Agent-based Approach for Web Crawling

M. Wack M. Bakhouya J. Gaber

Laboratoire SeT
Universite de Technologie de Belfort-Montbeliard
90010 Belfort Cedex, France
Email: {gaber, mohamed.bakhouya, maxime.wack}@utbm.fr

Abstract: Since its creation in 1990, World Wide Web has increased the popularity of Internet which becomes an important source of information or services for all people over the world. The dynamic nature of the Web draws attention to the need for continuous support and updating of Web information retrieval systems. Web crawling is the process of discovery and maintenance of large-scale web data. Crawlers achieve this process by following the Web pages hyperlinks to automatically download a partial snapshot of the Web. In this paper, an agent-based approach, through three scenarios, for parallel and distributed Web crawling is presented. Simulations with ns2 show that the cloning based mobile agents scenario outperforms the single and multiple mobile agents scenarios.
Keywords: web crawling, mobile agent, agent cloning, self-regulation, parallel random walk.

1. Introduction

The web crawling system is the system responsible for maintaining the search engine database and incorporating the changes from the web. More precisely, a crawler starts at a node and then follows the edges to search other nodes. Once a page is fetched, it determines the IP address of a chosen URL's host name, opens socket connection to the corresponding server, asks for the particular document and downloads it. The obtained downloaded pages may be stored or indexed for other applications (e.g., search engine or Web cache) [8] [18]. In other words, the main purpose for designing Web crawlers is to retrieve Web pages and add them or their representations to a local repository. Such a repository may then serve particular application needs such as those of Web search engines.

Several crawling systems have been described in the literature such as Google crawler [7], Mercator web crawler [18] [16], WebFountain [12], UbiCrawler [5] [6], Garcia-Molina's crawler [11] and Shkapenyuk and Suel's crawler [23]. Most of these web crawlers are centralized and assigned all the task from downloading pages from the web, and processing them, to integrate the results in the search engine database. Centralized Web crawling systems are however inflexible due to the risk of bottlenecks and the presence of a single point of failure. Also, they are often unsatisfactory, because of possible dynamic changes in the network connection. In addition, the information available in the Internet is very dynamic; there is no central supervision on the growth of the Internet and it is difficult to make an estimation of the amount of information available through it [8] [19]. As the size of the Web grows, it becomes imperative to use parallel and distributed crawling process, in order to achieve downloading pages in a reasonable amount of time. For this reason, J. Hammer and J. Fiedler in [13] and [15] initiated the concept of Mobile Crawling by the use of crawling units. In this approach, crawlers are dispatched to remote Web severs for local crawling and processing of Web documents. After crawling a specific Web server, they dispatch themselves either back at the search engine machine, or at the next Web server for further crawling. According to [20], the model offers speed and efficiency compared to current crawling techniques. However, important issues should be addressed such as a more efficient resource management and a more decentralized control of the architecture. Another approach proposed in [21] and [22] uses mobile crawlers for fully distributed crawling. In this paper, techniques from the random walk theory [2] [9] together with a mobile agent-based approach with a cloning mechanism are used for web crawling.

Mobile agent technology offers a new computing paradigm in which a program, in the form of a software agent, can suspend its execution on a host computer, transfer itself to another agent-enabled host on the network, and resume execution on the new host [14]. More precisely, a mobile agent is an autonomous program that can move between sites of the network and perform computations at these sites on behalf of a user or an application [10] [17]. The mobile agents include the computation along with their data and execution state. By moving to the location sites, the agent can autonomously crawl the web while eliminating the transfer of intermediate results across the network and thus reducing the end-to-end latency. Also, an agent has the ability to perceive its environment and react to changes that occur in it [17].

Several applications have shown clear evidence of benefiting from the use of mobile agents such as electronic trading, distributed information retrieval and information dissemination [3]. However, the security issue is one of the most important drawbacks for the mobile agents paradigm. Most of the modern mobile agents platforms offer several authentication methods to secure the host computer and its resources from malicious and unsigned code [14].

The proposed agent-based approach in this paper is analyzed through three scenarios for parallel and distributed Web crawling. The proposed approach suits dynamic setting in WWW where information might dynamically and unpredictably change and a complete knowledge of this change is difficult to obtain.

