# Do Your Worst to Make the Best: Paradoxical Effects in PageRank Incremental Computations*

(Extended Abstract)

Paolo Boldi† Massimo Santini‡ Sebastiano Vigna*

**Abstract**

Deciding which kind of visit accumulates high-quality pages more quickly is one of the most often debated issue in the design of web crawlers. It is known that breadth-first visits work well, as they tend to discover pages with high PageRank early on in the crawl. Indeed, this visit order is much better than depth first, which is in turn even worse than a random visit; nevertheless, breadth-first can be superseded using an omniscient visit that chooses, at every step, the node of highest PageRank in the frontier.

This paper discusses a related, and previously overlooked, measure of effectivity for crawl strategies: whether the graph obtained after a partial visit is in some sense representative of the underlying web graph as far as the computation of PageRank is concerned. More precisely, we are interested in determining how rapidly the computation of PageRank over the visited subgraph yields relative ranks that agree with the ones the nodes have in the complete graph; ranks are compared using Kendall's $\tau$.

We describe a number of large-scale experiments that show the following paradoxical effect: visits that gather PageRank more quickly (e.g., highest-quality-first) are also those that tend to miscalculate PageRank. Finally, we perform the same kind of experimental analysis on some synthetic random graphs, generated using well-known web-graph models: the results are almost opposite to those obtained on real web graphs.

## 1 Introduction

The search for the better crawling strategy (i.e., a strategy that gathers early pages of high quality) is by now an almost old research topic (see, e.g., [8]). Being able to collect quickly high-quality pages is one of the major design goals of a crawler; this issue is particularly important, because, as noted in [11], *even after crawling well over a billion pages, the number of uncrawled pages still far exceeds the number of crawled pages.*[1]

Usually, a basic measure of quality is PageRank [28], in one of its many variants. Hence, as a first step we can compare two strategies by looking at how fast the cumulative PageRank (i.e., the sum of the PageRanks of all the visited pages up to a certain point) grows over time. Of course, this comparison can only be performed after the end of the crawl, because one needs the whole graph to compute PageRank.

We list a number of classical visit strategies:

- *Depth-first* order: the crawler chooses the next page as the *last* that was added to the frontier; in other words, the visit proceeds in a LIFO fashion.
- *Random* order: the crawler chooses randomly the next page from the frontier.
- *Breadth-first* order: the crawler chooses the next page as the *first* that was added the frontier; in other words, the visit proceeds in a FIFO fashion.

*This work has been partially supported by MIUR COFIN "Linguaggi formali e automi: metodi, modelli e applicazioni" and by a "Finanziamento per grandi e mega attrezzature scientifiche" of the Università degli Studi di Milano.

†Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, via Comelico 39/41, I-20135 Milano, Italy. {boldi,vigna}@dsi.unimi.it

‡Università di Modena e Reggio Emilia, via Giglioli Valle I-42100 Reggio Emilia, Italy, and Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche, Via Moruzzi 1, I-56010 Pisa, Italy. massimo.santini@iit.cnr.it

[1] In the present paper by "uncrawled page" we are referring only to those pages that actually exist and will be crawled sometimes in the future, did the crawl go on forever.

- *Omniscient* order (or quality-first order): the crawler uses a queue prioritised by PageRank [8]; in other words, it chooses to visit the page with highest quality among the ones in the frontier. This visit is meaningless unless a previous PageRank computation of the entire graph has been performed *before* the visit, but it is useful for comparisons. A variant of this strategy may also be adopted if we have already performed a crawl and so we have the (old) PageRank values of (at least some of the) pages.

Both common sense and experiments (see, in particular, [4]) suggest that the above visits accumulate PageRank in a growingly quicker way. This is to be expected, as the omniscient visit will point immediately to pages of high quality. The fact that breadth-first visit yields high-quality pages was noted in [26].

There is, however, another and also quite relevant problem, which has been previously overlooked in the literature: if we assume that the crawler has no previous knowledge of the web region it has to crawl, it is natural that it will try to detect page quality during the crawl itself, by computing PageRank on the region it has just seen. We would like to know whether doing so it will obtain reasonable results or not. This question is even more urgent than it might appear at first, because we will most probably stop the crawl at a certain point anyway, and use the graph we have crawled to compute PageRank.

To answer this question, of course, we must establish a measure of how good is a PageRank approximation computed on a subgraph. Comparing directly PageRank values would be of course meaningless because of normalisation. However, we can compare two orders:

- the order induced on the subgraph by PageRank (as computed on the subgraph itself);
- the order induced on the whole graph by PageRank; of course, this order must be restricted to the nodes of the subgraph.

In this paper, we use Kendall's $\tau$ as a measure of concordance between the two ranked lists (a similar approach was followed in [22] to motivate the usefulness of a hierarchical algorithm to compute PageRank). The results we obtain are paradoxical: *many strategies that accumulate PageRank quickly explore subgraphs with badly correlated ranks*, and viceversa. Even more interestingly, these behaviours have not been reproduced by random graphs generated with two well-known models.

## 2 Kendall's $\tau$ and Its Computation

There are many correlation indices that can be used to compare orders[2]. One of the most widely used and intuitive is Kendall's $\tau$; this classical nonparametric correlation index has recently received much attention within the web community for its possible applications to rank aggregation [13, 12, 10] and for determining the convergence speed in the computation of PageRank [17]. Kendall's $\tau$ is usually defined as follows[3]:

**Definition 1 ([20], pages 34–36)** Let $r_i, s_i \in \mathbf{R}$ $(i = 1, 2, \ldots, n)$ be two rankings. Given a pair of distinct indices $1 \leq i, j \leq n$, we say that the pair is:

- *concordant* iff $r_i - r_j$ and $s_i - s_j$ are both nonzero and have the same sign;
- *discordant* iff $r_i - r_j$ and $s_i - s_j$ are both nonzero and have opposite signs;
- an *r-tie* iff $r_i - r_j = 0$;
- an *s-tie* iff $s_i - s_j = 0$;
- a *joint tie* iff $r_i - r_j = s_i - s_j = 0$.

