Counting Ancestors to Estimate Authority

Jian Wang and Brian D. Davison
Department of Computer Science and Engineering
Lehigh University
Bethlehem, PA 18015 USA
{jiw307, davison}@cse.lehigh.edu

ABSTRACT

The AncestorRank algorithm calculates an authority score by using just one characteristic of the web graph—the number of ancestors per node. For scalability, we estimate the number of ancestors by using a probabilistic counting algorithm. We also consider the case in which ancestors which are closer to the node have more influence than those farther from the node. Thus we further apply a decay factor δ on the contributions from successively earlier ancestors. The resulting authority score is used in combination with a content-based ranking algorithm. Our experiments show that as long as δ is in the range of [0.1, 0.9], AncestorRank can greatly improve BM25 performance, and in our experiments is often better than PageRank.

Categories and Subject Descriptors

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

General Terms

Algorithms

Keywords

Probabilistic counting, link analysis, PageRank

1. INTRODUCTION

A web page's authority score is usually computed by analyzing the link structure of web graph. In this paper, we define `ancestors` to be the set of unique nodes with one or more directed paths to the target node. We analyzed the link structure based on the following assumptions:

- Nodes with more ancestors are more important than those with fewer ancestors.
- Nearby ancestors should have more influence than those further away.

These give us the intuition to calculate the decayed number of ancestors, that is, the decayed sum of all ancestors. For every node in the graph, we combine this information with BM25 result and our experiments show that AncestorRank can boost BM25 performance greatly. The intuition of AncestorRank is not unlike PageRank [2], yet is more straightforward and in our experiments has performance comparable or better than that of PageRank.

SIGIR'09, July 19–23, 2009, Boston, Massachusetts, USA.
ACM 978-1-60558-483-6/09/07.

2. BACKGROUND

PageRank is a popular and well-studied link-based algorithm used to estimate authority. The intuition behind AncestorRank is similar to that of PageRank, except for two major differences. In our algorithm, a node has the same influence on its children, regardless of how many children the node has. In addition, the scores produced by AncestorRank do not form a probability distribution and thus are not intended to be representative of the actions of any kind of surfer.

A naive algorithm to count exactly the number of ancestors would be expensive: $O(n^2)$. Simpler approaches that would be faster might over count nodes because of multiple paths, or would require significant storage resources. In order to retain scalability for large graphs but not to over count the number of ancestors, we turn to a probabilistic counting approach, originally proposed by Flajolet and Martin [4]. A more recent refinement was offered by Becchetti et al. [1]. It is simpler and more accurate when the distance under consideration is small. It can also be viewed as a generalization of the ANF algorithm [5]. Their definition of `supporter` is same as our definition of `ancestor`. Becchetti et al. also provide an adaptive estimation process to select the best probabilistic representation. Their search process updates ϵ, the probability of initializing a bit in a node's bit vector to 1, by multiplying it by γ (called MULT in their work) after every iteration, until the representation can accurately estimate the value for the current node.

3. USING ANCESTOR INFORMATION

We estimate the number of ancestors from distance 1 to i, where for any distance greater than i, there are no additional ancestors for any node in the webgraph. After that, we apply the decay factor δ to count the decayed number of ancestors for every node.

$$N(x) = \delta^0 N_1(x) + \delta^1 (N_2(x) - N_1(x)) + \ldots + \delta^{i-1}(N_i(x) - N_{i-1}(x))$$

δ is the decay factor, $0 \leq \delta \leq 1$. When δ is 0.0, we simply count the number of inlinks (parents). When δ is 1.0, we count the number of ancestors without decay. $N_i(x)$ is the sum of ancestor counts distance 1 to i. $N_i(x) - N_{i-1}(x)$ is the number of new ancestors appearing in distance i.

After computing the decayed number of ancestors, we order all nodes by this value, and linearly combine each node's rank with the rank of the query-specific IR score based on content. The IR score that we choose to combine is the OKAPI BM25 [8] weighting function, and the parameters are set to be the same as Cai et al. [3].

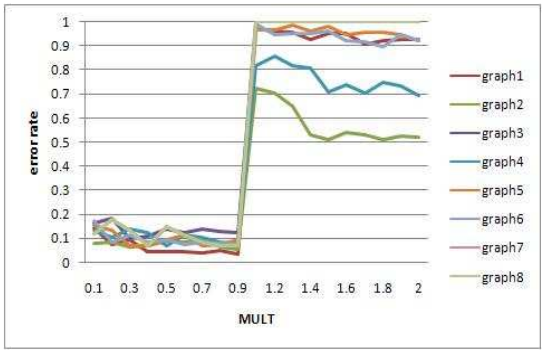

Figure 1: Error rate of probabilistic counting on 8 test graphs.

4. EXPERIMENTAL SETUP

We use the TREC (http://trec.nist.gov/) GOV collection, which contains a 1.25M web page crawl of the .gov domain from 2002. To test ranking algorithms on the GOV corpus, we chose the topic distillation task in the web track of TREC 2003, which has 50 queries.

Since we combine our ranked list with BM25, we use it as the baseline. In addition, we chose some popular authority ranking algorithms with which we can compare, including PageRank and Global HITS (denoted as GHITS) in which the hub and authority calculation of HITS [7] is applied to the full web graph, not just a query-specific subgraph. All algorithms have been linearly combined with BM25.