The rest of the paper is organized as follows. In section 2 the WWW model is presented. In section 3, the proposed agent-based scenarios are presented and the asymptotic time complexity of them as well as the expected performance is described. Section 4 presents the experimentation result. Conclusion is given in section 5.

2. WWW Model

WWW network can be viewed as an undirected, connected graph $G= (S,V)$, where S is the set of sites ($|S|=n$) labeled with URLs and V the set of links ($|V|=m$) connecting the sites. Let information and services be stored in web pages spread over sites. A site s_i is considered to be connected to a to site s_j, if there is a web link between s_i and s_j.

Recall that a random walk on a graph is a stochastic process that iteratively visits the vertices of the graph. From a given site, the walk process moves at the next step to an adjacent site chosen uniformly at random [2]. The visited sites can be written as a sequence of sites $(s_1, s_2, s_3,..., s_n)$. The state of the random walk after t hops can be described by a probability distribution X_t over the graph. An important result showed by Broder and al. in [9] states that the cover time needed for p parallel random walks to visit all the vertices in a graph is $O(\frac{m^2\log^3(n)}{p^2})$.

In this paper, the possibility of embedding the mobile agent approach and the random walk technique in designing distributed algorithm for intelligent Web crawling is described and analyzed. The aim is to develop a random walk agent based approach with cloning capabilities on WWW network that satisfies the following criteria: the number of steps required for the walk to converge (i.e. to visit n pages) should be reasonable.

3. Crawling Process

In the crawling process, three scenarios for Web crawling are considered. In the first scenario, a single mobile agent starts from a seed page (i.e. starting page) and then uses external links within it to get access to other pages. The mobile agent chooses randomly between these links, determines the IP address of the chosen URL's host name and moves to the corresponding server. The mobile agent repeats this process with the new pages reached and offering external links to follow. In this scenario, the single mobile agent eliminates the transfer of intermediate results across the network and thus reduces the end-to-end latency. However, the time to crawl all n Web pages could be unreasonable for large value of n (see corollary 1, section 3.1). To reduce this latency problem, multiple mobile agents and mobile agent cloning scenarios could be more suitable for the Web crawling process. In the multiple agents' scenario, an initial population of mobile agents is initially created and dispatched randomly for parallel Web crawling. This scenario should allow the agents to visit all the n candidate pages in a reasonable amount of time compared to the single mobile agent scenario (see corollary 2, section 3.2). In the mobile agent-cloning scenario, an initial population of mobile agent starts, in first time, on its seed page. At each hop, mobile agent determines the IP address of randomly chosen URL, creates replications (i.e. clones) to itself, pass tasks to this clones that move to further pages. This scenario should allow mobiles agents to cover a much wide area of information space in a reasonable amount of time (see corollary 3, section 3.3) compared to the single mobile agent scenario and the multiple mobile agents' scenario.

The rest of this section presents the performance analysis of these scenarios. This analyze consider the over time needed to visit n pages with a single mobile agent, multiple mobile agents and mobile agent with a cloning scenario respectively.

3.1 Single Mobile Agent Scenario

This scenario is the straightforward one to consider for crawling with mobile agents. Let us consider n pages to be crawled by one mobile agent that performs a random walk among these pages. The analysis of this scenario can be derived directly from the well-known theorem of Broder and al. [9] as a corollary.

Corollary 1. Let us consider the simple mobile agent scenario with n pages to be crawled. A mobile agent starts from each seed page and parses it to extract all linked URLs. The mobile agent repeats the process until all the n pages have been visited. The cover time required until all the pages have been visited by mobile agents in parallel is $O(n^2\log^3(n))$.

Proof. The theorem of Broder and al. [9] states that the cover time to visit all vertices of graph with n vertices by p parallel random walks is $O(\frac{m^2\log^3(n)}{p^2})$. Since n parallel random walks ($n=p$) participating to the crawling process and since $m=O(n^2)$, the cover time becomes $O(n^2\log^3(n))$.

3.2 Multiple Mobile Agents Scenario

According to this scenario, at each seed page, a set of mobile agents is created. At each hop, a mobile agent analyses the page, examines the external links within it, and chooses randomly the next link to follow. The mobile agent moves to corresponding IP address host name and repeats the process until all pages have been visited. Unlike the single mobile agent scenario, the web crawling is made in parallel by p mobile agents. In this way, they should be able to cover a larger area of web space.

