The Power of Local Information in PageRank

Marco Bressan
TAO, INRIA-CNRS, LRI
Université Paris-Sud
Gif-sur-Yvette, France
marco.bressan@inria.fr

Enoch Peserico
Dip. Ing. Informazione
Università di Padova
Padova, Italy
enoch@dei.unipd.it

Luca Pretto
Dip. Ing. Industriale
Università di Padova
Padova, Italy
pretto@dei.unipd.it

ABSTRACT

Can one assess, by visiting only a small portion of a graph, if a given node has a significantly higher PageRank score than another? We show that the answer strongly depends on the interplay between the required correctness guarantees (is one willing to accept a small probability of error?) and the graph exploration model (can one only visit parents and children of already visited nodes?).

Categories and Subject Descriptors: F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems – Computations on discrete structures

General Terms: Algorithms, Theory

Keywords: PageRank; local computation; graph ranking

1. LOCAL PAGERANK

PageRank [8], originally devised to measure web page centrality, has become the reference algorithm for ranking nodes of a graph in many application domains (to name but a few, web crawling, bioinformatics, natural language processing, and social networks analysis – see [6]). On a graph G of n nodes, the PageRank score of a node v is defined as:

$$P(v) = \frac{1-\alpha}{n} + \alpha \sum_{u:u\to v} \frac{P(u)}{\text{outdegree}(u)} \ , \qquad \sum_{v\in G} P(v) = 1$$

where $u \to v$ denotes that u is a parent of (i.e. has an outgoing link to) v and the *damping factor* α is some constant in the interval $(0, 1)$. Thus, $\frac{1-\alpha}{n} \leq P(v) \leq 1$.

PageRank is often employed to analyse graphs, such as web or social networks, so large that they are difficult to store, "snapshot", or even access in their entirety (e.g. for privacy limitations [10]). Thus, considerable effort [2, 3, 4, 5, 9] has been applied to minimize the portion of the graph one must explore to compute PageRank (and other global graph properties) for a restricted set of target nodes.

A typical model for this *local PageRank* problem involves a *link server* [3] that responds to any query about a given node with a list of all nodes pointed by it, and a list of all nodes pointing to it (like Google's `link:` does). The goal is to minimize the number of queries necessary to determine which of two given nodes has the greater PageRank score (allowing an arbitrary answer if the two scores are within a factor $1 + \epsilon$). Alternative models allow instead *jump* and (*random*)*crawl* queries [4, 5], that return respectively a random node of the graph, and a (random) subset of its neighbours.

This work analyses the impact, on the number of queries necessary for local PageRank computations, of the interplay between exploration model and correctness guarantees; and in particular between the requirements that the exploration be "local" (i.e. visiting only parents and children of already visited nodes) and that the algorithm always return a correct answer (rather than allowing for a small probability of error).

2. DETERMINISTIC AND LAS VEGAS

Our technical report [6] proves that any local PageRank algorithm that is always correct may in general have to visit all but a vanishingly small fraction of a graph. This includes deterministic algorithms as well as Las Vegas randomized algorithms (that guarantee a correct answer, but not a bounded number of queries); it does not include Monte Carlo randomized algorithms (that may provide an incorrect answer with positive probability). Our negative result holds even if comparing the two top ranking nodes, even if the ratio between their PageRank scores is large (regardless of the scores' values). Crucially, it holds for *every* exploration model providing any combination of *jump*, (*random*)*crawl* and *link server* queries – and, more in general, for any exploration model in the literature.

