JXP: Global Authority Scores in a P2P Network

Josiane Xavier Parreira, Gerhard Weikum

2005
**JXP: Global Authority Scores in a P2P Network**

Josiane Xavier Parreira  
Max-Planck Institute for Computer Science  
Stuhlsatzenhausweg 85  
66123 Saarbrücken, Germany  
jparreir@mpi-sb.mpg.de

Gerhard Weikum  
Max-Planck Institute for Computer Science  
Stuhlsatzenhausweg 85  
66123 Saarbrücken, Germany  
weikum@mpi-sb.mpg.de

**ABSTRACT**

This document presents the JXP algorithm for dynamically and collaboratively computing PageRank-style authority scores of Web pages distributed in a P2P network. In the architecture that we pursue, every peer crawls and indexes Web fragments at its discretion, driven by the thematic profile or overlay neighborhood of the peer. The JXP algorithm runs at every peer, and is initialized by a local authority computation on the basis of the locally available Web fragment. Peers collaborate by periodically “meeting” with other peers in the network. Whenever two peers meet they exchange their local information and use this new information to improve their local authority scores. Even though only local computations are performed, the JXP scores approximate the global importance of pages in the entire network. The storage demand of each peer is linear in the number of Web pages and the locally stored Web fragment. Experiments show the quality and practical viability of the JXP algorithm.

**1. INTRODUCTION**

This paper is motivated by efforts towards building a peer-to-peer (P2P) Web search engine. P2P networks [26, 23, 22] have proven to be a scalable alternative to traditional client/server systems. However, the characteristics of such networks, namely, no central processing and high dynamics (peers constantly joining and leaving the network), pose a challenge when designing a new search engine for a P2P network. We assume that every peer has a full-fledged Web search engine and can crawl and index interesting Web fragments at its discretion, driven by thematic profiles of the user or the neighborhood in some form of “semantic” overlay network. Peers collaborate on difficult queries that cannot be satisfactorily answered with the locally available index alone (using query routing strategies [14, 8, 4], but they are autonomous in terms of their crawling strategies and what data they keep in their local indexes).

In the context of Internet search engines, link-based ranking algorithms that assign authority scores to pages, based on their “importance” on the Web, have been proven to make the search more effective [5, 18]. For instance, Google [2] uses PageRank, an Eigenvector-based algorithm that determines the importance of a page based on the importance of the pages that point to it. The PageRank computation is quite expensive as it involves iteratively computing the principal Eigenvector of a matrix derived from the Web link graph. An alternative but equivalent view of PageRank is that it computes stationary probabilities of a Markov chain that corresponds to a random walk on the Web. Recent work has made progress on efficiently computing PageRank scores [17, 16, 6, 12], but the high storage demand of the sparse but nonetheless huge - underlying matrix seems to limit this kind of link analysis to a central server with very large memory. The most relevant prior work on distributed PageRank computation is [27], but this method assumes that sites compute local PageRank values based on the data that they originally host, thus strongly relying on the assumption that sites have disjoint fragments of the Web graph and are relatively reliable servers.

In this document we propose the JXP, an algorithm for dynamically computing, in a decentralized P2P manner, global authority scores when the Web graph is spread across many autonomous peers. The peers’ graph fragments may overlap arbitrarily, and peers are (a priori) unaware of other peers’ fragments.

The main idea of the JXP algorithm is as follows. Each peer computes the authority scores of the pages that it has in its local index, by locally running the standard PageRank algorithm. To avoid confusion with the true, global PageRank values, we refer to these local scores as the (peer-specific) PageWeight scores of the pages known by the peer. Note that a page may be known and indexed by multiple peers, and these may have different PageWeights for that same page. A peer gradually increases its knowledge about the rest of the network by meeting with other, randomly chosen, peers and exchanging information. To improve the initial PageWeight scores and approximating the global authority of pages, a peer uses what it learns from the other, randomly met, peers, combined with its own local information, for recomputing the PageWeight scores. Although the computations are strictly local, our goal is towards a notion

---

*Partially supported by the EU within the 6th Framework Programme under contract 001907 “Dynamically Evolving, Large Scale Information Systems” (DELIS).
of importance of the pages in the whole web graph. This process, in principle, runs forever, and our experiments indicate that the resulting JXP scores quickly converge to the true, global PageRank scores.