Corollary 2. Let us consider the multiple mobile agent scenario with n pages to be crawled. Let p mobile agents, at mean, start at each seed page. The expected number of rounds so that all the n web pages are visited is

$O(\frac{n^2\log^3(n)}{p^2})$, where n is the number of pages to be crawled and p is the average number of mobile agents dispatched initially by each seed page.

Proof. The theorem of Broder and al. [9] states that the cover time to visit all vertices of graph with p parallel random walks is $O(\frac{m^2\log^3(n)}{p^2})$. Since let consider that each seed page sends, at mean, p mobile agents initially, then there are np parallel random walks. Thus, the cover time is $O(\frac{m^2\log^3(n)}{(np)^2})$ and since $m=O(n^2)$, the cover time becomes $O(\frac{n^2\log^3(n)}{p^2})$.

3.3 Mobile Agent Cloning Scenario

In this scenario, each seed page starts with a single mobile agent that in order to crawl the n pages. At first time, the mobile agent examines external links within seed page, clones itself in c_0 mobile agents and sends them randomly to the neighbor pages, similarly with the multiple mobile agents' scenario. This initial population moves randomly and will create new mobile agents until all the pages are visited. The difference between the multiple mobile agents' scenario and the mobile agent-cloning scenario is the cloning mechanism. In these two scenarios, seed page sends an initial population of mobile agent. In multiple mobile agents' scenario, initial population does not change according to time but in mobile agent cloning scenario, it evolves from time to time. The increasing of the agent population size will increase resource demands in the network, which will affect the overall performance. Regarding the agent cloning operation, the distributed algorithm proposed by Amin and Mikler in [1] is used to regulate and control dynamically the number of clones spawned in the network. This approach is inspired by stigmergetic propriety of "ant colony" to facilitate coordination between mobile agents. More precisely, mobiles agents with minimum cognitive capabilities communicate with each other using pheromones that assist them to select an appropriate action. The intensity of pheromones disposed by agents at each node visited is determined by the equation $e^{-\lambda\Delta t}$, where Δt is the time since the deposition of pheromone and λ is a constant value fixed between 0 and 1. The controller of each agent contains the action selection algorithm that is defined as follows. An agent visiting a node at time t_b extracts the value of the value of the pheromone that was disposed at time t_a ($t_a \leq t_b$) using the equation $e^{-\lambda(t_b-t_a)}$. If this value is above a certain termination threshold *Max*, the agent kills itself. On the other hand, if the pheromone value reduces below a cloning threshold *Min*, the agent clones itself. But, if the pheromone is comprised between the termination and cloning thresholds (*Min* and *Max*), the agent neither clones nor kills itself. In this case, it migrates or moves to another peer node.

A new self-adaptive distributed algorithm inspired by the human immune system and proposed recently in [4] could also be used to regulate the agents' population size in large scale networks. The immune system presents emergent properties to make self-regulating and self-adapting in dynamically changing environment [3]. In this algorithm, each mobile agent selects locally an appropriate behavior to its environment state from the following ones: death, moving or cloning without using any global or constant threshold parameters [4].

Corollary 3. Let us consider the mobile agent cloning-scenario with n pages to be crawled and with an initial population size c_0 started at seed page. When a mobile agent visits a given page, it can clone itself and send its clones to the neighboring pages. The expected number of rounds required so that all the pages have been visited is $O(\frac{n^2\log^3(n)}{c_t^2})$, where c_t is the number of mobile agents (i.e. clones) spawned at each time t.

Proof. The theorem of Broder and al. [9] states that the cover time of p parallel random walks to visit n vertices of the graph is $O(\frac{m^2\log^3(n)}{p^2})$. Since let consider that at each seed page sends an initial population of mobile agents which will increased at each time. In addition, if each seed page sends an initial population of mobile agents. Then at worst of the cases there are nc_t random walks in the network. Thus the cover time is $O(\frac{m^2\log^3(n)}{(nc_t)^2})$ and like as $m=O(n^2)$, the cover time becomes $O(\frac{n^2\log^3(n)}{c_t^2})$.