Let $C$, $D$, $T_r$, $T_s$, $J$ be the number of concordant pairs, discordant pairs, $r$-ties, $s$-ties and joint ties, respectively; let also $N = n(n-1)/2$. Of course $C + D + T_r + T_s - J = N$. Kendall's $\tau$ of the two rankings is now defined by

$$\tau = \frac{C - D}{\sqrt{(N - T_r)(N - T_s)}}.$$

[2] For a thorough account on this topic, consult for example [18].

[3] There are actually various subtly different definitions of $\tau$, depending on how ties should be treated. The definition we use here is usually referred to as $\tau_b$.

Kendall's $\tau$ is always in the range $[-1, 1]$: $\tau = 1$ happens iff there are no non-joint ties and the two total orders induced by the ranks are the same; $\tau = -1$, conversely, happens iff there are no non-joint ties and the two total orders are opposite of each other; thus, $\tau = 0$ can be interpreted as lack of correlation.

**Knight's Algorithm.** Apparently, evaluating $\tau$ on large data samples is not so common, and there is a remarkable scarcity of literature on the subject of computing $\tau$ in an efficient way. Clearly, the brute-force $O(n^2)$ approach is easy to implement, but inefficient. Knight [21] presented in the sixties an $O(n \log n)$ algorithm for the computation of $\tau$, but the only implementation we are aware of belongs to the SAS system.

Because of the large size of our data, we decided to write a direct, efficient implementation of Knight's algorithm, with a special variant needed to treat ties as required by our definition. This variant was indeed briefly sketched at the end of Knight's original paper, but details were completely omitted[4].

First of all, we sort the indices $\{ 1, 2, \ldots, n \}$ using $r_i$ as first key, and $s_i$ as secondary key. More precisely, we compute a permutation $\pi$ of the indices such that $r_{\pi(i)} \leq_{\pi(j)}$ whenever $i \leq j$ and, moreover, if $r_{\pi(i)} = r_{\pi(j)}$ but $s_{\pi(i)} < s_{\pi(j)}$ then $i < j$.

Once the elements have been sorted, a linear scan is sufficient to compute $T_r$: a maximal sequence of indices $i, i+1, \ldots, j$ such that $r_{\pi(i)} = r_{\pi i+1} = \cdots = r_{\pi(j)}$ determines exactly $\binom{j-i+1}{2}$ $r$-ties. Moreover, with another linear scan, and in a completely analogous way, one can compute $J$ (this time, one looks for maximal intervals where *both* $r$ and $s$ are constant). After the computation of $D$ (described below), the indices will be sorted using $s_i$, so we shall able to compute in a similar way $T_s$.

The knowledge of $D$ is now sufficient to deduce $\tau$; the computation of $D$ is indeed the most interesting part of Knight's algorithm. Remember that all indices are at this point sorted using $r_i$ as first key, and $s_i$ as secondary key. We apply a stable merge sort to all indices using $s_i$ as key. Every time, during a merge, we move an item *forward*, we increase the number of discordant pairs by the number of skipped items. Thus, for instance, passing from the order

$$2, 1, 0, 3$$

to natural order requires first ordering two sublists of length 2

$$1, 2, 0, 3,$$

which increases by 1 the number of discordant pairs, and then moving 0 to the first place, getting to

$$0, 1, 2, 3$$

and bringing the overall number to 3. Note that the same amount can be more easily calculated using a bubble sort: you just need two exchanges to move 0 to the front, and then one to move 1 to its final position (the idea of using bubble sort appears in Kendall's original paper [19]), but you cannot expect to run bubble sort on a graph with 100 million nodes.

# 3 Measuring the Quality of a Page with PageRank

PageRank [7] is one of the best-known methods to measure page quality; it is a static method (in that it does not depend on a specific query but rather it measures the absolute authoritativeness of each page), and it is based purely on the structure of the links, or, if you prefer, on the web graph. As it was recently noticed [11], PageRank is actually a set of ranking methods that depends on some parameters and variants; its most common version can be described as the behaviour of a random surfer walking through the web, and depends on a single parameter $\alpha \in (0, 1)$ (the *dumping factor*).

The random surfer, at each step $t$, is in some node $p_t$ of the graph. The node $p_{t+1}$ where the surfer will be in the next step is chosen as follows:

- if $p_t$ had no outgoing arcs, $p_{t+1}$ is chosen uniformly at random among all nodes;
- otherwise, with probability $\alpha$, one of the arcs going out of $p_t$ is chosen (with uniform distribution), and $p_{t+1}$ is the node where the arc ends; with probability $1 - \alpha$, $p_{t+1}$ is once more chosen uniformly among all nodes.

[4]The complete Java code of our implementation of Knight's algorithm above is available under the GNU General Public License.

The PageRank of a given node $x$ is simply the fraction of time that the random surfer spent in $x$, and will be denoted by $\mathrm{PR}_G(x)$.

Another, maybe more perspicuous, way of defining PageRank is the following. Let $A = (a_{ij})$ be an $n \times n$ matrix ($n$ being the number of nodes in the graph), defined as follows:

- if $i$ has no outgoing arcs, $a_{ij} = 1/n$;
- if $i$ has $d > 0$ outgoing arcs, and $(i, j)$ is an arc, then $a_{ij} = \alpha/d + (1 - \alpha)/n$;
- if $i$ has $d > 0$ outgoing arcs, but $(i, j)$ is not an arc, then $a_{ij} = (1 - \alpha)/n$.

