaide.edu.au	2.91	5.27	0.08901	0.08321	0.06970	0.009757
curtin.edu.au	2.91	5.55	0.05306	0.03118	0.02771	0.005799
jcu.edu.au	5.04	4.44	0.04823	0.02957	0.02719	0.004614
monash.edu.au	8.45	6.54	0.04101	0.02048	0.02022	0.003934
anu.edu.au	10.42	5.03	0.04516	0.02446	0.02760	0.004945

TABLE IV

THE SPEARMAN'S FOOTRULE DISTANCE FOR DS SUBGRAPHS ON THE AU DATASET.

To summarize, ApproxRank significantly outperforms SC and both baseline algorithms for the DS subgraphs.

E. Performance on the BFS Subgraph

We next experiment on graphs created by a Breadth First Search crawler, **BFS** subgraphs. We use a BFS crawler, where the crawl starts from seeded page *http://www.sounddesign.unimelb.edu.au/web/biogs/gallery/P000517g.htm*. We consider a sequence of **BFS** subgraphs, as the subset of pages that are reached by the crawler ranges from $0.1\%, 0.5\%, 2\%, 5\%, 8\%, 10\%, 12\%, 15\%$, to 20%. We note that the pages in a BFS subgraph can be in different domains.

Since a majority of links in the Web graph are intra-domain links [27], and these intra-domain links may connect local pages and external pages in **BFS** subgraphs, the interaction between local pages and the external pages can have a more significant impact on the ranking of the subgraph. If this is true, we can expect a negative impact on the performance of the algorithms for **BFS** subgraphs.

Figure 7 reports on the distances for the **BFS** datasets. We first observe that the distances are much larger compared to those in Table IV that reports on **DS** graphs, for the same **AU** dataset. For example, for the **BFS** subgraph of size 10%, the distances of ApproxRank and local PageRank are 0.0197 and 0.153, respectively. The corresponding values for the **DS** subgraph for `anu.edu.au`, of size 21.86%, (the last row of Table IV), is 0.004945 and 0.04516, respectively. In general, the distances on the **BFS** subgraphs appear to be an order of magnitude greater, compared to a **DS** subgraph of similar size.

Our second observation is that ApproxRank generally shows an order of magnitude improvement in comparison to the two baseline algorithms. Since the interaction between local pages and external pages may be intra-domain links, there are much more number of external pages for **BFS** subgraphs. SC becomes very expensive to estimate the influence scores for all external pages so we did not obtain the SC ranking for the larger subgraphs. For the smallest two **BFS** subgraphs in Figure 7, ApproxRank outperforms SC significantly.

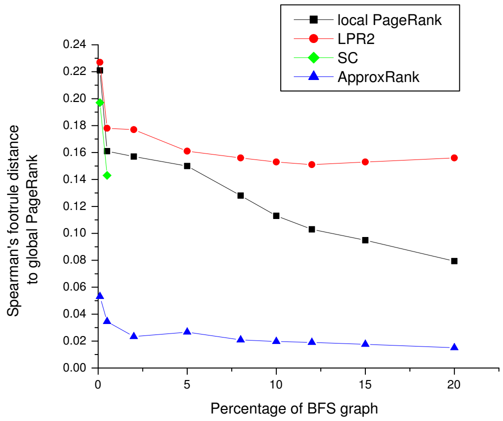

Fig. 7. Spearman's footrule distance for BFS subgraphs on AU dataset.

We also note that the worst accuracy was shown by LPR2 for all **BFS** subgraphs. This again can be explained by the heavy connectivity between the subgraph and external pages. Unlike ApproxRank that modifies the transition probabilities, LPR2 simply connects a local page to the artificial page even when there are multiple links in the global graph. Hence on **BFS** subgraphs, LPR2 further underestimates this connectivity.

F. Runtime Performance

We compare the runtime efficiency of ApproxRank in comparison to the SC approach. We also report on the runtime of the global PageRank algorithm and local PageRank to provide a context.

The disadvantage for SC runtime performance is that it computes the supergraph for each subgraph. In the process of creating the supergraph, it expands the local graph of size n by estimating the influence of each candidate outgoing page on the local graph. To decide the influence of each page, it estimates the PageRank for a graph of size $(n{+}1)$. This implies that the creation of the supergraph involves the PageRank estimation for many graphs of size $(n + 1)$. ApproxRank, on the other hand, processes the global graph for one time and determines the transition matrix A_{approx} for its random walk. When the rankings on multiple subgraphs need to be computed, we can preprocess the global graph for one time, and decide A_{approx} for each subgraph with only local cost.