4. Simulation Results

The proposed agent-based approach is evaluated by a Network Simulator NS2 [24]. The simulation abstracts any considerations about networking issues such as bandwidth constraints and time processing. The first part of our study is to compare the running time of these various scenarios. The time complexity analysis of these scenarios shows that the cloning mobile agent scenario is the best scenario to traverse the WWW network with a time $O(\frac{n^2\log^3(n)}{c_t^2})$ compared to the single mobile agent scenario with a time $O(n^2\log^3(n))$ and with multiple mobile agent scenario with a time $O(\frac{n^2\log^3(n)}{p^2})$, where $p=c_0$ and in our case $c_0 << c_t$ for each $t \neq 0$.

According to the time complexities described above, the performance of a scenario depends on the number of pages to be crawled and the number of mobile agents participating in the crawling process. Such as shown in Fig. 1, the single

mobile agent scenario shows a complexity time greater than those of multiple mobile agents and mobile agent cloning scenarios. Since the multiple mobile agents scenario allows a seed page to create a fixed number of mobile agents (5 mobile agents in our experimentation) and dispatch them in different directions, it allows mobile crawler agents to work in parallel, but shows a running time greater than that of a mobile agent cloning scenario. The cloning operation allows mobile agents to cover a much wide area of the information web space in a shorter time as shown in Fig. 1.

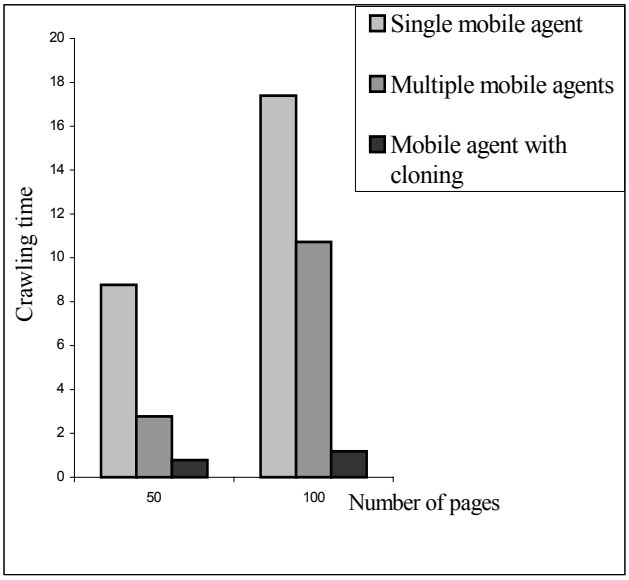

Fig. 1. Crawling time comparison of the three mobile agent scenarios.

5. Conclusion and Future Work

This paper analyzes three scenarios for web crawling using the mobile agent-based approach and random walk technique. From the above performance evaluation with ns2 simulations, the agent-based approach with cloning scenario outperforms the single agent-based and the multiple agent-based scenarios.

Future work will address additional simulations with ns2 to evaluate the approach performance when storage and bandwidth communication are considered and compare it with other approaches proposed in the literature. In this paper, we have concentrated our effort first to demonstrate the viability of the proposed agent based approach with three scenarios with a complexity analysis and a validation by simulations with ns2.

Further research will address also security issues and agent-based learning mechanisms to efficiently coordinate the mobile crawler agents in order to reduce routing and searching costs and get benefits from cooperative caching.

Acknowledgements

The authors would like to thank the reviewers for their careful reviews and suggestions which greatly improved the quality of the paper.

References

[1] K A Amin and A R Mikler, Dynamic agent population in agent-based distance vector routing, Second International Workshop on Intelligent Systems Design and Applications, Atlanta, USA, August 2002, pp. 195-200.

[2] H Baala, O Flauzac, J Gaber, M Buid and T El-Ghazawi, A self-stabilizing distributed algorithm for spanning tree construction in wireless ad hoc networks, Journal of Parallel and Distributed Computing (JPDC), Vol. 63, No. 1, January 2003, pp. 97-104.

[3] M Bakhouya, Self-adaptive approach based on mobile agent and inspired by human immune system for service discovery in large scale networks, PhD Thesis No. 34, Universite de Technologies de Belfort-Montbeliard, 2005.

[4] M Bakhouya and J Gaber, Adaptive approach for the regulation of a mobile agent population in a distributed network, In 5th International Symposium on Parallel and Distributed Computing (ISPDC), Timisoara, Romania, IEEE Press, 6-9 July 2006.

