Web Graph Analyzer Tool

Konstantin Avrachenkov
INRIA Sophia Antipolis
2004, route des Lucioles, B.P.93
06902, France
Email: K.Avrachenkov@sophia.inria.fr

Danil Nemirovsky
St.Petersburg State University
35, Universitetsky Pr.,Peterhof,
St. Petersburg, 198504, Russia
Email: Danil.Nemirovsky@gmail.com

Natalia Osipova
INRIA Sophia Antipolis
2004, route des Lucioles, B.P.93
06902, France
Email: Natalia.Osipova@sophia.inria.fr

***Abstract*— We present the software tool "Web Graph Analyzer". This tool is designed to perform a comprehensive analysis of the Web Graph structure. By Web Graph we mean a graph whose vertices are Web pages and whose edges are hyper-links. With the help of the Web Graph Analyzer we can study the local graph characterictics such as numbers and sets of incoming/outgoing links to/from a given page, the page level relative to a given root page, and the global graph characteristics such as PageRank, Giant Strongly Connected Component, the number of dangling nodes. The Web Graph Analyzer has a user friendly GUI that allows an easy collection of a part of WWW and its thorough investigation. The Web Graph Analyzer is based on the Oracle DBMS which scales well with the large volumes of data.**

I. Introduction

During the last decade the information network World Wide Web (WWW) has transformed our society and economy. One of the main reasons behind the amazing success of WWW is its hyper-link structure. Hyperlinks are the pointers from some particular places of Web pages to other Web pages. They are alike citations in scientific literature. However, there are important differences between WWW and the network of scientific citations. It is much more common in WWW that two documents have reciprocal links. Furthermore, the Web pages and hyper-links not only can be created but also can be deleted. Thus, WWW is a much more dynamic information network than the network of scientific citations.

Due to the easy accessibility of the Web documents and the small world effect of WWW (almost any two Web documents are a few clicks from each other), the navigation in WWW is very simple. This fact has largely contributed to the popularity of Web surfing. However, even though Web surfing is easy and fun, the Web has billions of documents and it is often a challenging problem for a user to find relevant and high quality information on a subject in which he/she is interested. Numerous Search Engines try to solve the problem of the information search. The Google search engine is among the most succesful search engines if not the most successful one. Its success is attribuited primarily to the introduction of link-based ranking criterion PageRank [5] for the search results. PageRank criterion takes into account the structure of the Web Graph. By Web Graph we mean a directed graph whose vertices are Web pages and whose edges are hyper-links.

Curiously enough, a large number of search engine optimization companies has been created for the analysis and optimization of the hyper-link structure. Of course, most search engines and search engine optimization companies are very secretive about their studies of the Web Graph. The analysis of the Web graph is also very interesting not only from the commercial point of view but also its analysis is very interesting and challenging from the academic point of view. Still there are virtually no software tools available for the comprehensive, efficient and user friendly analysis of the Web Graph. Here we present such a software tool which we call "Web Graph Analyzer".

The rest of the paper is organized as follows: In Section II we define the Web Graph and its most important characteristics. In the ensuing Section III we outline the algorithms for the computation of the Web Graph characteristics. Then, in Section IV we present the Web Graph Analyzer tool and explain its design and its GUI. In Section V we describe the performance test for the Web Graph Analyzer. We conlude the paper with Section VI.

II. Web Graph

The World Wide Web hyper-text structure can be represented as a directed graph $G = \{V, E\}$, where Web pages are vertices and hyper-links are edges. In short, the graph of the World Wide Web hyper-text structure is called *Web Graph* (WG). Let $V = \{1, 2, ..., |V|\}$ denote the set of Web Pages and let $E = \{v \to u | v, u \in V\}$ denote the set of hyper-links. Thus, the notation $v \to u$ means that there is a hyper-link from page v pointing to page u. If there are multiple hyper-links from one Web page to another, we consider these multiple hyper-links as a single edge of the Web Graph. We also disregard the hyper-links inside a single Web page. Therefore, the Web Graph does not have loops, but, of course, it has cycles.

