CMEM2001: Proceedings of the 10th International Conference on Computational Methods and Experimental Measurements, Y.V. Esteve, G.M. Carlomagno, and C.A. Brebbia, eds., June 2001. WIT Press, Ashurst, Southampton, UK, pp. 967-976.

Preliminary Study of Web Scouts/Foragers for a Bioinformatic Application: A Parallel Approach

R.L. Walker
Computer Science Department, University of California at Los Angeles, Los Angeles, California 90095-1596 USA
EMail: rwalker@cs.ucla.edu

Abstract

Adaptive Web scout/forager prototypes for an experimental Search Engine, Tocorime Apicu, based on a biological model and the movement of the sun are being developed using the methodologies of active networks, evolutionary computations, and scientific computing. This model has built-in mechanisms which allow it to adapt to an ever changing surrounding environment. This ever changing environment is similar to snapshots of Internet traffic which vary considerably based on 1) the time of day, 2) time zones, 3) various holiday and/or vacation patterns that exists throughout the world, and 4) ever occurring natural disasters. The sun can be viewed as a mechanism that provides order to what would appear as a very chaotic set of unrelated events over a period of time. The Internet prototypes emulate this biological model by incorporating their ability to detect and avoid network congestion. Network probes were deployed throughout the Internet in search of Internet Service Providers (ISPs) hosting HyperText Markup Language (HTML) services in order to initiate the process of developing customized routes for the retrieval of Web pages by Web scouts/foragers. Web scouts used the information obtained by the Web scout probes to detect network congestion. If network congestion is not detected in a given part of the world by a Web scout, additional Web scouts will be sent to other web site in the immediate area with the possibility of deploying foragers based on the abundance of information. This paper focuses on the development and implementation of the Web scout probes.

1 Introduction

Network probing software was developed to emulate the initial foraging and scouting behavior [7],[12],[4],[16][19] of the chosen biological system utilizing the methodologies of active networks and evolutionary computations. The social structure of this biological system contains communicating agents which possess built-in mechanisms for the survival of existing colonies. These survival techniques are used to 1) search for food, 2) mark territory, and 3) protect the individuals within the colony. Various techniques are employed to continuously disperse agents within the biological system so as to serve the needs of an existing colony in their location of ever changing food sources which are prone to change drastically over a relatively short period of time. This aspect of the biological model is being emulated to facilitate customized routing.

2 Related Work

The methodology of active networks (ANs) [22],[17],[6],[21] has been used for building and dynamically deploying network protocols. This approach was developed to support novel network services such as routing, caching, transcoding, combining, filtering, regulating, and processing transmitted packets. Applications associated with this methodology are 1) firewalls, 2) Web proxies, 3) nomadic computing, 4) routers, 5) transport gateways, 6) application services supporting video conference users, and voice and handwriting recognition, and 7) signaling facilities. Signaling facilities can be used as a network probe facility in which the user queries the status of networks for useful resource discovery, diagnostics, network monitoring, etc.

Various schemes for implementing communicating agent which can incorporate the methodologies associated with ANs have been developed using evolutionary computation methodologies [20], genetic algorithms (GA) and genetic programming (GP), which are needed to promote an efficient evolution of software agents which in turn can be used to locate HyperText Markup Language (HTML) hosts. Software agents [13] were developed for distributed network routing and restoration in which the agents were provided with the capabilities of 1) acting on their own, and 2) communicating with other agents. This scheme used genetically programmed agents to decentralize the control of network management by achieving partial control as opposed to complete control. Partial control redistributes responsibility away from the centralized controller to a host of partial controllers.

Navigation algorithms [10] utilizing/relying on the evolutionary process of GP were employed to construct broadcasting algorithms needed in the scheduling of agent actions resulting in the creation of a communication graph. The resulting graph reflected the need to schedule agent actions which lead to a system of paths which would in turn provide the minimum

broadcast time. The system of paths was created during runtime without thorough knowledge of the communication topology.