[5] P Boldi, B Codenotti, M Santini and S Vigna, Trovatore: Towards a highly scalable distributed web crawler, In Poster Proceedings of the 10th International World Wide Web Conference, Hong Kong, China, 2001, pp. 140-141.

[6] P. Boldi, B. Codenotti, M. Santini and S. Vigna, Ubicrawler: A scalable fully distributed web crawler, Software: Practice & Experience, Vol. 34, No. 8, 2004, pp. 711-726.

[7] S Brin and L Page, The anatomy of a large-scale hypertextual Web search engine, In Proceedings of the 7th World Wide Web Conference, 1998, pp. 107-117.

[8] A Z Broder, M Najork and J L Wiener, Efficient URL caching for World Wide Web crawling, The Twelfth International World Wide Web Conference, May 2003, Budapest-Hungary, pp. 20-24.

[9] A Z Broder, Anna R Karlin, P Raghavan and E Upfal, Trading space for time in undirected s-t connectivity, ACM Symposium on Theory of Computing (STOC), 1989, pp. 543-549.

[10] D Chess, C Harrison and A Kershenbaum, Mobile agents: are they a good idea?, IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, 1995, available at http://www.eecs.harvard.edu/cs262/

[11] J Cho and H Garcia-Molina, Parallel crawlers, In Proceedings of the 11th International World Wide Web Conference, Hawaii-USA, 2002, ACM press, ISBN 1-58113-449-5, pp. 124-135.

[12] J Edwards, K S McCurley and J A Tomlin, An adaptive model for optimizing performance of an incremental web crawler, In Proceedings of the 10th International World Wide Web Conference, May 2001, pp. 106-113.

[13] J Fiedler and J Hammer, Using the web efficiently: mobile crawlers, In Proceedings of the Seventeenth AoM/IAoM International Conference on Computer Science, San Diego CA., 1999, Maximilian Press Publishers, pp. 324-329

[14] W Jansen and T Karygiannis, Mobile agent security, NIST Special Publication 800-19, National Institute of Standards and Technology, Computer Security Division,

Gaithersburg, MD 20899, Oct. 1999, available at http://csrc.nist.gov/publications/

[15] J Hammer and J Fiedler, Using mobile crawlers to search the web efficiently, International Journal of Computer and Information Science, Vol. 1, No. 1, 2000, pp. 36-58.

[16] A Heydon and M Najork, Mercator: A scalable extensible web crawler, World Wide Web, Vol. 2, No. 4, Dec. 1999, pp. 219-229.

[17] S Honiden, A Ohsuga, Y Tahara and M Hattori, Intelligent mobile agent system on the internet, In the proceeding of International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet, 2000.

[18] M Najork and A Heydon, High-performance web crawling, In J. Abello, P. Pardalos and M. Resende, editors, Handbook of Massive Data Sets, ISBN 1-4020-0489-3, Kluwer Academic Publishers, 2002, (pp. 25-45).

[19] G. Pant, P. Srinivasan and F Menczer, Crawling the Web, In M. Levene and A. Poulovassilis, editors: Web Dynamics, Springer-Verlag, 2003, pp. 153-178.

[20] O Papapetrou, Location aware web-crawling with the use of migrating crawlers, M.Sc. University of Cyprus, Dept. of Computer Science, PhD Thesis, Dec. 2003, available at http://www.l3s.de/~papapertou

[21] O Papapetrou, S Papastavrou and G Samaras, Distributed indexing of the web using migrating crawlers, In Proceedings of the Twelfth InternationalWorld Wide Web Conference (WWW), Budapest, Hungary, 2003.

[22] O Papapetrou, S Papastavrou and G Samaras, Ucymicra: Distributed indexing of the web using migrating crawlers, In Proceedings of 7th East-European Conference on Advanced Databases and Information Systems, Dresden, Germany, 2003, Springer, LNCS 2798, ISBN 3-540-20047-9, pp. 133-147.

[23] V Shkapenyuk and T Suel, Design and implementation of a high-performance distributed web crawler, In Proc. of International Conference on Data Engineering (ICDE), 2002, IEEE Press, pp. 357-368

[24] O Wittner, Nework simulator, In the Faculty of Information Technology, Mathematics and Electrical Engineering, Department of Telematics, 2000.