The Web Graph has local and global caracteristics. Among local caracteristics the most important are: numbers and sets of incoming/outgoing links to/from a given page, the page level relative to a given root page and also average values and distributions of these caracterictics. Among global caracteristics we can mention PageRank (PR), Giant Strongly Connected Component (Giant SCC) and the number of dangling nodes.

For completeness of the presentation, let us provide more detailed definitions and properties of the above mentioned caracteristics.

Thus, we have the following definition of the number of incoming/outgoing edges.

Definition 1: The number of incoming/outgoing edges to/from a given page is equal to the number of pages which have links pointing/pointed to/by the given page.

Definition 2: The level of a page v relative to a root page u corresponds to the minimal distance from the root page u to the given page v.

In other words, the page level shows what minimal number of "clicks" a user should do to reach the given page starting from the root page. One can extend the above definition to the case of several root pages. In that case the page level is selected as the minimal level among all levels corresponding to the root pages.

Definition 3: A dangling node is a page which does not have links to the other pages of the Web Graph.

PageRank (PR) is the eigenvector centrality measure of the Web Graph. Its elements represent the authority or popularity of Web pages. PageRank is used by the search engine Google as one of the principle criteria to rank answers to a user's query. The formal definition of PageRank is given below.

Definition 4: Denote by n the total number of pages on the Web and define the $n \times n$ hyperlink matrix P as follows. Suppose that page i has $k > 0$ outgoing links. Then $p_{ij} = 1/k$ if j is one of the outgoing links and $p_{ij} = 0$ otherwise. If a page does not have outgoing links, the probability is spread among all pages of the Web, namely, $p_{ij} = 1/n$. In order to make the hyperlink graph connected, it is assumed that a random surfer goes with some probability to an arbitrary Web page with the uniform distribution. Thus, the PageRank is defined as a stationary distribution of a Markov chain whose state space is the set of all Web pages, and the transition matrix is

$$\tilde{P} = cP + (1-c)(1/n)E, \tag{1}$$

where E is a matrix whose all entries are equal to one and $c \in (0,1)$ is the probability of not jumping to a random page (it is chosen by Google to be 0.85). The Google matrix $\tilde{P}$ is stochastic, aperiodic, and irreducible, so there exists a unique row vector π such that

$$\pi\tilde{P} = \pi, \quad \pi\underline{1} = 1, \tag{2}$$

where $\underline{1}$ is a column vector of ones. The row vector π satisfying (2) is called a PageRank vector, or simply PageRank.

PageRank has the following useful interpretation: If a Web surfer follows a hyperlink with probability c and jumps to a random page with probability $1-c$, then PageRank π_i can be interpreted as a stationary probability that the surfer is at page i.

Definition 5: The Strongly Connected Component (SCC) is a subgraph of G in which from any page there is a way to any other page of that subgraph. Namely, G^{SCC} is a SCC if $\forall v, u \in G^{SCC} \quad \exists\{v_1, v_2, ..v_n\} \in G^{SCC}$ such that $\exists v \to v_1, v_1 \to v_2, ..., v_n \to u$.

It turns out that in the Web Graph there is a Giant Strongly Connected Component, which is significantly larger than the other SCCs of the Web Graph. For every Web page owner it is important to know, if the page belongs to the Giant SCC or not. In the case when a page does not belong to the Giant SCC the probability for a Web surfer to find it is typically very small.

III. Algorithms Description

Let us outline algorithms that are used in the software tool "Web Graph Analyzer".

A. PR computation algrithms

There are several methods for PR computation. The method which is used by Google is Power Iteration (PI) method. It is based on successive multiplications of the PR approximation vector by the transition matrix given in (1).

Algorithm 1: Power Iteration method

0. The initial approximation is chosen as the uniform distribution vector $\pi^{(0)} = (1/n)\underline{1}^T$.
1. The k-th PR approximation vector is calculated by
$$\pi^{(k)} = \pi^{(k-1)}\tilde{P}, \quad k \geq 1. \tag{3}$$
2. The method stops when the required precision ε is achieved, i.e., $||\pi^{(k)} - \pi^{(k-1)}||_1 \leq \varepsilon$. If the precision is not yet achieved then return to Step 1.