When two peers meet they temporarily form the union of their Web graph fragments; for representing the unknown part of the Web graph (which is spread across many further peers) a state-lumping technique for Markov chains is used. After recomputing PageWeights, only the resulting authority scores of each peer’s pages of interest are kept. Technically, the computation involves some difficulties because of the need for proper normalization with partial knowledge of the Web graph; another complication is that the graphs of two peers may have radically different sizes and may arbitrarily overlap. It is important to emphasize that peers do not accumulate the graph fragments that they learn about when meeting other peers. So we ensure that the storage requirements are low, linear in the number of pages of interest and the local index size, and the PageRank computation is scalable, as the algorithm always runs on relative small graphs, independent of the number of peers in the network.

The locally recomputed PageWeights already reflect the importance of a page in the entire network, but different peers may have very different views, e.g., in terms of the size of their local graphs. Therefore, the JXP algorithm can optionally normalize authority scores based on PeerWeights that reflect the reputation and trust of peers.

The rest of the document is organized as follows. Section 2 briefly discusses related work. A quick review of the standard PageRank algorithm is presented in Section 3. In Section 4 we present the JXP algorithm for computing authority scores. Section 5 shows preliminary experimental results of the algorithm. Section 6 concludes this paper and presents ideas for future work.

2. RELATED WORK

Link-based authority ranking has received great attention in the literature, starting with the seminal work of Brin and Page [5] and Kleinberg [18]. Good surveys of the many improvements and variations are given in [10] and [19].

In [27] Wang and DeWitt presented a distributed search engine framework, in which the authority score of each page is computed by performing the PageRank algorithm at the Web server that is the responsible host for the page, based only on the intra-server links. They also assign authority scores for each server in the network, based on the inter-server links, and then approximate global PageRank values by combining local page authority scores and server authority values. This work bears relationships to the work by Haveliwala et al. [17] that postulates a block structure of the link matrix and exploits this structure for faster convergence of the global PageRank computation. Our idea is related to the approach of Wang and DeWitt in the sense that we also use local page scores and peer scores, but our algorithm does not require any particular distribution of the pages among the sites whereas the method by Wang and DeWitt relies on the fact that the graph fragments of the servers are disjoint. In particular, the JXP algorithm can work in scenarios where peers are completely autonomous and crawl and index Web data at their discretion, resulting in arbitrarily overlapping graph fragments.

Chen et al. [11] proposed a way of approximating the PageRank value of a page locally, by expanding a small subgraph around the page of interest, placing an estimated PageRank at the boundary nodes of the subgraph, and running the standard algorithm. In a P2P scenario, this algorithm would require the peers to query other peers about pages that have links to their local nodes, and pages that point to pages that point to local pages, and so on. This would be a significant burden for a highly dynamic P2P network. The JXP algorithm, on the other hand, requires much less interaction among peers.

Other techniques for approximating PageRank-style authority scores with partial knowledge of the global graph use state-lumping techniques from the stationary analysis of large Markov chains [20, 12]. These techniques have been developed for the purpose of incremental updates to authority scores when only small parts of the graph have changed.

Dynamic computation in a P2P network is not an issue in this prior work. Another work related to this topic is the one by Broder and Lempel [6], where they have presented a graph aggregation method in which pages are partitioned into hosts and the stationary distribution is computed in a two-step approach, combining the stationary distribution inside the host and the stationary distribution inter-hosts. A storage-efficient approach to computing authority scores is the OPIC algorithm developed by Abiteboul et al. [3]. This method avoids having the entire link graph in one site, which, albeit sparse, is very large and usually exceeds the available main memory size. It does so by randomly (or otherwise fairly) visiting Web pages in a long-running crawl process and performing a small step of the PageRank power iteration (the numeric technique for computing the principal Eigenvector) for the page and its successors upon each such visit. The bookkeeping for tracking the gradually approximated authority of all pages is carried out at a central site, the Web-warehouse server. This is not a P2P algorithm either.