Now $A$ turns out to be an aperiodic and irreducible stochastic matrix, so there is exactly one probability vector $r$ satisfying

$$Ar = r.$$

The PageRank of a node $x$, $\mathrm{PR}_G(x)$, is exactly $r_x$. The computation of PageRank is a rather straightforward task, that can be accomplished with standard tools of linear algebra [14, 17]. The dumping factor $\alpha$ influences both the convergence speed and the results obtained, but it is by now customary to choose $\alpha = 0.85$.

# 4 Using Kendall's $\tau$ to Contrast Crawl Strategies

The rank assigned by PageRank to a page is not important *per se*; it is the relative order of pages with respect to PageRank that is actually interesting for search engines. This observation will guide us in what follows.

Consider the following typical scenario: $G$, the "real" web graph (unknown to the crawler), has $N$ nodes; we perform some crawl of $G$, which determines a certain node order (the visit order) $x_1, x_2, \ldots, x_N$. For each $n = 1, 2, \ldots, N$, let $G_n = G[\{x_1, x_2, \ldots, x_n\}]$ be the subgraph of $G$ induced by the first $n$ visited nodes. In other words, $G_n$ is the graph collected at the $n$-th step of the traversal.

In a real-world crawler, we will most probably stop the crawl at a certain point, say, after $n$ pages have been crawled, typically with $n \ll N$. If you run PageRank on $G_n$ you obtain page rank values that will usually not coincide with the rank values that those pages have in $G$. Even worse, their *relative order* will be different.

Let $\tau_n$ be defined as Kendall's $\tau$ computed on $\mathrm{PR}_{G_n}(x_1)$, $\mathrm{PR}_{G_n}(x_2)$, $\ldots$, $\mathrm{PR}_{G_n}(x_n)$ and $\mathrm{PR}_G(x_1)$, $\mathrm{PR}_G(x_2)$, $\ldots$, $\mathrm{PR}_G(x_n)$. Clearly, $\tau_N = 1$, as at the end of the crawl $G$ is entirely known, but the sequence $\tau_1, \tau_2, \ldots, \tau_N$ (hereafter referred to as the $\tau$ *sequence of the visit*) only depends on the visit order; it is not necessarily monotonic, but its behaviour can used as a measure of how good (or bad) is the chosen strategy with respect to the way PageRank order is approximated during the crawl. The main goal of this paper is to present experimental results comparing $\tau$ sequences produced by different visit strategies. We shall also consider $\tau$ sequences of subgraphs not obtained through visits, and use them for comparison.

Observe that a $\tau$ sequence has nothing to do with the convergence speed of PageRank, but rather with its tolerance to graph modifications, or, if you prefer, with its stability. There is a rich stream of research (see, for example, [24, 27, 1, 3, 23, 25]) concerning the robustness of PageRank with respect to graph modifications (node and/or link perturbation, deletion and insertion). Many authors observed, in particular, that PageRank is quite stable under graph modifications [3], at least when the changing pages do not have high PageRank; even removing a large portion of pages, as many experiments suggest [27], turns out to have small influence on PageRank. Some papers also propose variants of the PageRank algorithm [1] that can be computed adaptively and robustly with respect to graph evolution.

However, most of these studies (with the exception of [25]) measure the stability of PageRank by computing the distance (usually, under $L^1$ or $L^2$ norm) between the PageRank vectors. However interesting this metric might be, it is not directly related to our measure. One might indeed object that $\tau$ is not a completely fair way to analyse PageRank computation: after all, two PageRank vectors may turn out to be poorly correlated only because of small value fluctuations in pages with low PageRank. Otherwise said, since $\tau$ is not continuous, small perturbations in the PageRank vector may have disastrous effects.

Nevertheless, the real-world usage of PageRank is, by its very nature, discontinuous—PageRank is essentially used only to rank pages, its real numerical value being irrelevant. Moreover, even if fluctuations may only concern low-ranked pages, thus not affecting the mutual order of the top nodes, this fact should not be underestimated; if a query is satisfied only by a set of pages of low rank, their relative order will be for the user as worthy as the order of the top nodes. The latter effect is noticeable not only for some esoteric queries, but typically also for relevant

queries of interest only to a small community: you will not expect that a query like *group torsion* or *analog editing* is satisfied by pages with high PageRank, but the relative order of the pages that satisfy those queries is certainly meaningful.

# 5 Experimental Setup

**Graphs.** We based our experiments on four web graphs:

- a 41,291,594-nodes snapshot of the Italian domain `.it`, performed with UbiCrawler [4] (later on referred to as the *Italian graph*);
- a 118,142,155-nodes graph obtained from the 2001 crawl performed by the WebBase crawler [15] (later on referred to as the *WebBase graph*);
- a 10,000,000-nodes synthetic graph generated using the Copying Model proposed in [22], with suitable parameters and rewiring (later on referred to as the *copying-model graph*);
- a 10,000,000-nodes synthetic graph generated using the Evolving Network Model proposed in [2], with suitable parameters and rewiring (later on referred to as the *evolving-model graph*).

The two synthetic graphs are much smaller than the others, because by their random nature there would be no gain in using a larger node set.

For each graph, we computed the strongly-connected components, and discarded the nodes that were not reachable from the giant component: we decided to do so because otherwise a single visit would not collect the whole graph, and the visit order would not depend only on the visit strategy and on a single starting node.