Table V and VI provide runtime details for ApproxRank, SC, and local PageRank, for the **TS** subgraphs and the **DS** subgraphs. The second column, *#nodes in local graph*, reports on the number of pages in the subgraph; the third column to the fifth column report on the runtime of local PageRank, ApproxRank, and SC, respectively. The sixth column, the value k, shows the number of external pages selected by SC and added to the local graph through each expansion. The last 3 columns report the number of external pages in the first three expansions of SC, which reveal the cost of SC to some extent.

For the global graph **politics** with 4382829 pages, the global PageRank computation takes 5480 seconds. ApproxRank shows an order of magnitude or better runtime performance, and its execution ranges from 484 to 571 seconds. The runtime of the SC approach largely depends on the number of external pages reached by the local graph through expansions. For example, for TS subgraph **socialism**, the initial graph is 12991 pages and SC considers 15936 pages in the third expansion. The runtime for SC is 652 seconds and is slightly worse than ApproxRank. However, for the larger TS subgraphs, **conservatism** of 42797 pages, and **liberalism** of 61724 pages, the SC solution is at least five times as expensive compared to ApproxRank.

Table VI report the runtime on DS subgraphs for the **AU** dataset. The cost of global PageRank on this global graph of 3884199 pages is 7035 seconds with 131 iterations. The runtime for ApproxRank ranges from 110 to 468 seconds. SC shows very poor runtime performance. For the first few rows of the table the runtime ranges from 894 to 2047 seconds. However, for the last rows, where the graph is much larger, the performance of SC sharply degrades. In some cases, e.g., the last two rows, the SC performance is even worse than the exact computation of global PageRank. The high overhead of SC is a trade-off with the lack of access to the global graph.

The runtime for SC on BFS subgraphs is much higher than the runtime on TS and DS subgraphs. The running time of SC was 14655 seconds for the BFS subgraph of size 19420, while for the other types of subgraphs of similar size, the runtime of SC was 652 seconds for TS subgraph **socialism** of size 12991 and 1310 seconds for DS subgraph $bond.edu.au$ of size 19559. ApproxRank, on the other hand, seems not as sensitive to the subgraph types. For example, the runtime for ApproxRank on the BFS subgraph of size 19420 is 142 seconds, and ApproxRank takes 484 seconds on **socialism** and 110 seconds on $bond.edu.au$.

VI. Conclusions

We propose a framework of an exact solution and an approximate solution for computing PageRank on a subgraph. The IdealRank algorithm is an exact solution, and the ApproxRank algorithm estimates PageRank scores for the subgraph. We show that IdealRank scores converge to the true PageRank scores that are obtained through global computation. We conduct error analysis for the ApproxRank scores and test ApproxRank algorithm on various types of subgraphs on two datasets. We compare ApproxRank and a stochastic complementation (SC) approach and show ApproxRank has similar or superior performance to SC but with lower overhead. We demonstrate that ApproxRank predict PageRank scores accurately for a variety of subgraphs. The ApproxRank outperforms two baseline algorithms in an order of magnitude.

Acknowledgment

We thank Jason V. Davis for providing data and useful discussions. We thank Samir Khuller and Felix Naumann for their comments. This research is partially supported by the National Science Foundation grant IIS0430915.

References

[1] J. V. Davis and I. S. Dhillon, “Estimating the global pagerank of web communities,” in *KDD ’06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA: ACM Press, 2006, pp. 116–125.

[2] S. Brin and L. Page, “The anatomy of a large-scale hypertextual web search engine,” in *WWW7: Proceedings of the seventh international conference on World Wide Web 7*. Amsterdam, The Netherlands, The Netherlands: Elsevier Science Publishers B. V., 1998, pp. 107–117.

[3] J. M. Kleinberg, “Authoritative sources in a hyperlinked environment,” *Journal of the ACM*, vol. 46, no. 5, pp. 604–632, 1999. [Online]. Available: citeseer.ist.psu.edu/kleinberg99authoritative.html

[4] A. Gulli and A. Signorini, “The indexable web is more than 11.5 billion pages,” in *WWW ’05: Special interest tracks and posters of the 14th international conference on World Wide Web*. New York, NY, USA: ACM Press, 2005, pp. 902–903.