The number of flops needed for the method to converge is of the order $\frac{\log \varepsilon}{\log c}\, nnz(P)$, where $nnz(P)$ is the number of non-zero elements of the matrix P [7]. We note that the relative error decreases uniformly for all pages.

Several proposals (see extensive survey papers [7] and [3]) have recently been put forward to accelerate the power iteration algorithm.

We emphasize that the implementation of the PI method does not really need to multiply the PR approximation vector by the transition matrix. The transition matrix $\tilde{P}$ is composed of the hyperlinkmatrix P which is very sparse and the rank-one matrix E. The latter fact is explained by the following equivalent form of (3):

$$\pi^{(k)} = c\pi^{(k-1)}P + (1-c)(1/n)\underline{1}, \quad k \geq 1.$$

There is another class of efficient methods for PR computation. These methods are probabilistic Monte Carlo (MC) methods [2], [4]. The Monte Carlo methods are based on simulation of a random surfer going through the Web Graph.

The principle advantages of the probabilistic Monte Carlo type methods over the deterministic methods are: the PageRank of important pages is determined with high accuracy already after the first iteration; MC methods have natural parallel implementation; and MC methods allow continuous update of the PageRank as the structure of the Web changes.

Monte Carlo algorithms are motivated by the following convenient formula that follows directly from the definition of the PageRank:

$$\pi = \frac{1-c}{n}\underline{1}^T[I-cP]^{-1} = \frac{1-c}{n}\underline{1}^T\sum_{k=0}^{\infty}c^kP^k. \qquad (4)$$

There are two interpretations of this formula.

Consider a random walk $\{X_t\}_{t\geq 0}$ that starts from a randomly chosen page. Assume that at each step, the random walk terminates with probability $(1-c)$, and makes a transition according to the matrix P with probability c. It follows from (4) that the end-point of such random walk has a distribution π.

The second interpretation is that π is also a distribution of frequency of visiting all pages during the random walk.

According to these interpretation there are several versions of MC method. In [2] we performed an extensive investigation and comparison of different Monte Carlo methods. Below we present the most efficient version.

Algorithm 2: MC complete path stopping at dangling nodes

For this algorithm we use a slightly different definition of the hyperlink matrix that disregards dangling nodes. Namely, let Q be a modified hyperlink matrix with elements defined as $Q_{ij} = 1/k$, if page i has $k > 0$ outgoing links, and a link points to page j, and 0, otherwise.

Then, we simulate the random walk $\{Y_t\}_{t\geq 0}$, whose transitions are governed by matrix Q, starting exactly k times from each page. The random walk $\{Y_t\}_{t\geq 0}$ can be terminated at each step either with probability $(1-c)$ or when it reaches a dangling node.

Finally, for any page j, evaluate the estimate of π_j as the total number of visits to page j divided by the total number of visited pages.

B. SCC computation algorithm

The next algorithm allows us to determine an SCC, which includes a given root page.

Algorithm 3:

0. The root page is selected.
1. With the depth search algorithm the set S_1 of all pages reachable from the root page is found.
2. With the depth search algorithm the set S_2 of all pages from which the root page is reachable is found.
3. $S = S_1 \bigcap S_2$ is an SCC.

C. Level computation algorithm

For computating the levels first one or several root pages are selected. The selection is the user's choice. Denote by $lvl(u)$ the level of page u. Then, the level computation algorithm is as folows:

Algorithm 4:

0. $S_1 = \{\text{root pages}\}, lvl(u) = 1, u \in S_1$
1. $S_2 = \{v \in G \mid \exists u \to v, u \in S_1, lvl(v) = null\}$. If $S_2 = \emptyset$, Stop.
2. $lvl(v) = lvl(u) + 1$, $v \in S_2$, $u \in S_1$.
3. $S_1 = S_2$, $S_2 = \emptyset$.
4. Go to Step 1.