The resulting graph sizes were 41,291,594 for the Italian graph, 95,688,816 for the WebBase graph, 4,863,948 for the copying-model graph and 5,159,894 for the evolving-model graph.

**Visit seeds.** For each of the four graphs produced as above, we performed visits starting from three different seeds (a.k.a. starting URLs). We decided to perform visits from the highest-ranked, lowest-ranked and median-ranked node within the giant component (choosing seeds within the giant component guarantees that the whole graph will be collected at the end of the visit).

**Visit strategies.** For each of the graphs and every visit seed, we performed four different kinds of visits: `BF` (breadth-first traversal), `DF` (depth-first traversal), `RF` (random-first traversal: the next node to be visited is chosen at random from the frontier), `QF` (quality-first traversal: the next node to be visited is the one having highest PageRank in the frontier; here, by PageRank we mean PageRank in the real graph, not in the subgraph), and `IQF` (inverse-quality-first traversal: in this case, we choose instead the node having the *lowest* PageRank in the frontier).

**Sampling.** Computing PageRank and Kendall's $\tau$ on a large graph is a heavy computation task. Due to the large size of our data, and to the large number of different visits, we decided to spread logarithmically our sample points, so to avoid sampling too many times very large graphs. In general, samples are multiplicatively spaced at least by $\sqrt{2}$, albeit in several cases we used a thicker sampling. Note also that when the subgraph sizes exceed half of the original graph size, all curves become smooth, so using denser samples in that region is probably useless.

**Computational time.** We performed our PageRank and $\tau$ computation on five dual-processor PCs and two multiprocessor servers (the latter were essential in handling the largest samples). Overall, we used about 1600 hours of CPU user time.

**Tools.** All our tests were performed using the WebGraph framework [6, 5]. Both the Italian and the WebBase graphs are available at `http://webgraph-data.dsi.unimi.it/` (recall, however, that we only used the portion of WebBase that can be reached from the giant component). The two synthetic graphs were produced using COSIN [9].

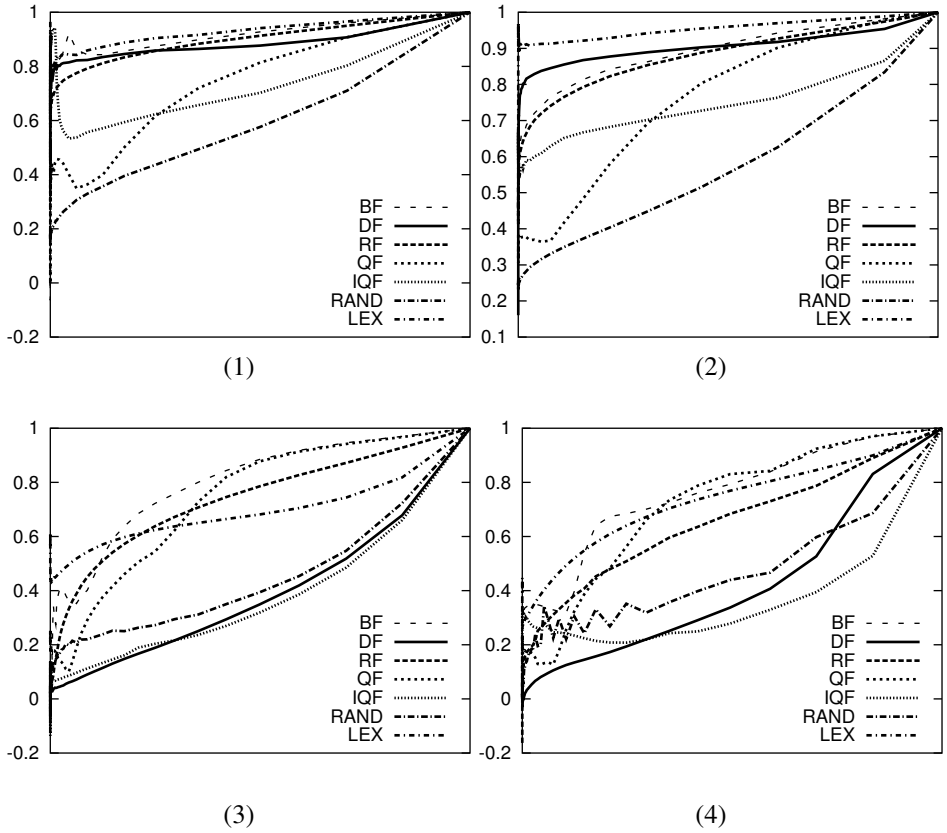


Figure 1: $\tau$ sequences for (1) the Italian graph, (2) the WebBase graph, (3) the copying-model graph and (4) the evolving-model graph, starting from the highest-ranked node in the giant component. See also Figure 3.

# 6 Experimental Results

**Comparing $\tau$ sequences.** Figure 1 represents the $\tau$ sequences obtained during the visits of the four graphs, starting from the node with highest PageRank within the giant component (in the Appendix, you can find similar graphs for other seeds). In each graph, we also plotted the $\tau$ sequence `RAND` corresponding to a random node order (note that this is quite different from a random-first traversal, because it is not a graph visit); additionally, the diagrams contain an extra line, called `LEX`, whose meaning will be explained in the next section. The horizontal axis represents time (or, more precisely, the number of nodes that have been visited), whereas the vertical axis represents the value of $\tau$.

**Cumulative PageRank.** For comparison, we computed (Figure 2) the cumulative PageRank (the sum of all PageRanks of the pages collected so far) during the visits of the Italian graph.