A search-for-service methodology [8] built on a load distribution system was used to retrieve complete information on the availability of resources within a local area network (LAN) as well as wide area networks (WANs). These mechanisms incorporate the ability to delegate jobs to random network host which in turn create network profiles within a LAN based on the rejection rate and chaotic traffic across the network. This approach builds non-deterministic WAN search mechanisms on the deterministic LAN search mechanisms. A search engine framework called NetMiner [3] studied the usage of data mining and information retrieval (IR) on the Internet for all types of multimedia files. This framework used the Internet directories supplied by the Internet Service Providers (ISPs) such as Yahoo, Excite NetDirectory, World Wide Yellow pages, and The Whole Internet Catalog as the starting point for information retrieval. Page-to-page drilling was used to follow each anchor (link) in a Web document until there were no new pages to retrieve – a process which may take a month or more because of the time required to build its corresponding data warehouse.

3 Development of the Web Probe Dispatchers

The efficiency of Internet applications is being tested by adding new applications which compete for the same network resources. The need for adaptive congestion control and avoidance at the application level is reflected in the side-effects of the current non-adaptive application mechanisms which reveal self-similarity among network transmissions. The exponential growth of Web documents, the incorporation of multimedia applications with real-time demands, and a steady increase in World Wide Web (WWW) users will lead to refinements in efficient design and implementation of probe/scout/foraging mechanisms needed by the information retrieval system associated with this project. The competition for bandwidth will reward the adaptive and efficient applications. The incorporation of AN methodologies enhances the development and incorporation of an adaptive biological model associated with this experimental Search Engine.

3.1 Rationale

The approach taken by Yahoo has some merit which is not apparent when viewing their strategy for Web page inclusion. Yahoo allows the user to provide suggested categories for newly submitted Web pages. A human editor eventually reviews the submittal form for Web page category placements, and Web crawlers are then released to traverse the hypertext links found within the submitted pages. Some advantages of this approach are: 1) links within the submitted page are accessible at the time of submittal, and 2) the overhead associated with locating this page as well as the other pages that are referenced within the submitted page is greatly reduced via

Table 1: IP addresses.

Class	IP address range	Number of possible host
A	0.x.x.x to 127.x.x.x	2^7 networks but each can support up to 2^{24} hosts (approximately 16 million)
B	128.0.x.x to 191.255.x.x	2^{14} networks but each can support up to 2^{16} hosts (approximately 65535)
C	192.0.1.x to 223.255.255.x	2^{21} networks but each can support up to 256 hosts

page-to-page drilling techniques. Some disadvantages of this approach are: 1) the scope of Yahoo's page retrieval scheme is limited by page designers who view Yahoo as a provider of adequate Web hosting facilities, 2) links within the submitted page may eventually expire or be relocated/rehosted with the same or different ISP provider, and 3) page-to-page drilling which may take months.

The initial step in this methodology evolves from the transmittance of Web scout probes to all ISPs in a manner similar to reliable flooding [2],[9],[15]. The probes provide the Web scouts with their results which reflect their maiden voyage to the various ISPs. Table 1 presents the possible number of Internet Protocol (IP) addresses supporting WWW services which is based on the class structures associated with the IPv4 and eventually, the IPv6 protocols. The current class structures reflect the three distinct network classes which have resulted in limitations on the dissemination of IP addresses. Class A can support up to 2^7 (approximately 128) networks, class B can support up to 2^{16} (approximately 16384) networks, and class C can support up to 2^{21} (approximately 2 million) networks. There are 2^{32} possible IP addresses since they are represented by a 32 bit number.

3.2 Role of the Web Probe Dispatchers

The characteristics of the dispatcher encompasses the essential features needed to release and coordinate the scout probes (network agents). Each dispatcher has a finite scope area which allows it to only see ISPs within a circle whose radius is given by a value V (its visibility) and whose center is the dispatcher. This approach provides a basis for a co-evolutionary methodology which is required to calculate a continuous assessments of the impact of GP robustness (brittleness) associated with the cooperative behavior and its effectiveness among communicating agents. The robustness of a GP program is defined as the ability of agents to cope with noisy or unknown situations (unknown test data) within a GP application when com-

munication among multiple agents is due to effective job separation. New and potentially improved behavior patterns are found to evolve through the use of a fitness measure associated with co-evolutionary strategies.