Analyzing the structure of the WG we found that the fraction of pages with high level numbers is small and those pages with high level numbers have very few links. Thus, it seems to be computationally efficient to calculate the PR approximation taking into account only the first few levels of the WG. For such a "restricted" PageRank we have to select the subgraph G^l of the graph G according to the page levels as follows: $G^l = \{u \in G | lvl(u) \leq l\}$.

The notion of levels is also very useful for the computation of query dependent link based criteria [6].

IV. Web Graph Analyzer Tool

To realize the methods listed in the previous section and to provide a framework to implement the other Web Graph specific methods, we developed a software tool called "Web Graph Analyzer" (WGA).

The WGA tool is designed for the solution of the following problems:

- collecting and storing information about Web pages and hyper-links;
- searching pages/hyper-links by their URL names or by their neighbour pages URLs;
- computing WG local characteristics:
  - finding sets of incoming/outgoing links for a page;
  - finding in- and out- degree distributions;
  - computing page levels;
  - ...
- computing WG global characteristics:
  - conputing PageRank with PI and MC methods;
  - SCC detection;
  - ...
- user friendly representation of the WG structure analysis.

A. Software Choice

Analyzing the Web Graph, one has to store and to process a huge volume of data. An efficient way to store and to process that huge volume of data is to use a Data Base (DB). The number of Web pages is not only huge but also it grows from day to day. Also the quality and the interest of the results of the most algorithms improves with the increased volume of the collection. This is why when creating a system for the analysis of the WG structure, it is very important to make the system scalable from the very beginning.

To retrieve the information about the WG structure, a special program, Crawler, was developed using Java and PL/SQL. More details about the Crawler are given in one of the subsections below.

When realizing and running any algorithms on the Data Base, we need to extract the data from the DB, to copy it

into the memory of the computer, to process it in some way and to write the results back to the DB. The best way to realize all the operations with data stored in the DB is to use a Data Base Management System (DBMS). An example of high-performance, scalable and secure DBMS system is the Oracle DBMS. Oracle DBMS can maintain and process a very large amount of data. Its high performance is due to the use of programming language PL/SQL. Oracle DBMS uses stored procedures which run inside the kernel of DBMS in the most efficient order and provide fastest execution.

As Oracle has the same interpretor for all the operating systems there is no need to recompile the code when the user needs to work on another platform.

The multi-user access is already realized in the Oracle DBMS, so we can organize information retrieval and calculation processes at the same time. For instance, it becomes easy to realize parallel PageRank calculations with the MC method on multiprocessor computer working with the same Data Base. Also we can make run several Crawlers on different computers in a local network updating the same Data Base.

All algorithmes mentioned in the previous section are written in the PL/SQL language and are realized as stored procedures, which run inside the kernel of DBMS.

The system principle scheme is presented in Figure 1.

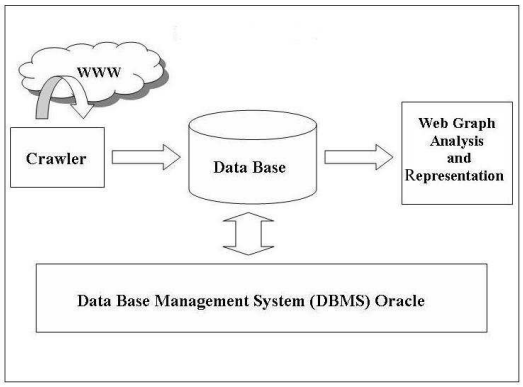

Fig. 1. System structure

Graphical User Interface is realized using Oracle Developer Tool. It is a special development environment to represent the work with the Oracle DB in the user friendly manner.

B. Data Base Structure

Our DB system consists of the following tables:

- pages;
- links;
- auxiliary tables;
- tables to store information for the Crawler.