In [24], Sankaralingam et al. presented a P2P algorithm in which the PageRank computation is performed at the network level, with peers constantly updating the scores of their local pages and sending these updated values through the network. Shi et al. [25] also compute PageRank at the network level, but they reduce the communication among peers by distributing the pages among the peers according to some load-sharing function. In contrast to these P2P-style approaches, our JXP algorithm performs the actual computations locally at each peer, and thus needs a much smaller number of messages.

3. REVIEW OF PAGERANK

The basic idea of PageRank is that if page p has a link to page q then the author of p is implicitly endorsing, i.e., giving some importance to page q. How much p contributes to the importance of q is proportional to the importance of p itself.

This recursive definition of importance can be described by the stationary distribution of a Markov chain that describes a random walk over the graph, where we start at an arbitrary page and in each step choose a random outgoing edge from the current page. To ensure the ergodicity of this Markov chain (i.e., the existence of stationary page-visit probabilities), additional random jumps to uniformly chosen target pages are allowed with small probability α. Formally, the PageRank of a page q is defined as:
PageRank is defined as

$$PR(q) = \alpha \times 1/N + (1 - \alpha) \times \sum_{p|p\rightarrow q} PR(p)/\text{out}(p)$$  \hspace{1cm} (1)$$

where \(N\) is the total number of pages in the link graph, \(PR(p)\) is the PageRank score of the page \(p\), \(\text{out}(p)\) is the outdegree of \(p\), the sum ranges over all link predecessors of \(q\), and \(\alpha\) is the random jump probability, with \(0 < \alpha < 1\) and usually set to a value like 0.15.

PageRank values are usually computed by initializing a PageRank vector with uniform values \(1/N\), and then applying a power iteration method, with the previous iteration’s values substituted in the right-hand side of the above equation for evaluating the left-hand side. This iteration step is repeated until sufficient convergence, i.e., until the PageRank scores of the high-authority pages of interest exhibit only minor changes.

4. THE JXP ALGORITHM

The goal of the JXP algorithm is to approximate global authority scores by performing local computations only, with low storage costs, and a moderate number of interactions among peers. JXP runs on every peer in the network, where each peer stores only its own local fragment of the global graph. The algorithm does not assume any particular assignment of pages to peers, and overlaps among the graph fragments of the peers are allowed.

The JXP algorithm has three components that are described in the subsequent subsections:

1. the local PageWeight computation based on an extended local graph,
2. the interaction with other peers, chosen at random, and
3. optional considerations on the normalization of the resulting PageWeights, from an individual peer’s viewpoint, by taking into account the PeerWeights, the relative authority or trust of the peers.

4.1 Local PageWeight Computation

For the local approximation of the global graph, as viewed from a peer with partial knowledge of the link structure, we construct an extended local graph. There are two different cases to consider: initial PageWeight computations by a peer that is just by itself, and PageWeight refinements when two peers meet. In the first case, we add to the local graph a special node, that we call the world node, representing the part of the global graph that is not stored at and not known to the peer. This is an application of the state-lumping techniques used in the analysis of large Markov models [20]. In the second case, when two peers meet we form the union of the two local graphs and extend them by a world node. We discuss the graph merging for meeting peers in Subsection 4.2. In both cases, the local PageWeight scores are then computed by running the PageRank power iteration algorithm on this extended local graph. Figure 1 depicts a peer’s local graph with the additional world node.

The world node has special features, regarding its own PageWeight and how it is connected to the local graph. As it represents all the pages that are not stored at the peer, we take all the links from local pages to external pages and make them point to the world node. In the same way, as the peer learns about external links that point to one of the local pages, we assign these links to the world node. (This is when the peer meets with another peer, see Subsection 4.2.) For a better approximation of the total authority score mass that is received from external pages, we weigh every link from the world node based on how much of the authority score is received from the original page that owns the link. Another special feature of the world node is that it contains a self-loop link, that represents links from external pages pointing to other external pages. The PageWeight of the world node is equivalent to the sum of PageWeights of the external pages that it represents. During the local PageWeight computation the probability of a random jump to the world node is also set proportional to the number of pages it represents.