[5] *http://www.useit.com/alertbox/web-growth.html*.

[6] S. Chakrabarti, M. van den Berg, and B. Dom, “Focused crawling: a new approach to topic-specific web resource discovery,” in *WWW ’99: Proceeding of the eighth international conference on World Wide Web*. New York, NY, USA: Elsevier North-Holland, Inc., 1999, pp. 1623–1640.

[7] M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori, “Focused crawling using context graphs,” in *VLDB ’00: Proceedings of the 26th International Conference on Very Large Data Bases*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 527–534.

[8] A. Balmin, V. Hristidis, and Y. Papakonstantinou, “ObjectRank: Authority-based keyword search in databases,” in *VLDB*, 2004. [Online]. Available: citeseer.ist.psu.edu/balmin04objectrank.html

[9] R. Varadarajan, V. Hristidis, and L. Raschid, “Explaining and reformulating authority flow queries,” in *Proceedings of the IEEE International Conference on Data Engineering*, 2008.

subgraph	#nodes in	local PR	ApproxRank	SC	k	#ext nodes	#ext nodes	#ext nodes
	local graph	(seconds)	(seconds)	(seconds)		1st expansion	2nd expansion	3rd expansion
conservatism	42797	63	542	3002	1711	25870	55156	71336
liberalism	61724	69	571	3483	2468	51283	93653	110481
socialism	12991	7	484	652	519	4170	11540	15936

TABLE V

THE RUNTIME COMPARISON ON TS SUBGRAPHS.

subgraph	#nodes in	local PR	ApproxRank	SC	k	#ext nodes	#ext nodes	#ext nodes
	local graph	(seconds)	(seconds)	(seconds)		1st expansion	2nd expansion	3rd expansion
acu.edu.au	13785	8	319	894	551	1172	6519	13769
bond.edu.au	19559	11	110	1310	782	1826	7918	16502
canberra.edu.au	25501	15	114	1700	1020	3590	10521	20705
cdu.edu.au	29039	25	152	2059	1161	4068	14176	24767
ballarat.edu.au	31724	22	134	2037	1268	1501	15215	27242
cqu.edu.au	36948	16	128	2047	1477	4029	15709	28955
csu.edu.au	100191	59	165	5306	4007	7609	36445	58557
adelaide.edu.au	113181	91	267	6276	4527	13714	45358	73579
curtin.edu.au	113221	80	197	6552	4528	6924	41595	67271
jcu.edu.au	195691	135	272	10327	7827	15705	60966	108644
monash.edu.au	328062	346	468	20292	13122	15489	90993	150890

TABLE VI

THE RUNTIME COMPARISON ON DS SUBGRAPHS.

[10] R. Varadarajan, V. Hristidis, L. Raschid, M. Vidal, H. Rodriguez, and L. Ibanez, “Flexible and efficient querying and ranking on hyperlinked data sources,” University of Maryland CLIP Lab Technical Report, Tech. Rep., 2008.

[11] A. Gulli and A. Signorini, “Building an open source meta-search engine,” in *WWW ’05: Special interest tracks and posters of the 14th international conference on World Wide Web*. New York, NY, USA: ACM Press, 2005, pp. 1004–1005.

[12] T. Suel, C. Mathur, J.-W. Wu, J. Zhang, A. Delis, M. Kharrazi, X. Long, and K. Shanmugasundaram, “Odissea: A peer-to-peer architecture for scalable web search and information retrieval.” in *WebDB*, 2003, pp. 67–72.

[13] D. Zeinalipour-Yazti, V. Kalogeraki, and D. Gunopulos, “Information retrieval techniques for peer-to-peer networks,” *Computing in Science and Engg.*, vol. 6, no. 4, pp. 20–26, 2004.

[14] N. Eiron, K. S. McCurley, and J. A. Tomlin, “Ranking the web frontier,” in *WWW ’04: Proceedings of the 13th international conference on World Wide Web*. New York, NY, USA: ACM Press, 2004, pp. 309–318.

[15] A. N. Langville and C. D. Meyer, “Updating markov chains with an eye on google’s pagerank,” *SIAM J. Matrix Anal. Appl.*, vol. 27, no. 4, pp. 968–987, 2006.