# 7 Interpretation of the Experimental Results

**Cumulative PageRank.** The graph of cumulative PageRank (Figure 2 and Figure 4) confirms other experiments reported by the literature: if you want to collect pages with high PageRank, you should use a breadth-first (`BF`) traversal; this strategy is overcome only by the omniscient quality-first `QF`. Obviously, using an inverse-quality-first traversal (`IQF`) produces the poorest results. Also, depth-first `DF` is only marginally better than `IQF`, whereas, a bit surprisingly, `RF` is very good, even though worst than `BF`. Note that these results, most of which are well-known or folklore, do not appear to be much influenced by the seed.

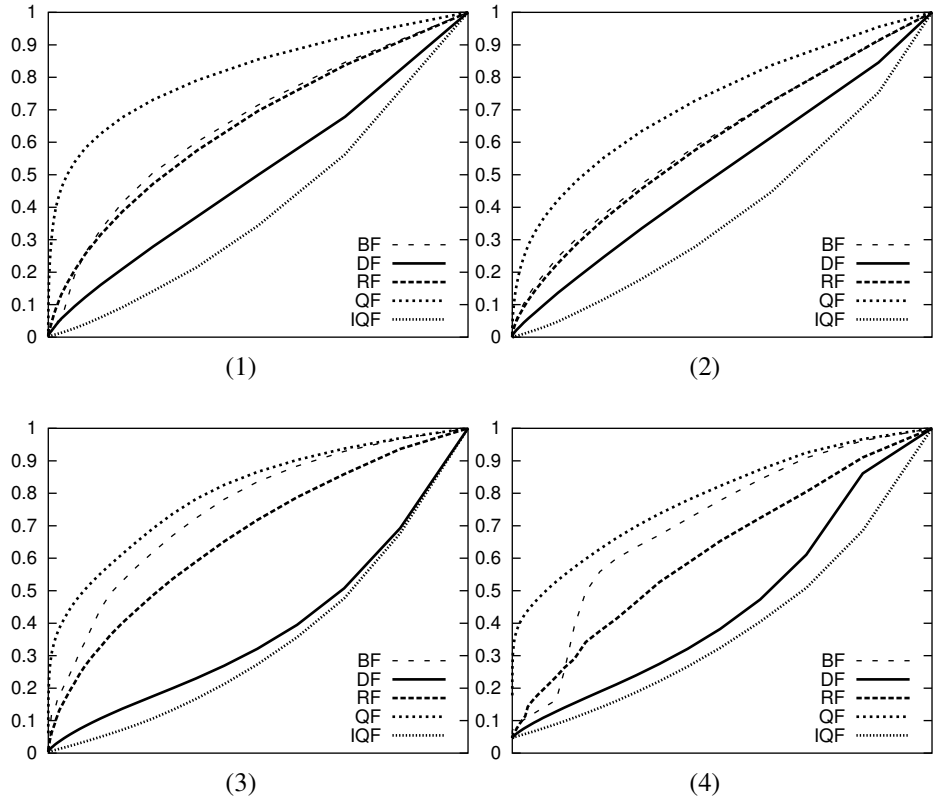


Figure 2: Cumulative PageRank obtained during a visit of (1) the Italian graph, (2) the WebBase graph, (3) the copying-model graph and (4) the evolving-model graph, starting from the highest-ranked node in the giant component.

**PageRank: locality and collective nature.** We now turn to the study of the $\tau$ sequences (Figure 1). Looking at the diagrams of the Italian and of the WebBase graphs, we observe the following facts.

1. Even though $\tau$ sequences do not increase monotonically, they anyway tend to grow quite steadily, after a brief chaotic transient.

2. DF and BF are, in most cases, comparable; often, DF is even better. Moreover, they are both quite close to RF. Note that these behaviours are in sharp contrast with the case of cumulative PageRank, where BF was far better than DF.

3. QF, which used to be the best strategy for cumulative PageRank, is now by far the worst possible strategy. Even IQF is better!

4. All the visit strategies considered are anyway more $\tau$-effective than collecting nodes at random: the line of RAND is constantly below all other lines, in all graphs.

Apparently, the rôles of visit strategies are now completely scrambled. Note that the situation is rather robust: it is the same for all seeds (see Figure 3) and for both graphs, and we indeed tested it against other, smaller real-world examples (not presented here) obtaining quite similar behaviours.

To try to understand the very reasons behind the observed facts, it might be helpful to look at the diagrams in logarithmic (horizontal) scale (not shown here). Under this scale, one observes that all visits present an initial burst that is absent in RAND, whose growth is quite steady. Our explanation for this fact is that, at the beginning of a visit, the crawl tends to remain confined within a single website (or a set of websites).

Indeed, a recent work [16] makes an analysis quite similar to ours with a different purpose: to prove that PageRank computation can be made more efficient by computing local PageRanks first. The authors motivate their algorithm by showing that local PageRanks agree when they are computed on the local subgraph instead of the whole graph, and they use (a version of) Kendall's $\tau$ to measure agreement (they perform their experiments on a small 683,500 pages snapshot Stanford/Berkeley site).

Essentially, inside a single site PageRank depends heavily on the local links. This phenomenon, that we might call *PageRank locality*, explains why any visit performs better than a random selection of nodes.

With the aim of obtaining an empirical confirmation for our conjecture of PageRank locality, we tried another experiment. We computed the $\tau$ sequence of subsets of nodes collected in lexicographic order; in other words, we started with the first URL (in lexicographic order), and proceeded collecting URLs in this way. The resulting $\tau$ sequences are shown in the diagram as LEX: since LEX is a most local way of collecting nodes, we expect the $\tau$ sequence to be distinctly good. And, indeed, LEX lies always above all other lines.