Let \(\text{Internal} = \{\text{int}_1, \text{int}_2, \ldots, \text{int}_n\}\) be the set of local pages and \(\text{External} = \{\text{ext}_1, \text{ext}_2, \ldots, \text{ext}_{1\to N}\}\) the set of external pages. As peers gradually learn about external pages, we also define \(\text{Known}\) as a subset from \(\text{External}\) that contains pages that the peer has learned about and that have links to one of the pages in \(\text{Internal}\). Then the PageWeight of the world node \(W\) and the weights of a link from the world node \(W\) to some local node \(a\in\text{Internal}\) can be expressed as:

$$\text{PageWeight}(W) = \sum_{i \in \text{Internal}} \text{PageWeight}(i)$$
$$\text{PageWeight}(W) = \sum_{i \in \text{Internal}} \text{PageWeight}(i)$$

$$\text{weight}(W \rightarrow a) = \frac{1}{\text{PageWeight}(W)} \times \sum_{i \in \text{Known} & i \rightarrow a} \text{PageWeight}(i)$$

$$\text{Outdegree}(i)$$

When the local PageWeight computation on the extended local graph terminates, the PageWeights of pages learned from other peers are also stored at the local peer, as they can be used for better estimation of the weights of links from the world node before the next local evaluation of PageWeights. We also estimate a PageWeight score for the pages that are still not known by the peer, based on an estimation of the total number of the pages in the graph. Following the same formulation above, let \(\text{Unknown} = \{\text{ukwn}_1, \text{ukwn}_2, \ldots, \text{ukwn}_m\}\), where \(m = N - (\text{size}(\text{Internal}) + \text{size}(\text{Known}))\). The PageWeights of the unknown pages are defined as

$$\text{PageWeight}(\text{ukwn}_i) = (1 - \sum_{i \in \text{Internal}} \text{PageWeight}(i)) + \sum_{j \in \text{Known}} \text{PageWeight}(j)) \times \frac{1}{m}$$

Figure 1: Extended local graph of a peer
for all pages $ukwn$, in Unknown.

Before the execution of the local PageWeight algorithm, an initialization procedure, described in Algorithm 1, is performed. This procedure estimates the size of the global graph, creates the world node and attaches it to the local graph, sets an initial PageWeight scores $(1/N)$ for all pages on the local graph, and $(N-n)/N$ for the world node), and runs the PageRank power iteration algorithm on the extended local graph, to improve the previous scores.

Algorithm 1 Initial PageWeight Computation

1: input: local graph $G$ and est. size of global graph $N$
2: $n \leftarrow \text{size}(G)$
3: Create world node $W$
4: $\text{PageWeight}(p | p \in G) \leftarrow 1/N$
5: $\text{PageWeight}(W) \leftarrow (N-n)/N$
6: add $W$ to $G$
7: run PageRank power iteration on $G$
8: Update(PageWeights)

4.2 Peer Meetings

Algorithm 2 shows the pseudo code of the PageWeight algorithm. It starts with the peer choosing another peer in the network at random to exchange information. Once the peers have exchanged their local knowledge, it is time for them to combine both local and external information. It is important to point out that this process runs at both peers independently of each other. So we fully preserve the autonomy of peers and asynchronous nature of communication and computation in a P2P network.

Algorithm 2 The JXP Algorithm

1: input: local graph $G$, world node $W$, known pages $KP$
2: repeat
3: Contact a RandomPeer in the network and exchange information
4: $\text{extG} \leftarrow$ local graph of RandomPeer
5: $\text{extW} \leftarrow$ world node of RandomPeer
6: $\text{extKP} \leftarrow$ list of known pages of RandomPeer
7: $KP \leftarrow \text{combinePageLists}(KP, \text{extKP})$
8: $\text{mergedG} \leftarrow \text{combineGraphs}(G, \text{extG})$
9: $\text{mergedW} \leftarrow \text{combineWorldNodes}(W, \text{extW})$
10: add $\text{mergedW}$ to $\text{mergedG}$
11: run PageRank power iteration on $\text{mergedG}$
12: Update(PageWeights)
13: Update($W$)
14: Discard($\text{extG}, \text{extW}, \text{extKP}, \text{mergedG}, \text{mergedW}$)