The table ”pages” is used for page indexation. Every page has its own identity index. In auxiliary tables these indecies are used instead of the full pages URLs. This minimizes stored volumes of information and decreases processing time. In the table ”pages” all information about the page is stored: page index, URL, PR, number of ingoing and outgoing links, page level, and possibly page content.

Table ”links” stores the information about hyper-links between Web pages. For storing the link information the method of adjacency list is used [1]. Since the Web Graph is sparse, the method of adjacency list avoids unnecessary storage of many zero elements of the hyperlink matrix. Furthermore, the adjacency list not only saves a lot of storage space but also simplifies significantly the operations with WG.

The auxiliary tables are used to store all the intermediate data which is need to implement the algorithms.

Tables with the information for crawler store such information as the intitial page URL, the Web filters, the extensions of the pages we want to collect and so on. All this information is needed to organize the data collection from the Web.

C. Crawler

The Crawler is designed to collect samples of WG as fast as possible and at the same time to use efficiently and carefully computation and network resources.

A number of Crawler instances can be run in local area network to use computation resources efficiently. All instances use one DB to store the collected data but different computers to retrieve and to parse Web pages. The Crawlers concurrently distribute among themselves Web pages and then treat these pages autonomously.

A flexible system is used to restrict a sector of the crawled Web. The sector can be limited by URLs, IP addresses or by the country of the Web server location. Two lists of regular expressions are used to restrict the sector by URL. The lists are called ”filter” and ”fence”. If a URL satisfies at least one regular expression from the filter list, that page can be downloaded for the further analysis. If a URL satisfies at least one regular expression from the fence list, the Web page with that URL is not downloaded. A URL has to pass both the filter and the fence to be downloaded. Filtration by IP address is carried out in a similar way. One can specify ranges of IP addresses that can be accessed and ranges of IP addresses that are not allowed to be accessed. The IP addresses are resolved by URL. The table of correspondence country-IPaddress is used to filter by country of the server location.

While working the Crawler maintains two buffers of pages: the buffer of downloaded pages and the buffer of new pages. The first one is used to resolve indices of URL received by parsing. The length of that buffer is limited. When the limit is reached the pages from head of the buffer is removed and then new pages are inserted in the tail.

The buffer of new pages is responsible for pages downloading and distributing among Crawler instances. When the buffer becomes empty the crawler tries to get new pages from DB. If its attempt is successful it starts to work on the retrieved pages. If it fails, the Crawler instance goes into the waiting state. The waiting state of the Crawler instance can be noticed by other crawler instances that are working with the same DB. In this case they free a number of their new pages that will be passed to the waiting Crawler instance.

If only a single Crawler instance is run in the local network the mode of its functioning depends on the buffer length of new pages. If the length is one, then the Crawler goes through a chain of pages until it reaches a page without outgoing links. Then it sends a request to DB for a new URL. This mode closely corresponds to the Deapth-First-Search (DFS) traversal method [1]. If the buffer length were unlimited, then the mode corresponds to Width-First-Search (WFS) traversal method [1]. If the buffer length is finite, then we get some traversal method which is in between DFS and WFS.

D. Algorithms Implementation

All algorithms of the system are realized as stored procedures in PL/SQL language.

The use of the Oracle DBMS indexation scheme significantly decreases the computation time for all algorithms. Every table in the DB has one or several indices which makes it possible to work with the information in the most efficient order.

The numbers of incoming and outgoing links are efficiently computed using the table ”links”.

The table ”pages” is often updated according to the new/updated information.

The algorithm of PR computation with the PI method is realized in the following way: first the intial values $1/n$ is assigned to the PR column. Then, for every iteration $\pi^{(k+1)}$ is computed using the previous value of $\pi^{(k)}$ according to the table ”links”. The updated value of PR on every step is stored in the table ”pages”.

PR computation with the MC method is realized according to the Algorithm 2. For the MC method realization the recursive procedure which simulates the Web surfer is run many times. It starts a random walk from each record of the table ”pages”. With the probability c it goes to the randomly selected pages according to the table ”links”. With the probability $1-c$ it starts a new random walk from the next page in the list of pages. When the simulation is stopped, the PR is computed as the fraction of the number of visites of a particular page to the total number of visites. Then the value of PR is updated in the in table ”pages”.