Locality can also explain the success of DF (which beats BF in the first half of the WebBase graph). Depth-first visits tend to wander a while in a site, and then, as soon as a cross-site link is chosen, jump to another site, wander a while, and so on. This behaviour is closer to LEX than to a breadth-first visit.

However, locality cannot explain other phenomena, such as how an inverse quality visit can beat for a long time an omniscient visit. We believe that the paradoxical behaviour of priority-based visits can only be explained by the *collective nature* of PageRank. To have a good $\tau$-correlation with PageRank, besides being local, a subgraph must be also representative of the collection of pages, that is, *sufficiently random* (with respect to PageRank ordering). Said otherwise, *high authoritativeness cannot be achieved without the help of minor pages*. Selecting blindly the most authoritative pages leads to a world where authority is subverted[5].

**Random models.** The graphs for the two chosen random models and are quite disappointing. The PageRank-cumulation behaviour (Figure 2) follows in a relatively close way what happens for real graphs (albeit the models are somehow too good to be true—breadth-first visits beat by far and large random visits, which is not true on real graphs). When, however, we turn to $\tau$-sequences (Figure 1), we find marked differences. For example, DF works much more poorly than it does in real web graphs, whereas QF seems very powerful while it is not. A partial explanation for this is that the chosen models do not take locality into account, hence failing to show the behaviours discussed above. Certainly there is a lot of room for improvement.

[5] Again, even the opposite choice—selecting the pages with lowest authority—works sometimes better, which should lead us to reconsider gossiping versus good newspapers as a mean for selecting authoritative sources of information.

# 8 Conclusions and Further Work

Our results, as the title suggests, are somehow paradoxical: strategies that work quite well when you want to accumulate pages with high quality in a short time tend to behave rather poorly when you try to approximate PageRank on a partial crawl, and viceversa. Our explanation for this paradox is PageRank locality. Of course, the results presented in this paper are still very preliminary, and lack any theoretical investigation.

Finally, we address some issues about our methodology and some possible further direction.

- *Treatment of dangling links.* In this paper, we computed the PageRank of a subgraph ignoring the fact that some nodes of the subgraph would contain many outlinks that are not there simply because our crawl is partial. Some authors recently suggested [11] to modify the definition of PageRank to take this phenomenon into account. It would be interesting to know whether their version of PageRank makes things better or not.

- *Comparing only top nodes.* Another possible direction is considering a different metric that does not take into account the whole order, but only the relative order of the $k$ top elements, where $k$ is either constant, or a fixed ratio of nodes; this approach might also be modified so to use a definition of $\tau$ that works also when the orders are not defined over the same set of nodes, like the ones given in [13]. Note, however, that in real-world applications the whole order is very important (even more important than the order of high-quality nodes, as explained at the end of Section 4).

- *More strategies.* Our results would suggest the implementation of crawl strategies that work effectively with respect to both PageRank and $\tau$. A natural candidate strategy would be a mixed order, using a depth-first intra-site visit, and operating breadth-first for extra-site links. This strategy would be quite natural for a parallel or distributed crawler, in that every worker might proceed in depth-first order within a site (respecting, of course, the usual politeness assumptions) whereas out-of-site links might be treated in a breadth-first manner: UbiCrawler [4] was originally designed to work with this strategy. Another interesting strategy could be performing a crawl in lexicographic order (the next node to be visited is the first node of the frontier in lexicographic order).

- *More random models.* The two graph models we have tried presented behaviours that did not quite match with real-world graphs. A natural investigation would be trying to discover if other more sophisticated models work better.

# References


[1] Serge Abiteboul, Mihai Preda, and Gregory Cobena. Adaptive on-line page importance computation. In *Proceedings of the twelfth international conference on World Wide Web*, pages 280–290. ACM Press, 2003.

[2] Reka Albert, Albert-Laszlo Barábasi, and Hawoong Jeong. Diameter of the World Wide Web. *Nature*, 401(6749), September 1999.

[3] Monica Bianchini, Marco Gori, and Franco Scarselli. Inside pageRank. *ACM Transactions on Internet Technologies*, 2004. To appear.

[4] Paolo Boldi, Bruno Codenotti, Massimo Santini, and Sebastiano Vigna. Ubicrawler: A scalable fully distributed web crawler. *Software: Practice & Experience*, 34(8):711–726, 2004.

[5] Paolo Boldi and Sebastiano Vigna. The WebGraph framework II: Codes for the World–Wide Web. Technical Report 294-03, Università di Milano, Dipartimento di Scienze dell'Informazione, 2003. To appear as a poster in *Proc. of DCC 2004*, IEEE Press.

[6] Paolo Boldi and Sebastiano Vigna. The WebGraph framework I: Compression techniques. In *Proc. of the Thirteenth International World Wide Web Conference*, pages 595–601, Manhattan, USA, 2004.

[7] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1/7):107–117, 1998.

[8] Junghoo Cho, Hector García-Molina, and Lawrence Page. Efficient crawling through URL ordering. *Computer Networks and ISDN Systems*, 30(1–7):161–172, 1998.

[9] Debora Donato, Luigi Laura, Stefano Leonardi, and Stefano Milozzi. A library of software tools for performing measures on large networks, 2004. `http://www.dis.uniroma1.it/ cosin/html_pages/COSIN-Tools.htm`.

[10] Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the tenth international conference on World Wide Web*, pages 613–622. ACM Press, 2001.

[11] Nadav Eiron, Kevin S. McCurley, and John A. Tomlin. Ranking the web frontier. In *Proceedings of the 13th conference on World Wide Web*, pages 309–318. ACM Press, 2004.

[12] Ronald Fagin, Ravi Kumar, Kevin S. McCurley, Jasmine Novak, D. Sivakumar, John A. Tomlin, and David P. Williamson. Searching the workplace web. In *Proceedings of the twelfth international conference on World Wide Web*, pages 366–375. ACM Press, 2003.