The lists of known pages are combined by averaging the PageWeight scores of the pages from both lists. Pages that were unknown to the peer until the current iteration are added to the local list of known pages. Local graphs are combined by simply forming the union of nodes and connecting nodes by the known links between them. The PageWeight computation will always yield a correct result even if there is no link between the graphs, because of the world node and the corresponding random jump probabilities.

Combining the world nodes of the two meeting peers consists of merging their list of outgoing links, removing links that originally come from a page that is already represented in the merged graph, and adjusting the PageWeight of the combined world node to reflect the sum of PageWeights of pages that do not belong to the combined graph. The new world node that results from this merging is then connected to the merged graph and the PageRank power iteration algorithm is again performed, yielding updated PageWeight values. The graphs are then disconnected and the local world node is recreated from the merged world node, by keeping only the links that point to a page in the local graph. The PageWeight of the world node is also recomputed. The extended graph, world node, and list of pages that belong to the other peer are then discarded. Figure 2 illustrates the process of combining and disconnecting local graphs and world nodes.

4.3 Considering PeerWeights

Two meeting peers may have fairly different characteristics in terms of their local index sizes, knowledge or awareness of the global graph characteristics (e.g., because one peer has already met many other peers, whereas the other peer just joined the P2P network), and trustworthiness or recognition by other peers. Thus, when we combine the two local graphs it could make sense to treat the two peers with different weights. The JXP algorithm can optionally take PeerWeights into account. To this end, it weighs all edges $p \rightarrow q$ that has been known to both peers is set to $\text{PeerWeight}_1 + \text{PeerWeight}_2(p \rightarrow q)$ where the subscripts 1 and 2 refer to the two meeting peers. This gives higher weight to the view of the more
authoritative, knowledgable, or credible peer. For computing PeerWeights we have a number of options. The simplest approach is to construct a peer graph with peers as nodes and edges between two peers if there is a link between two pages that are locally indexed by the two peers. This graph would be gradually constructed as peers meet over time; but given that it is orders of magnitude smaller than the Web graph, the peers’ local information about this peer graph could be easily disseminated in the P2P network so that peers learn the full peer graph more quickly. An alternative, which we would actually prefer, is to compute PeerWeights on the basis of the peers’ behavior and trustworthiness or recognition in the P2P community. This would serve to discriminate “good” peers from “bad”, i.e., selfish or even malicious, peers. Recently, various approaches have been proposed in the literature for monitoring and tagging peers, see e.g., [15, 21, 7]. The JXP framework can easily incorporate such techniques, and we are currently investigating the implementation issues.

5. EXPERIMENTS

We evaluated the performance of the PageWeight algorithm on two different datasets: a synthetic generated data-set, and subsets of the Amazon.com dataset. The synthetic web graphs were generated using the “recursive matrix” (R-MAT) model [9], a powerful tool that can, given a few parameters, quickly generate realistic web graphs, with the power law degree distribution property. The Amazon.com dataset contains information about approximately 126,000 products (mostly books) offered by Amazon.com. The data was obtained in February 2005, through a Web Service provided by Amazon.com [1], and it is equivalent to a partial crawl of the corresponding web site. The graphs were created by considering the products as a node in the graph. For each product, pointers to similar recommended products are available in the dataset. These pointers define the edges in our graphs.

5.1 Setup

After creating the graphs, we distributed the nodes into the peers. Nodes are assigned to peers either at random, allowing some overlap among the local graphs, or simulating a crawler in each peer, starting with a random seed page and following the links and fetching the next nodes, in a breadth-first approach, up to a certain predefined depth.