3.3 Limitations of the Web Dispatcher

The rapid release of a series of scout probes can have an adverse effect on the receiving host (ISP server) as well as on the sending host (dispatcher). The scout probes are created as child processes which can overtake the resources provided within the dispatcher's environment. The most efficient use of probes is related to the process of reliable flooding where monitors are used so to adequately control and coordinate valuable information returned by each probe. The ISP server may interpret the probes as a form of flooding which results in request being queued at the router level and/or server level. The worst case scenario results in extending the life of a probe beyond the amount of time allocated to it as a means of locating the ISP and retrieving valuable information. The dispatcher's resources become overloaded when too many probes have been allocated – thus preventing new processes from being stored in the process table until some of the outstanding ones have completed their probe of a selected ISP. The characteristics of the probes are

- built-in autonomy - it can work unaided;
- reactive - it keeps interacting with the environment within appropriate time limits;
- pro-active - capable of acting on their own;
- social - communicates with it's dispatcher which sends the information to Web scout/forager dispatcher(s);
- robust - ability to cope with the ever-changing network environment.

4 Testing the Web Probe Mechanisms

4.1 Releasing the Web Probes

The maiden voyage of Web probes to locate ISP hosting WWW services starts with the dispatcher generating approximately 2048 child processes on a continuous basis. This continuous set of processes are dispatched to randomly chosen IP addresses. A 2048 buffer is used to store the PID associated with each child process which in turn is used to track the life span of each probe allocated in accordance with a dynamic time-to-live (TTL) being based on the dispatcher's workload. The *sleep()* system call was used in this initial implementation to provide breaks in the dispatching of new scout probes so to emulate reliable flooding.

Table 2: Summary of Access Log for Web Scout Probe Dispatchers.

Item	Web Probes Version A
Access Log Duration	7 Days
Access Log Start Date	25 Jan 2001
Access Log Stop Date	01 Feb 2001
Total Requests	3213939.0
Avg Requests/ Day	459134.1
Number of Simultaneous Probes	1, 2, 4, or 8
Number of HTML Servers Located	21609

The purpose of randomly choosing ISPs – as opposed to performing sequential searches – is due to an attempt to avoid flooding an ISP which may support multiple Internet services via one host. The limitations of a fixed buffer size of 2048 will become more apparent when the Web scout probe/scout/forager dispatchers continuously transmits 1) probe scouts, 2) Web scouts, and 3) Web foragers. When testing only the dispatching of probes, the speed of the dispatcher's CPU does not play a factor in size of the buffer.

The integrated test of this approach reflects the dispatching process of the three previously mentioned components needed in this project as a means of adequately retrieving Web pages. The use of randomly chosen ISPs should provide the Web page indexers with a diverse set of Web pages reflecting the random selection of ISPs. A limitation of using randomly chosen Web hosting ISPs is revealed in the location of an ISP which hosts non-English Web pages. Filters will be needed during the developmental stages of this aspect of the experimental search engine in order to perform real time discovery/retrieval of Web pages, assess the demands placed on the ISP hosting this experimental search engine, and as certain the resource requirements of the LAN associated with the nodes in this distributed implementation. These filters play a vital role in reducing the impact of foreign languages on the fitness calculations used by the indexer when building the corresponding IR system.

The Web probes emulate a search-for-service strategy when locating existing as well as new ISPs hosting Web services. Finding these hosts, eliminates one problem, but other problems may occur when one of the following sources are found:

1. ISP responds as a Web server but unable to translate the IP address
2. An ISP access error

HTTP/1.0 403 Forbidden

Content-Type: text/html

<HEAD><TITLE>Error</title></head><body><h1>Error 403 </h1>Forbidden by proxy ACL check.</body>

3. Another ISP access error

   HTTP/1.1 403 ERROR

   Content-Type: text/html

   Content-Length: 154

   <h1>Error - 403</h1><HR><PRE>

   Failed to connect to server:

   162.89.0.100 (80)

4. An ISP provider may be offline for upgrades, servicing such backups, natural disasters, blackouts, etc.

4.2 Computational Measures

A feasibility study was conducted to determine a transmission rate for the release of the scout probes using 4 parallel dispatchers. The different dispatchers in this study, labeled Version A, released either 1, 2, 4, or 8 simultaneous probes. Each dispatcher was restricted to one of the previous numbers throughout the study. This study was also used to reveal any shortcomings associated with each dispatcher (node) maintaining full transmission buffers of 2048 probes. All of the probes targeted the same set of IP addresses, but due to the number of simultaneously released probes, the same ISP was not probed by all of dispatchers simultaneously. Table 3 present the results associated with this feasibility test.