[13] Ronald Fagin, Ravi Kumar, and D. Sivakumar. Comparing top k lists. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 28–36. Society for Industrial and Applied Mathematics, 2003.

[14] T. Haveliwala. Efficient computation of pagerank. Technical report, Stanford University, 1999.

[15] Jun Hirai, Sriram Raghavan, Hector Garcia-Molina, and Andreas Paepcke. Webbase: A repository of web pages. In *Proc. of WWW9*, Amsterdam, The Netherlands, 2000.

[16] Sepandar D. Kamvar, Taher H. Haveliwala, Christopher D. Manning, and Gene H. Golub. Exploiting the block structure of the web for computing pagerank. Technical report, Stanford University, 2003.

[17] Sepandar D. Kamvar, Taher H. Haveliwala, Christopher D. Manning, and Gene H. Golub. Extrapolation methods for accelerating pagerank computations. In *Proceedings of the twelfth international conference on World Wide Web*, pages 261–270. ACM Press, 2003.

[18] Maurice Kendall and Jean Dickinson Gibbons. *Rank Correlation Methods*. Edward Arnold, 1990.

[19] Maurice G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1–2):81–93, 1938.

[20] Maurice G. Kendall. *Rank Correlation Methods*. Hafner Publishing Co., New York, 1955.

[21] William R. Knight. A computer method for calculating kendall's tau with ungrouped data. *Journal of the American Statistical Association*, 61(314):436–439, June 1966.

[22] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal. Stochastic models for the web graph. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, page 57. IEEE Computer Society, 2000.

[23] Amy N. Langville and Carl D. Meyer. Deeper inside pageRank. *Internet Mathematics*, 2004. To appear.

[24] Hyun Chul Lee and Allan Borodin. Perturbation of the hyper-linked environment. In *Computing and Combinatorics, 9th Annual International Conference, COCOON 2003, Big Sky, MT, USA, July 25-28, 2003, Proceedings*, volume 2697 of *Lecture Notes in Computer Science*, pages 272–283. Springer, 2003.

[25] R. Lempel and S. Moran. Rank stability and rank similarity of link-based web ranking algorithms in authority connected graphs, 2004. Information Retrieval (in print); special issue on Advances in Mathematics and Formal Methods in Information Retrieval.

[26] Marc Najork and Janet L. Wiener. Breadth-first search crawling yields high-quality pages. In *Proc. of Tenth International World Wide Web Conference*, Hong Kong, China, 2001.

[27] Andrew Y. Ng, Alice X. Zheng, and Michael I. Jordan. Stable algorithms for link analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 258–266. ACM Press, 2001.

[28] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, Stanford University, Stanford, CA, USA, 1998.

# Appendix

In Figure 3 we present the $\tau$ sequences for all the graphs and visit strategies and with three different starting URLs, as specified in the caption. A similar analysis is performed in Figure 4 for the cumulative PageRanks.