The cornerstone of this very general result is the notion of *ranking subgraph* for a given set of nodes – intuitively, a subgraph with enough information to determine their relative ranking. We show that any algorithm that is always correct always explores a ranking subgraph for the nodes it compares, and that in some cases each such subgraph almost coincides with the entire graph. This yields:

THEOREM 1. *Choose a damping factor $\alpha \in (0, 1)$, a score function $\Theta(1/n) \leq p(n) \leq \Theta(1)$, and a separation function $\Theta(1/n) \leq \epsilon(n) \leq \Theta(1)$. There exists a graph G of arbitrarily large size n containing nodes u and v such that:*

1. *$P(u) = \Theta(p(n)), \quad P(v)/P(u) \geq 1 + \Theta(\epsilon(n))$*
2. *u, v are the top ranking nodes in G*
3. *to decide if $P(v) > P(u)$, any deterministic local ranking algorithm and (any execution of) any Las Vegas local ranking algorithm need $n(1-O(p(n)\epsilon(n)))$ queries*

Theorem 1 extends to a much more general exploration model the $\Omega(n)$ queries lower bound for deterministic algorithms limited to make *link server* queries [7]. It extends

WWW 2013 Companion, May 13–17, 2013, Rio de Janeiro, Brazil.
ACM 978-1-4503-2038-2/13/05.

and improves the analogous $\Omega(\sqrt{n})$ lower bound for Las Vegas algorithms [7]. Furthermore, it does so not only for the problem of the relative ranking of two nodes, but also for the problem of computing within a factor $\sqrt{1+\epsilon}$ their PageRank scores [3] (obviously the solution to the latter yields a solution to the former). And it shows that, as the absolute score separation $p(n)\epsilon(n)$ decreases below $\Theta(1)$, any algorithm guaranteeing a correct output must explore *the entire graph, save possibly a vanishingly small portion.*

3. MONTE CARLO

Unlike deterministic and Las Vegas algorithms, Monte Carlo algorithms do not admit a characterization in terms of ranking graphs, and are not subject to the corresponding lower bounds. In fact, in the case of Monte Carlo algorithms the exploration model *does* make a difference. If the exploration must be *local*, in the sense that one may only visit parents or children of already visited nodes (in addition to the target nodes), then essentially the same bounds of Theorem 1 hold – albeit through a different proof technique:

THEOREM 2. *Choose a damping factor $\alpha \in (0,1)$, a separation $\epsilon > 0$, and a score function $\Theta(1/n) \leq p(n) \leq \Theta(1)$. For any Monte Carlo local ranking algorithm with confidence $\frac{1}{2}+\delta$ that can perform only* (random)crawl *and* link server *queries there exists a graph G of arbitrarily large size n containing nodes u and v such that:*

1. *$P(u), P(v) = \Theta(p(n)), \quad P(v) \approx (1+\epsilon)P(u)$*
2. *$\Omega(\delta n)$ queries are needed to decide if $P(v) > P(u)$*

On the other hand, if the exploration can be non-local, a Monte Carlo algorithm can achieve considerably better performance. This is true even if one is limited to *jump* and *randomcrawl* queries, where the only non-local exploration option is visiting a graph node chosen uniformly at random. Consider the following simple algorithm (which can be made oblivious to p [6]; the meaning of p and η is explained below):

Algorithm 1 *SampleRank*$(G, u, v, p, 1-\eta)$

Perform $8\log(\frac{8}{\eta})\frac{1}{p}(\frac{1+\epsilon}{\epsilon})^2$ random walks *(starting from a node returned by a* jump *query and at each step ending with independent probability $1-\alpha$)*
$\hat{P}(u), \hat{P}(v) \leftarrow$ fraction of random walks ending in u, v
return the ranking of u, v induced by $\hat{P}$

One can prove:

THEOREM 3. *Consider u, v such that $P(v) \geq (1+\epsilon)P(u)$ and $P(u) \geq p$. A call to SampleRank$(G, u, v, p, 1-\eta)$ has probability at least $1-\eta$ of providing their correct ranking while performing at most $\frac{14}{1-\alpha}\log(\frac{8}{\eta})\frac{1}{p}(\frac{1+\epsilon}{\epsilon})^2$ queries.*

The bounds of Theorem 3 are asymptotically tight except for very small PageRank scores, even for algorithms that can make *crawl* and *link server* queries in addition to *jump* and *randomcrawl* ones. More formally one can prove:

THEOREM 4. *Choose a damping factor $\alpha \in (0,1)$, a separation $\epsilon > 0$, and a score function $\Theta(1/n) \leq p(n) \leq \Theta(1)$. For any Monte Carlo local ranking algorithm with confidence $\frac{1}{2}+\delta$ that can perform* jump, (random)crawl *and* link server *queries there exists a graph G of arbitrarily large size n containing nodes u and v such that:*

1. *$P(u), P(v) = \Theta(p(n)), \quad P(v) \approx (1+\epsilon)P(u)$*
2. *$\Omega(\delta \cdot \min(1/p(n), n^{\frac{2}{3}}))$ queries are needed to decide if $P(v) > P(u)$*

This is the first analysis of the impact of the exploration model on Monte Carlo algorithms. Theorem 2 improves to $\Omega(n)$ the $\Omega(\sqrt{n})$ lower bounds of [3, 7]. Theorem 4 gives the first non-trivial lower bounds on the performance of algorithms using *jump* and *link server* queries for local PageRank computations, which almost (but not quite) match the bounds holding under the more restrictive *jump* and *randomcrawl* model [6].

4. CONCLUSIONS

Two ingredients are necessary for efficient local PageRank computations: *exploring the graph non-locally* and *accepting a small probability of error*. If either one is missing, visiting almost the entire graph may be necessary to just determine which of two nodes has the higher PageRank score. If both ingredients are present, then a very simple algorithm allows efficient computation of PageRank scores and rankings using, as the sole non-local operation, a query for a graph node chosen uniformly at random (and this is optimal even among the class of algorithms employing far more powerful exploration primitives, save possibly for very small PageRank scores).

All our results hold both for computing PageRank scores, and for computing PageRank rankings; the two problems appear then essentially equivalent.

In practical terms, our results imply that it is essentially impossible to guarantee correctness when attempting to compute PageRank scores precisely, or to separate nodes with relatively close PageRank scores, on any graph (like the web or many social networks) that evolves relatively quickly over time [1] and/or sports even small inaccessible portions [10]. Another practical consequence is that providing random web pages can be an extremely useful service – possibly more than providing the set of pages pointing to a given page.

5. ACKNOWLEDGEMENTS

This work was supported in part by Augure, and by Proj. AACSE and postdoc fellowships from Univ. Padova.

6. REFERENCES

[1] A. Anagnostopoulos, R. Kumar, M. Mahdian, E. Upfal, and F. Vandin. Algorithms on evolving graphs. *ITCS'12*.
[2] R. Andersen, C. Borgs, J. T. Chayes, J. E. Hopcroft, V. S. Mirrokni, and S.-H. Teng. Local computation of PageRank contributions. *Internet Mathematics*, 5(1), 2008.
[3] Z. Bar-Yossef and L.-T. Mashiach. Local approximation of PageRank and reverse PageRank. *CIKM'08*.
[4] C. Borgs, M. Brautbar, J. T. Chayes, and S.-H. Teng. A sublinear time algorithm for PageRank computations. *WAW'12*.
[5] M. Brautbar and M. Kearns. Local algorithms for finding interesting individuals in large networks. *ICS'10*.
[6] M. Bressan, E. Peserico, and L. Pretto. The power of local information in PageRank. Univ. Padova technical report, 2013. `http://www.dei.unipd.it/~pretto/localrank.pdf`.
[7] M. Bressan and L. Pretto. Local computation of PageRank: the ranking side. *CIKM'11*.
[8] S. Brin and L. Page. The anatomy of a large scale hypertextual Web search engine. *WWW'98*.
[9] Y.-Y. Chen, Q. Gan, and T. Suel. Local methods for estimating PageRank values. *CIKM'04*.
[10] R. Gross and A. Acquisti. Information revelation and privacy in online social networks. *WPES'05*.