The performance of the algorithm is evaluated by comparing the top-k ranking of the pages given by the JXP algorithm at each peer against the global top-k ranking of the pages given by the PageRank computation in the complete web graph. For this comparison we use Spearman’s footrule distance \( R(i,j) = \sum_{k=1}^{\min(k,N)} |\sigma_1(i) - \sigma_2(i)| \) where \( \sigma_1(i) \) and \( \sigma_2(i) \) are the positions of the page \( i \) in the first and second ranking. In case a page is present in one of the top-k rankings and does not appear in the second top-k ranking, its position in the latter ranking is considered to be \( k + 1 \). We normalized the Spearman’s footrule distance, to obtain values between 0 and 1, with 0 meaning that the rankings are identical, and 1 meaning that the rankings have no pages in common.

5.2 Results

We tested our algorithm on a subset of the Amazon.com dataset containing books from the category “Computers & Internet”, which contains 10,595 pages and 42,548 links. For comparison, we also tested JXP on a synthetic generated graph with 2,036 pages and 8,477 links. Results are shown in Figure 3, where the x-axis corresponds to the number of meetings a peer performed, and each line on the graph represents a different peer. For all the cases we chose the top-k level to be 100.

Besides the Spearman’s footrule distance, we also measured, after each meeting, the total number of pages known by the peer, and the linear score error, that we defined as being the average of the absolute difference between the PageWeight score and the global PageRank score of all known pages. Table 1 shows the average of the observed values, after a certain number of meetings, on the “Computers & Internet” subset, for the case when pages are randomly distributed among peers. The table clearly shows that both Spearman’s footrule distance and the linear score error quickly decrease with the number of meetings.

Based on these results we can conclude that, as the number of meetings among peers increases, the distance between the PageWeight scores and the PageRank scores, as well as the rankings produced by them, decreases at a high rate. Thus, PageWeight provides a good approximation to the global PageRank scores of the pages.

<table>
<thead>
<tr>
<th>Number of Meetings</th>
<th>Footrule Distance</th>
<th>Linear Error ((\times 10^{-5}))</th>
<th>Known Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5773</td>
<td>7.82</td>
<td>579.4</td>
</tr>
<tr>
<td>5</td>
<td>0.59773</td>
<td>7.20</td>
<td>10403.3</td>
</tr>
<tr>
<td>10</td>
<td>0.47106</td>
<td>6.50</td>
<td>10595</td>
</tr>
<tr>
<td>20</td>
<td>0.36186</td>
<td>5.33</td>
<td>10595</td>
</tr>
<tr>
<td>50</td>
<td>0.11694</td>
<td>2.12</td>
<td>10595</td>
</tr>
<tr>
<td>90</td>
<td>0.05531</td>
<td>0.99</td>
<td>10595</td>
</tr>
</tbody>
</table>

Table 1: Average results for “Computers & Internet” data.

6. CONCLUSIONS AND FUTURE WORK

In this paper we presented the JXP algorithm for dynamically computing authority scores of pages distributed in a P2P network. The algorithm runs in a completely decentralized manner, with every peer running the algorithm independently from the other peers in the network. The algorithm requires that peers meet randomly and exchange their local graph fragments, but the overall long-running process does not require any peer to keep more information other than its own local graph fragment and the PageWeights of the pages of interest. The computations themselves are strictly local, yet we can approximate the global importance of a page in the whole graph with acceptable accuracy. Our experiments, albeit preliminary, indicate that the algorithm performs very well, converges fairly quickly, and incurs low overhead.

Future work includes more comprehensive experimentation with larger graphs and a larger number of peers. We expect the algorithm to scale up well as the number of peers increase without increasing the local data volume and local overhead.

\(^2\)We also measured different top-k levels and obtained similar results.

\(^3\)The same behavior was observed for the other tested graphs, but these results are omitted for space reasons.
computational cost. We also aim at providing a mathematical proof for the convergence of the algorithm. Moreover, we plan to extend and explore the algorithm in different scenarios; for instance, we want to test the case that a peer chooses another peer not at random but according to some criteria based on “semantic” relationships between the local interest profiles. We expect that in a semantic overlay network, the PageWeights will converge to the global PageRank scores with even fewer interactions, reducing the number of meetings needed for a good approximation.

7. REFERENCES