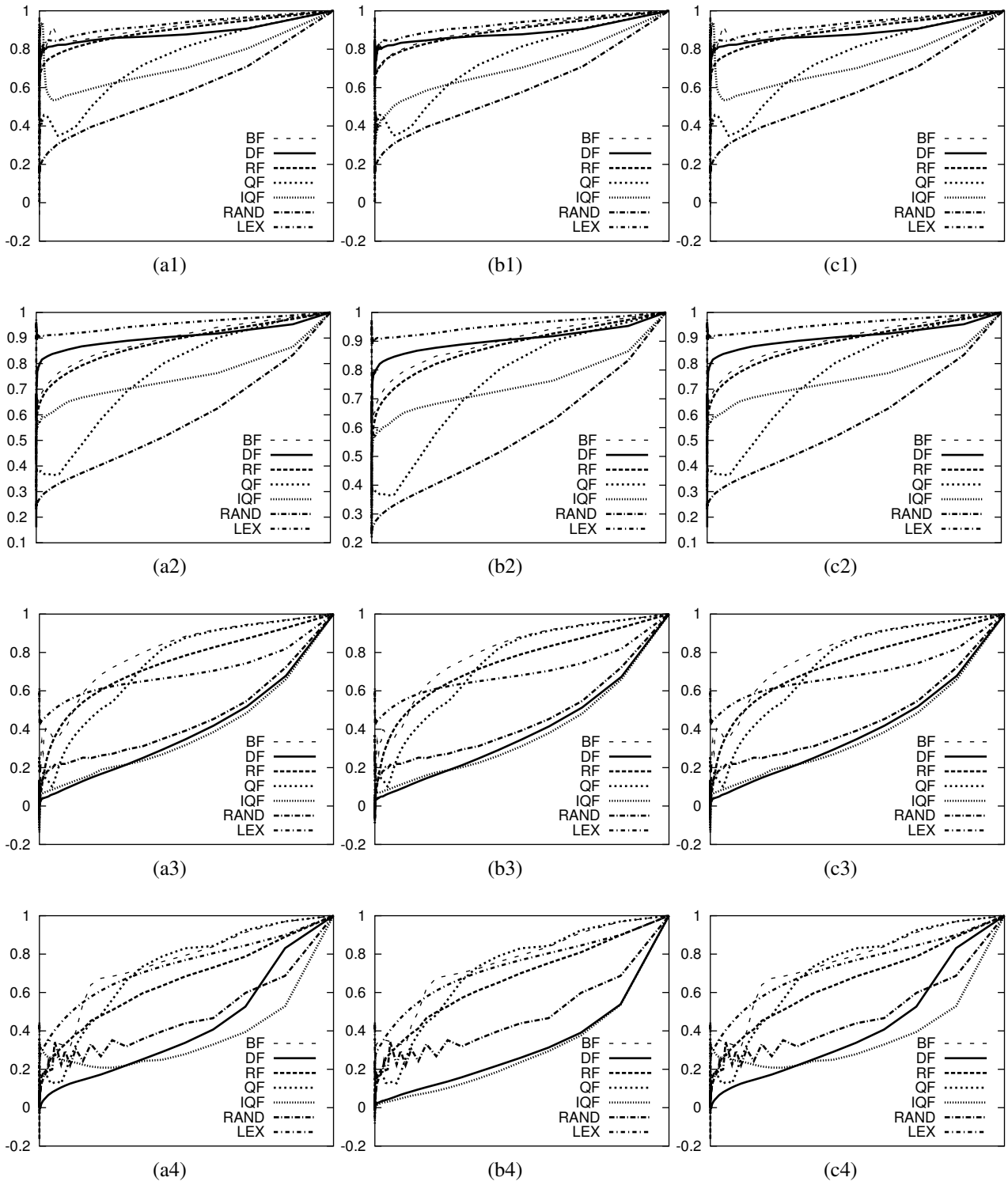


Figure 3: $\tau$ sequences for (1) the Italian graph, (2) the WebBase graph, (3) the copying-model graph, (4) the evolving-model graph, starting from (a) the highest-ranked, (b) the lowest ranked, (c) the median ranked node in the giant component.

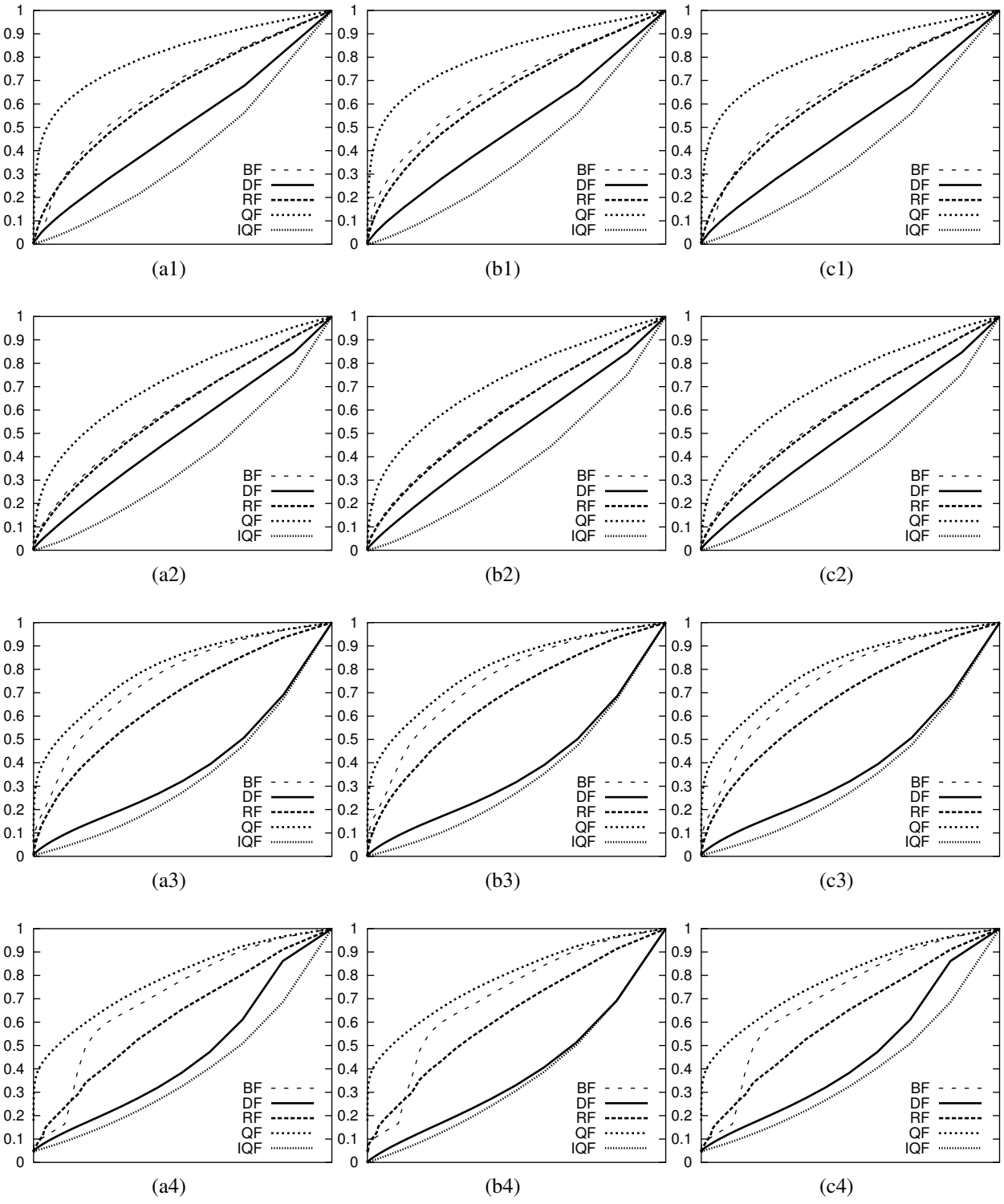


Figure 4: Cumulative PageRank obtained during a visit of (1) the Italian graph, (2) the WebBase graph, (3) the copying-model graph, (4) the evolving-model graph, starting from (a) the highest-ranked, (b) the lowest ranked, (c) the median ranked node in the giant component.