In order to compare our results with existing work, we evaluated the ranking algorithms using Precision@10, MAP (Mean Average Precision), RPrec, and NDCG@10 [6]. For every approach, we tune the combining parameter to get the best P@10 and output this result as the final performance.

In order to estimate the number of ancestors, we utilized Becchetti et al.'s adaptive probabilistic counting approach. We set the bit vector representation to be 64 bits and tuned the γ factor in the adaptive probabilistic counting algorithm. In every iteration, the current $\epsilon = last\_\epsilon \times \gamma$. In our experiments, we find that the γ factor can have a huge effect on the error rate, compared to the real number of ancestors. Figure 1 shows the average relative error rates for eight sample test graphs with different sizes, based on γ increasing by 0.1 from 0.1 to 2.0.

Generally speaking, the error rate is much smaller (3-17%) when γ is less than 1. In each test graph, the error rate is minimized when γ is 0.9 since it covers more possible values for ϵ. Yet it also requires more time to complete the whole process since more values are tested. Cognizant of this trade-off, we selected γ to be 0.5 in our experiments on the TREC 2003 web graph.

5. EXPERIMENTAL RESULTS

Table 1 shows the final performance for some sample δ increasing from 0.0 to 1.0. As long as the δ is in the range of [0.1, 0.9], most metrics show better performance of AncestorRank over BM25, PageRank and GHITS. A student's t-test confirms that the improvements shown over BM25 and PageRank for MAP, Rprec and NDCG are statistically significant ($p = .05$ for BM25 and $p = .10$ for PageRank).

For AncestorRank, we implemented the probabilistic counting algorithm to count from distance = 1 to distance = i when there are no additional ancestors in distance greater than i. Ideally we can increase the decayed sum with $\delta^{j-1}(N_j(x) - N_{j-1}(x))$ after we count the ancestors for distances j and $j-1$. The estimation for current node x stops when $N_j(x) = N_{j-1}(x)$. It costs $O(kN)$ memory in total, where k is the length of the bit vector representation and N is the number of nodes in the graph. In the adaptive algorithm with $\gamma < 1$, for every node, the appropriate ϵ in the current distance should be smaller or equal to the ϵ that is appropriate for the previous distance. Thus, for distance j, ϵ would decrease from approximating $\frac{1}{N_{max}(j-1)}$ to approximating $\frac{1}{N_{max}(j)}$ by applying γ as needed. $N_{max}(j)$ is the maximum of number of ancestors for any node at distance j, which is generally much smaller than N. $N_{max}(0)$ corresponds to the starting ϵ which we set to be 0.5. In addition, every time we want to use a new ϵ to count the number of ancestors for distance j, we need to go through the graph j times during that iteration. To sum up, for distance j, the number of times to read the whole graph is $\lceil j log_{\frac{1}{\gamma}} \frac{N_{max}(j)}{N_{max}(j-1)} \rceil$ when $\gamma < 1$. Thus AncestorRank is tractable, unlike a naive counting algorithm with $O(N^2)$ time complexity.

Table 1: AncestorRank and baselines performance

algorithm	P@10	MAP	RPrec	NDCG@10
AR(δ=0.0)	0.128	0.155	0.161	0.211
AR(δ=0.5)	0.134	0.177	0.171	0.243
AR(δ=0.7)	0.134	**0.178**	0.169	**0.245**
AR(δ=0.8)	0.134	0.171	**0.176**	0.240
AR(δ=1.0)	0.124	0.150	0.146	0.201
BM25	0.120	0.149	0.140	0.199
PageRank	**0.138**	0.153	0.153	0.218
GHITS	0.136	0.143	0.154	0.204

6. CONCLUSION

AncestorRank estimates authority by summing for each page its decayed number of unique ancestors. While this algorithm is conceptually simple, it appears to have comparable or better performance than PageRank. Since counting exactly the decayed number of ancestors is expensive, we leverage a probabilistic counting algorithm which provides fairly good estimates and is scalable in practice.

Acknowledgments

This work was supported in part by a grant from the National Science Foundation under award IIS-0545875. We also thank Xinlei Wu for some preliminary aspects of this work.

7. REFERENCES

[1] L. Becchetti, C. Castillo, D. Donato, R. Baeza-Yates, and S. Leonardi. Link analysis for web spam detection. *ACM Trans. Web*, 2(1):1–42, 2008.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proc. of the 7th International Conference on World Wide Web (WWW)*, pages 107–117. Elsevier, 1998.

[3] D. Cai, X. He, J.-R. Wen, and W.-Y. Ma. Block-level link analysis. In *Proc. of the 27th Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 440–447, July 2004.

[4] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *J. Comput. Syst. Sci.*, 31(2):182–209, 1985.

[5] P. B. Gibbons, C. Faloutsos, and C. R. Palmer. ANF: A fast and scalable tool for data mining in massive graphs. In *SIGKDD*, pages 81–90. ACM Press, 2002.

[6] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

[7] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[8] S. E. Robertson. Overview of the OKAPI projects. *Journal of Documentation*, 53(1):3–7, 1997.