The goal of this test was to probe 2^{22} IP addresses (approximately 4 million) in a 7-day period. This period of time was chosen since most search engines update their respected IR system every 7 to 10 days. A total of 21609 IP addresses were located that host some form of HTML service. The bests results – as reflected by the fitness measures – were obtained when 8 simultaneous probes were released. A projected usage is also shown which reflected the update of Version A resulting in Version B. This new version will require that each dispatcher patrol a distinct area of the Internet in which each transmits 8 simultaneous probes.

The fitness measure [10],[18] associated with developing Web scout probe/dispatcher algorithms was evaluated using

$$f = \frac{number\ of\ visited\ IP\ addresses}{time(seconds) * number\ of\ simultaneous\ probes}. \qquad (1)$$

Table 3: Version A: Web probe results for a 7-day period.

Number of Simultaneously Transmitted Probes	Number of Probes Released	Number of IP Addresses Located	Fitness	Cumulative Probability, P(M,i)	Number of Individuals Needed, I(M,i,z)
1	598738	4138	0.989977		
2	680697	4632	0.562746		
4	959220	6392	0.396503		
8	975284	6447	0.201572		
Totals	3213939	21609	0.354270	0.0005	1830629
Projected Probe Usage (Version B)	3901136	25788	0.201572	0.0006	1533971

By incorporating several different computational measures, the performance of the Web scout probe/dispatcher algorithms for this experimental search engine can be measured. The components of these measures include 1) the cumulative probability of success by generation i, $P(M,i)$, 2) the population size, M, and 3) a target probability, z. Each 7-day period is being treated as a generation. The cumulative probability is computed via the following equation:

$$P(M,i) = \frac{\sum_i \ HTML\ servers\ located}{total\ number\ of\ possible\ IP\ addresses}. \tag{2}$$

The number of individuals needed to produce a solution by generation i with probability greater than $z \approx 99\%$ is

$$I(M,i,z) = (i+1) \times \left\lceil \frac{log(1-z)}{log(1-P(M,i))} \right\rceil . \tag{3}$$

6 Conclusion

Future studies will provide a methodology for the development and implementation of adaptive Web agents (Web scout and crawler mechanisms) based on the methodologies of active networks, evolutionary computations, and the biological model for the experimental search engine. The statistical tools and methods for analyzing time series datasets from previous studies of Internet traffic will be used to organize data captured (provided by the Web scouts) at the packet level for network traffic between individual source/destination pairs. The proposed Web scouts/foragers will use the scout probe results combined with congestion control and avoidance to avoid areas of self-similarity within the Internet in their acquisition of Web documents.

7 Acknowledgements

The author wishes to thank Walter Karplus and Zhen-Su She for their direction and suggestions. This work was supported by the Raytheon Fellowship Program, Honeybee Technologies, and Tapicu, Inc.

References

[1] Arlitt, M.F. & Williamson, C.L. Internet Web Server: Workload Characterization And Performance Implications. *IEEE/ACM Transactions on Networking*, pp. 631-645, 1997.

[2] Bach, M.J. *The Design of the UNIX Operating System* Prentice Hall, Inc.: Englewood Cliffs, NJ, 1990.

[3] Cai, F.F. & Yu, Q. A Framework for Data Mining and Information Retrieval on the Internet. *Applications of High-Performance Computers in Engineering VI*, eds. M. Ingber, H. Power & C.A. Brebbia, WIT Press: Ashurst, Southampton, UK, pp. 467-476, 2000.

[4] Collett, T. Measuring Beelines to Food. *Science*, **287**, pp. 817-818, 2000.

