Random Surfer with Back Step

Marcin Sydow
Polish-Japanese Institute of Information Technology
ul. Koszykowa 86,
02-008 Warsaw, Poland
msyd@pjwstk.edu.pl

ABSTRACT

We present a novel link-based ranking algorithm RBS, which may be viewed as an extension of PageRank by back-step feature.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering

General Terms

Algorithms,Experimentation

Keywords

Ranking Algorithms, PageRank, Back Step

1. THE PROBLEM

In this paper we report development of a link-based ranking algorithm which is built on a random surfer model reflecting back steps made by a Web surfer.

Link analysis of the Web turned out to be a powerful approach in the context of exponentially growing Web with documents of extremely varying type, quality and size, where traditional text information retrieval techniques are not robust enough.

Link-based ranking algorithms are used for ordering results of search queries passed to search engines. More precisely, each document is assigned some *ranking score* which is computed by a ranking algorithm according to some intuitive model, and then documents are presented in non-increasing order of their ranking scores. Since the number of documents returned by a search engine may easily be far too large to be seen by a user, such ordering is extremely important.

Link-based ranking algorithms proved their value in practice. The most famous example is perhaps Google's PageRank [1] which is successfully applied in this leading search engine.

1.1 Traditional Random Surfer Model

PageRank is based on the random surfer model [1]. The model has a parameter $0 \leq a \leq 1$. More precisely, in each time step t a surfer, currently visiting a document $p(t)$, with probability a follows uniformly picked out-link or, with the remaining probability $1-a$ (called here a decay factor), randomly jumps to another page with uniform distribution over pages.

Unfortunately, the traditional Random Surfer model *does not* reflect some very common aspect of Web surfing i.e *reverting* to a page visited in the previous step. Reverting to previous pages constitutes a substantial part of Web surfing behavior, and, in our opinion, it *should* be reflected in ranking schemes.

There are many ways of implementing revisiting Web pages in Web browsers – one of the most common is via *back* and *forward* buttons. We build our model on a simplification of this approach – i.e. we assume that there is only a back button available.

In this paper we describe incorporating back step into ranking scheme by properly extending the random surfer model and report experimental results of RBS algorithm, which is our implementation of the proposed novel model.

WWW2004, May 17–22, 2004, New York, New York, USA.
ACM 1-58113-912-8/04/0005.

2. PREVIOUS WORK

There have been proposed some link-based ranking algorithms before (Hits, "Unified framework" or Salsa), which incorporate the notion of *backward flow* of rank, but in order to compute so-called *hub* score rather than to model back step.

On the other hand, Fagin et al. [3] analyse random walks with back steps in the context of Web surfing, but approach presented in that paper differs from ours. Authors do not aim at developing link-based ranking algorithm for Web search. Furthermore, they define their main model so that the result is equivalent to the result obtained in the traditional random surfer model, because they allow for back-step after any forward step, which is not realistic in our opinion. Due to this, our back step is formulated slightly differently. Most importantly, the computational methods proposed in [3] would be of prohibitively high time complexity when applied to large scale Web page collections, despite they are polynomial.

In contrast, our approach results in really effective method of computing rank score which reflects back-step and produces *different* ranking than PageRank.

3. RANDOM SURFER WITH BACK STEP

We extend the traditional random surfer model by introducing another parameter $0 \leq b \leq 1$, satisfying $a+b \leq 1$, which represents the *back-step* probability. Random surfer is visiting pages in discrete time steps, so that in each step while visiting a page p they may perform one of the following actions:

1. with probability a follow uniformly picked outgoing link of the page p. In case there are no out-links on the page, the surfer performs random jump described in the last point.
2. with probability b *reverts* to the previously visited page, assuming the previous step was of type 1 (follow link) – otherwise ignores reverting.
3. with the remaining probability $1-(a+b)$, randomly jumps to another page according to the uniform distribution.

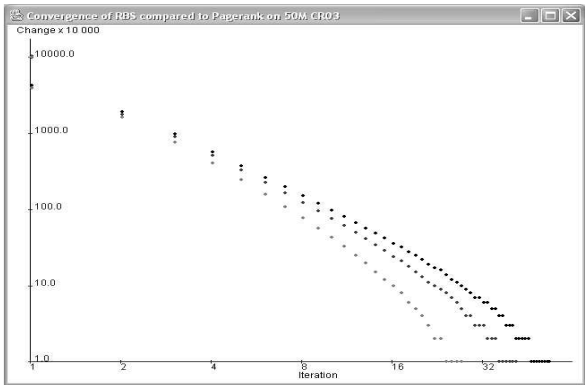

Figure 1: Convergence of RBS algorithm on CR03 compared to PageRank on the same graph. Vertical: change between iterations. Black: RBS, a=0.85,b=0.075; gray: RBS, a=0.8,b=0.1; light-gray: PageRank, decay = 0.15 (for comparison)