It is easy to organize parallel PR computation with MC methods. To do this, several recursive procedures are run at the same time. They put the collected data to the same auxiliary table according to which at the end of simulation PR is computed.

Also it is possible to oraganize parallel PR computation with the MC method and the new information collection from the Web as the Crawler works independently of the computational algorithms.

The level computation is organized according to Algorithm 4. To implement Algorithm 4 two auxiliary tables are created to store the sets S_1 and S_2. The levels of pages are stored in the table ”pages”. The root page can be either the page from which the crawling is started or it could be a page given by the user.

E. Graphical User Interface

The Graphical User Interface (GUI) is developed using Oracle Developer. All the functions of the interface are realized in the kernel of DBMS and are run on the server, which implies fast access to the data and the fast data analysis.

At the moment of the present article writing the GUI functions are as follows.

There is a possibility to set different parameters of the Crawler and to run the Crawler from GUI (see Figure 2). You can set such parametres as an initial URL, the file extensions which the downloaded pages must have. Also there is a possibility do not collect the pages with the given parts of URL. For example, if you make an experiment on a Web site that you know, you might prefer not to collect technical pages of that site.

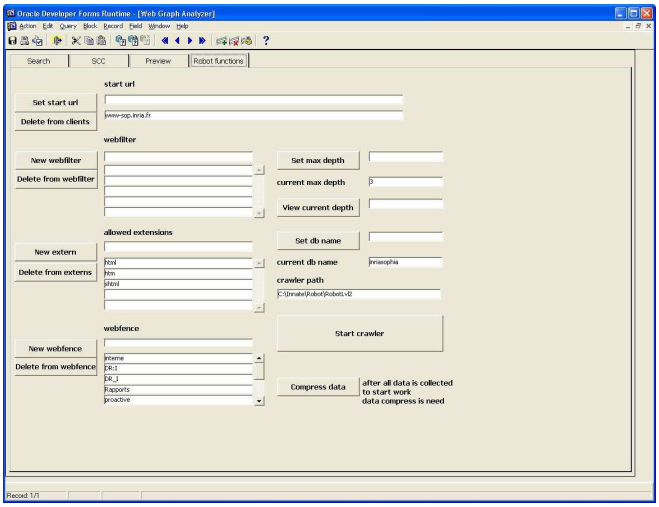

Fig. 2. Crawler parameters.

The user can compute a number of the WG characteristics. There is a possibility to calculate PR with PI and MC methods (see Figure 3). The user can set the number of iterations. Then, after the algorithms execution is finished, PR as well as the relative error are displayed. At each algorithms iteration there is a possiblity to delete the data and to start the calculations from the beginning. The user can find the numbers of incoming and outgoing links, calculate the levels of pages. Furthermore, the user is able to determine a SCC for a given root page.

The URL search is also realized in the interface (see Figure 4). Namely, it is possible to search hyper-links by their URL or by a part of URL. Then, the interface displays lists of incoming and outgoing links for a given page, the page level, the PR of the page calculated with different methods.

V. Performance Test

We performed several experiments on `www.inria.fr` site with the following volume of data: 300.000 pages and 1.500.000 links. The iterations of the PI method is compared with the iterations of the MC method for the PageRank computation. Here we discuss the results of PageRank computation

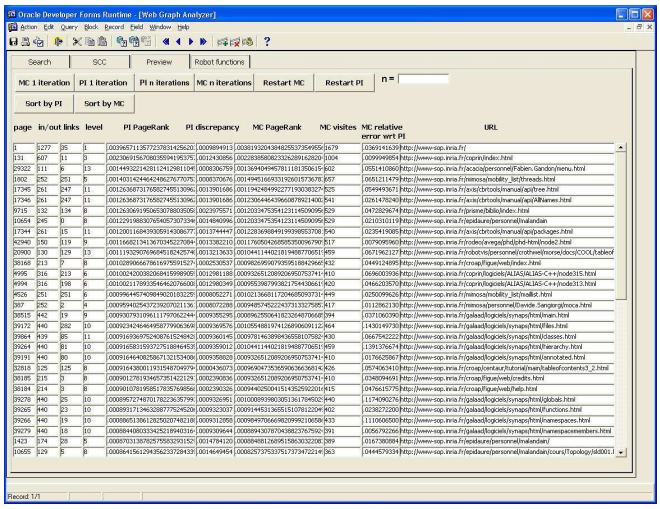

Fig. 3. PageRank computation methods.