[16] J. X. Parreira, D. Donato, S. Michel, and G. Weikum, “Efficient and decentralized pagerank approximation in a peer-to-peer web search network,” in *VLDB’2006: Proceedings of the 32nd international conference on Very large data bases*. VLDB Endowment, 2006, pp. 415–426.

[17] Y.-Y. Chen, Q. Gan, and T. Suel, “Local methods for estimating pagerank values,” in *CIKM ’04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*. New York, NY, USA: ACM Press, 2004, pp. 381–389.

[18] Y. Wang and D. J. DeWitt, “Computing pagerank in a distributed internet search system.” in *VLDB*, 2004, pp. 420–431.

[19] P. Berkhin, “A survey on pagerank computing,” *Internet Mathematics*, vol. 2, no. 1, pp. 73–120, 2005.

[20] A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas, “Link analysis ranking: algorithms, theory, and experiments,” *ACM Trans. Inter. Tech.*, vol. 5, no. 1, pp. 231–297, 2005.

[21] A. N. Langville and C. D. Meyer, *Google’s PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press, July 2006.

[22] S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub, “Extrapolation methods for accelerating pagerank computations.” in *WWW*, 2003, pp. 261–270.

[23] R. Motwani and P. Raghavan, *Randomized algorithms*. New York, NY, USA: Cambridge University Press, 1995.

[24] A. Z. Broder, R. Lempel, F. Maghoul, and J. Pedersen, “Efficient pagerank approximation via graph aggregation,” in *WWW Alt. ’04: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*. New York, NY, USA: ACM Press, 2004, pp. 484–485.

[25] G. M. D. Corso, A. Gullí, and F. Romani, “Fast pagerank computation via a sparse linear system,” *Journal of Internet Mathematics*, vol. 2, no. 3, pp. 251–273, 2005.

[26] S. Kamvar, T. Haveliwala, and G. Golub, “Adaptive methods for the computation of pagerank,” Stanford University, Tech. Rep., 2003. [Online]. Available: citeseer.ist.psu.edu/kamvar03adaptive.html

[27] S. Kamvar, T. Haveliwala, C. Manning, and G. Golub, “Exploiting the block structure of the web for computing pagerank,” Stanford Digital Library Technologies Project, Tech. Rep., 2003. [Online]. Available: citeseer.ist.psu.edu/kamvar03exploiting.html

[28] K. Aberer and J. Wu, “A framework for decentralized ranking in web information retrieval.” in *APWeb*, 2003, pp. 213–226.

[29] G.-R. Xue, H.-J. Zeng, Z. Chen, W.-Y. Ma, H.-J. Zhang, and C.-J. Lu, “Implicit link analysis for small web search,” in *SIGIR ’03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. New York, NY, USA: ACM Press, 2003, pp. 56–63.

[30] W. J. Stewart, *Introduction to the Numerical Solution of Markov Chains*. Princeton, NJ, USA: Princeton University Press, 1994.

[31] D. Hooley, “Collapsed matrices with (almost) the same eigenstuff,” vol. 31, no. 4, 2000, pp. 297–299.

[32] A. Y. Ng, A. X. Zheng, and M. I. Jordan, “Link analysis, eigenvectors and stability,” in *IJCAI*, 2001, pp. 903–910.

[33] S. Chien, C. Dwork, R. Kumar, D. R. Simon, and D. Sivakumar, “Link evolution: Analysis and algorithms,” *Internet Mathematics*, vol. 1, no. 3, 2003.

[34] Z. Nie, Y. Zhang, J.-R. Wen, and W.-Y. Ma, “Object-level ranking: bringing order to web objects,” in *WWW ’05: Proceedings of the 14th international conference on World Wide Web*. New York, NY, USA: ACM, 2005, pp. 567–574.

[35] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, “Rank aggregation methods for the web,” in *WWW ’01: Proceedings of the 10th international conference on World Wide Web*. New York, NY, USA: ACM Press, 2001, pp. 613–622.

[36] R. Fagin, R. Kumar, M. Mahdian, D. Sivakumar, and E. Vee, “Comparing and aggregating rankings with ties,” in *PODS ’04: Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. New York, NY, USA: ACM Press, 2004, pp. 47–58.

[37] *http://www.dmoz.org/*.