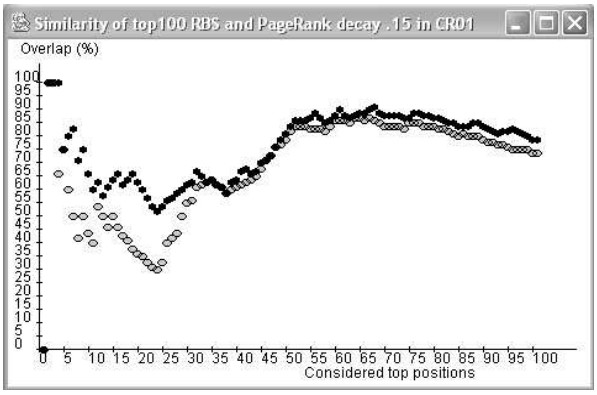

Figure 2: Top 100 results of RBS for two sets of parameters compared to PageRank on CR01. Vertical scale: percentage of common documents among 100 top-ranked documents. Full oval: a=0.85, b=0.085; empty oval: a=0.85, b=0.1 . RBS is compared to PageRank with decay=0.15. Notice that sets of top results are similar to that of PageRank, but there are substantial differences in ordering, especially among the top 30 positions

3.1 Probabilistic Approximation

Although the natural way of implementing the model is via history stacks (as in browsers), this approach would be of prohibitively high time complexity. Instead, we derive a *probabilistic approximation* model which makes fast computation possible. This is a similar approach to that of traditional PageRank with one novelty – for each page x we have to compute the amount of rank score which it gets, due to the back step, from any page y, being linked by x. We call this quantity $R_B(x, y)$, and define it as being proportional to *the conditional probability* of visiting the page x in the previous step assuming that in the current step y is visited:

$$R_B(x, y) = \beta(y)\frac{aR(x)}{outDeg(x)R(y)}, \qquad (1)$$

where $\beta(y)$ is some proportionality factor dependent on y, which is introduced to normalize the probabilities. After some computations, the final equation for the total amount of rank a page x gets through "back-step" flow is:

$$R_B(x) = ab\frac{R(x)}{outDeg(x)} \sum_{y \in OUT(x)} \frac{R(y)}{R_{IN}(y)}, \qquad (2)$$

Where $R_{IN}(y)$ denotes the amount of rank flow which y receives due to the "follow-link flow". Keeping other things similarly as in PageRank computation, we can adapt a power method to compute the resulting ranking vector R efficiently. We call our method RBS algorithm. It is possible to derive form the results obtained in [3], that a vector R exists and is unique under assumption $b < 1/2$. (Because of extreme space limitations see [2] for details). Our implementation of RBS requires twice more memory than PageRank, due to keeping and computing R_{IN}, but each iteration takes very similar time. We have tested our implementation on 80- and 50- million page samples of real Web graph from 2001 and 2003 (called here CR01 and CR03)[1] as well as on synthetic Web graph models. RBS has been convergent for all cases tested up to know, but generally needs more iterations compared to PageRank (fig. 1).

In general, the set of top-ranked documents is similar to that of PageRank, but the order is different (see fig. 2), what is a very important difference for ranking algorithms. Other experiments demonstrated sensitivity of RBS to changes of its parameters[2]. Below we present 20 top-ranked URLs returned by RBS on CR01:

```
Top 20 of RBS for a=0.65, b=0.085, CR01:
1 http://www.yahoo.com/
2 http://messenger.yahoo.com/
3 http://www.tucows.com/
4 http://www.microsoft.com/info/cpyright.htm
5 http://www.domaindirect.com/
6 http://news.tucows.com/
7 http://www.adobe.com/homepage.html
8 http://www.ibm.com/
9 http://www.adobe.com/misc/privacy.html
10 http://www.adobe.com/misc/comments.html
11 http://www.adobe.com/store/main.html
13 http://www.microsoft.com/
14 http://ispcentral.tucows.com/
15 http://home.netscape.com/
16 http://www.microsoft.com/misc/cpyright.htm
17 http://search.internet.com/
18 http://www.adobe.com/products/acrobat/readstep.html
20 http://www.adobe.com/products/main.html
```

Generally, all the top-ranked URLs represent important pages (note that they represent Web from 2001) – similarly to PageRank – but are ordered in a different way. Furthermore, we observed experimentally that RBS, compared to PageRank, seems to have capability of "appreciating" important pages, even if they are deeper in URL hierarchy and slightly punishes "hermetic" pages [3], even if they are regarded as important themselves. These interesting features of RBS will be further analyzed in future.

4. REFERENCES

[1] L.Page, S.Brin, R.Motwani, and T.Winograd. The pagerank citation ranking: Bringing order to the web. In *Stanford Digital Libraries Working Paper*, 1998.

[2] M.Sydow. Link Analysis of the Web Graph. Measurements, Models and Algorithms for Web Information Retrieval. (PhD Thesis) *Polish Academy of Sciences. Institute of Computer Science*, to appear in 2004.

[3] R.Fagin, A.Karlin, J.Kleinberg, P.Raghavan, S.Rajagopalan, R.Rubinfeld, M.Sudan, and A.Tomkins. Random walks with back buttons. *Annals of Applied Probability*, 11(3), 2001.

[1]Thanks are due to G.Wesley and T.Haveliwala, Stanford Web Base for CR01 and CR03 Web topology data files.

[2]They are not reported here due to space limitations

[3]Pages without links to other important sites