[5] Diot, C., Huitema, C. & Turletti, T. *Multimedia Applications should be Adaptive.* Technical Report, INRIA Sophia: Antipolis, France, 1995.

[6] Hicks, M., Kakkar, P., Moore, J.T., Gunter, C.A. & Nettles, S. PLAN: A Packet Language for Active Networks. *Proc. of ICFP'98*, ACM Press: Baltimore, MD, pp. 86-93, 1998.

[7] Lindauer, M. *Communication Among Social Bees.* Harvard University Press: Cambridge, Massachusetts, 1961.

[8] Michtchenko, A. Search-for-Service Strategy and Integration of LAN into Web Operating System. *Proc. of the High Performance Computing Symposium - HPC 2000*, ed. A. Tentner, SCS Press: San Diego, CA, pp. 231-235", 2000.

[9] Peterson, L.L. & Davie, B.S. *Computer Networks: A Systems Approach*, Morgan Kaufmann Publishers, Inc.: San Francisco, 1996.

[10] Seredynski, F. Broadcasting and Spanning Trees in Interconnection Networks: Genetic Programming Approach. *Proc. of Parallel Computing 95: State of the Art Perspectives*, eds. E. D'Hollander, F.J. Peters, G.R. Jouber & D. Trystram, Elsevier Science Ltd.: Amsterdam, Netherlands, pp. 697-700, 1996.

[11] Shenker, S. Fundamental Design Issues of the Future Internet. *IEEE J. of Selected Areas in Communication*, pp. 1-24, 1995.

[12] Luke, S. & Spector, L. Evolving Teamwork and Coordination with Genetic Programming. *Proc. of the 1996 Genetic Programming Conf.* eds. J.R. Koza, D.E. Goldberg, D.B. Fogel, & R.L. Riolo, MIT Press: Cambridge, MA, pp. 150-156, 1996.

[13] Sinclair, M.C. & Shami, S.H. Evolving Simple Software Agents: Comparing Genetic Algorithm and Genetic Programming Performance. *Proc. of the 2nd Int. Conf. on Genetic Algorithms in Engineering Systems: Innovations and Applications*, IEE: London, UK, pp. 421-426, 1997.

[14] Stein, L.D. Introduction to Human Genome Computing Via the World Wide Web (Chapter 1). *Guide to Human Genome Computing (2nd edition)*, ed. M.J. Bishop, pp. 1-40, Academic Press: San Diego, CA, 1998.

[15] Stevens, W.R. *UNIX Network Programming, 2nd ed.* Prentice Hall, Inc.: Upper Saddle River, NJ, 1998.

[16] Srinivasan, M.V., Zhang, S., Altwein, M. & Tautz, J. Honeybee Navigation: Nature and Calibration of the "Odometer." *Science*, pp. 851-853, **287**, 2000.

[17] Tennenhouse, D.L. & Wetherall, D.J. Towards an Active Network Architecture. *ACM Computer Communications Review*, **26(2)**, 1996.

[18] Walker, R.L. Dynamic Load Balancing Model: Preliminary Assessment of a Biological Model for a Pseudo-Search Engine. *LNCS 1800*, eds. J. Rolim et al., Springer-Verlag: Berlin Heidelberg New York, pp. 620-627, 2000.

[19] Walker, R.L. Dynamic Load Balancing Model: Preliminary Results for Parallel Pseudo-Search Engine Indexers/Crawler Mechanisms Using MPI and Genetic Programming. *Proc. of VECPAR'2000. LNCS Series*, Springer-Verlag: Berlin Heidelberg New York, 2000. To appear.

[20] Walker, R.L. Search Engine Case Study: Searching the Web Using Genetic Programming and MPI. *Parallel Computing*, **27(1/2)**, pp. 71-89, 2001.

[21] Wetherall, D.J. Developing Network Protocols with the ANTS Toolkit. *Design Review*, 1997.

[22] Wetherall, D.J., Guttag, J.V. & Tennenhouse, D.L. ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols. *Proc. of the IEEE OpenArch'98*, IEEE Press: New York, 1998.