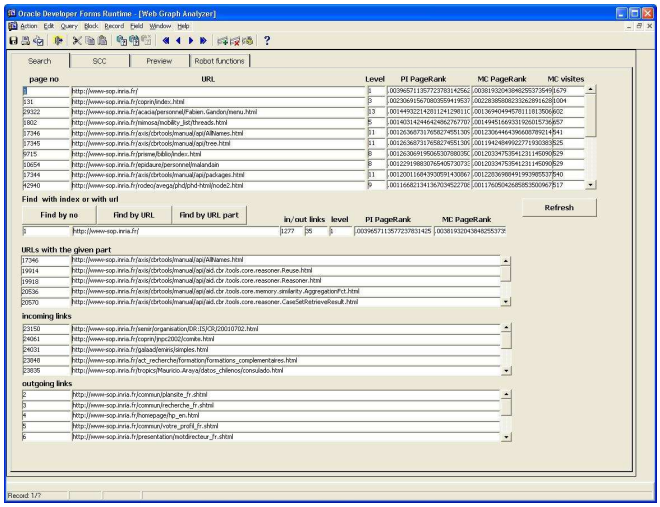

Fig. 4. Search and Analysis Page.

as the computation of this Web Graph characteristic is the most demanding. Therefore we carry out the performance test of our tool on the PageRank computation algorithms. The computation of one PI iteration took approximately 20 minutes and one iteration of the MC method took about 20% less time. For this test we have used a computer with Intel Pentium III 1 Ghz Processor and 512 Mbytes of RAM.

VI. Conclusions

We presented "Web Graph Analyzer" software tool, which is design to perform a thorough study of the World Wide Web hyper-text structure. In particular, "Web Graph Analyzer" allows one to investigate local graph characteristics such as in-/out-degrees and page level as well as global characteristics such as PageRank and Giant Strongly Connected Component. The graph analysis is performed with the help of a user friendly GUI. All numerical algorithms are efficiently realized using stored procedures of Oracle DBMS. The system scales well with large volumes of data.

In the future versions of the "Web Graph Analyzer" we plan to include the algorithms for more graph characteristics and to introduce the possibility of displaying plots.

References

[1] A.V. Aho, J.E. Hopcroft, and J.D. Ullman, *Data Structures and Algorithms*, Addison-Wesley, 1983.

[2] K. Avrachenkov, N. Litvak, D. Nemirovsky, and N. Osipova, "Monte Carlo Methods in PageRank computation: When one iteration is sufficient", University of Twente Technical Report no. 1754, March 2005, also submitted for journal publication.

[3] P. Berkhin, "A Survey on PageRank Computing", *Internet Mathematics*, v.2, no.1, pp.73-120, 2005.

[4] L.A. Breyer, "Markovian page ranking distributions: some theory and simulations", 2002, available at `http://www.lbreyer.com/preprints.html`

[5] S. Brin, L. Page, R. Motwami and T. Winograd, The PageRank citation ranking: bringing order to the Web, Stanford University Technical Report 1998, available at `http://dbpubs.stanford.edu:8090/pub/1999-66`.

[6] M. Henzinger, "Hyperlink Analysis on the World Wide Web", in Proceedings of The 6-th ACM Conference on Hypertext and Hypermedia (HT'05), September 2005, Salzburg, Austria.

[7] A.N. Langville and C.D. Meyer, Deeper Inside PageRank, Internet Mathematics, v.1, no.3, pp.335-380, 2003.