CNDS 2003: Proc. of Comm. Networks and Distributed Systems, SCS Press (Jan. 2003) 123–129.

Decentralized Approach to Information Discovery using Customized Routing

Reginald L. Walker Tapicu, Inc., P.O. Box 88492
Los Angeles, California 90009
rwalker@tapicu.com

Keywords: Information discovery, Customized routing, RS statistics, Stochastic optimization.

Abstract

Adaptive Web probe, scout, and forager prototypes for Tocorime Apicu HTML resource discovery system were developed based on the honeybee information discovery model. This model has built-in mechanisms that allow it to adapt to a changing environment. Emulators of these built-in mechanisms were applied to the ever-changing environment of Internet traffic which varies considerably, depending on: 1) time of day, 2) time zones, 3) various holiday and/or vacation patterns that exist throughout the world, and 4) naturally occurring disasters. The Internet prototypes emulate honeybee information search strategies for: locating forage sources, and detecting and avoiding foraging congestion. Specifically, the prototypes use rescaled adjusted range statistics to locate Internet service providers, retrieve information, and balance loads.

INTRODUCTION

The search strategies of the Tocorime Apicu [1] model [20] emulate unique characteristics of the honeybee model—its spatial organization along with species specificity, which in turn influences search strategies in the information ecosystem [17, 18]. Sets of probe, scout, forager, and page dispatchers form unique groups which are used to investigate the information ecosystem (WWW) in the process of retrieving and indexing its contents (see Figure 1). Each grouping is assigned a unique locale partition of the Internet to forage. Foraging groups—sets of probes, scouts, and foragers—are mapped to selected indexers of the Tocorime Apicu information sharing indexing (ISI) system [16, 19] which are responsible for data cleaning and intermixing the foraging results of the distinct locales using the information sharing model of honeybee colonies.

The decentralized approach reduces the inefficiency that is the result of overlapping the WWW search areas—this, in turn, leading to document duplication on a large scale. Restricting the duplication of pages within the Tocorime Apicu information sharing (IS) model is one of its design goals. Each dispatcher within its respective grouping communicates with one or two of the other dispatchers. The communication between the dispatchers allows each dispatcher to convey its view of the source it is foraging and of network congestion at various stages of the page retrieval process.

The dispatchers contain those features essential for releasing and coordinating the Web probes, scouts, and foragers. Each dispatcher has a finite scope, limiting its activity to those Internet service providers (ISPs) inscribed within a area whose radius is given by a value V (its visibility). This is shown in Figure 1.

THE OPTIMIZATION PROBLEM

Decentralized retrieval of Web pages located throughout the Internet combines adaptive solutions achieved by the various HTML Resource Discovery (HRD) system Web dispatchers [15, 14] and adaptive solutions to the network routing problem [9, 13]. The HRD system objectives include maximization of resource utilization and of overall LAN throughput. The dispatcher objectives are to minimize rejected request packets and guarantee quality of service (QoS). The routing procedure requires shortest path routing that minimizes "hops" between the source and randomly chosen ISPs. Factors that must be considered are connection requirements (end-to-end delay, delay variation, mean rate) and network conditions.

THE DISCOVERY PROCESS

Network probes are deployed throughout the Internet in search of ISPs hosting HTML services in order to initiate

[1] The word *Tocorime*, meaning "spirit"[2], comes from an ancient Amazon Indian language. *Apicu* comes from the Latin *apis cultura*, meaning "honeybee culture" or "the study of honeybees." The phrase *Tocorime Apicu* is used as "in the spirit of bee culture."

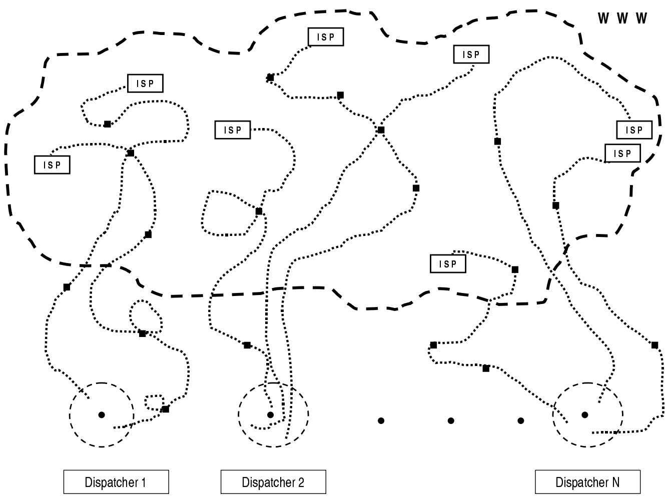

Figure 1: The WWW as an information ecosystem as viewed by Web dispatchers.

the development of customized routes for the retrieval of Web pages by Web scouts/foragers. Web scouts use the information obtained by the Web probes to detect network congestion. The various objectives just mentioned can be monitored by using rescaled adjusted range (RS) statistics [1, 6, 8]. It should be noted that RS statistic plots are normally used to monitor the self-similar nature of Web traffic for a single site, as opposed to simultaneous monitoring of multiple sites used in this adaptation of the honeybee information search strategies.

The Web probe dispatchers focus on locating ISPs hosting HTML services by using a search-for-service strategy. The probes form a crucial component of the Tocorime Apicu HRD system. Each located ISP is identified as useful by released probes if it provides the desired services. These marked sites are provided to the Web scout dispatchers. The probe dispatchers are not concerned with network congestion or any other aspect of page retrieval.

The Web scout dispatchers proceed with the site information provided by only its group probe dispatcher. Web scouts are released to each selected site periodically to gather and update Internet congestion traffic information. The scout dispatchers use this information on a per-site basis to determine the feasibility of retrieving information—Web pages—from a selected site. Site rankings are based on the results of the feasibility tests which use RS statistics to perform time series analysis on each site's Internet congestion data. The feasibility results for each site vary based on the time of day, time zone location with respect to the location of the HRD system, localized holiday and vacation patterns, and natural disasters. The periodic feasibility update of each site occurs within a random time period and is based on the workload of each scout dispatcher coupled with the update rate of the newly located sites provided by its corresponding probe dispatcher.

The forager dispatcher receives input from the scout dispatcher which makes retrieval decisions based on the conversion of congestion detection information into high-level congestion avoidance mechanisms before releasing foragers. The release of a forager can only occur if the scout dispatcher indicates that the feasibility results pass the QoS requirements imposed by the HRD system. This layer of congestion avoidance incorporates the information from mechanisms used to customize routes between the location of the HRD system and each selected ISP. These snapshots of source/destination traffic flow can change drastically over relatively short periods of time—depending on the release and return of each forager. The second layer of congestion avoidance is handled implicitly by Internet routers and switches between the source and destination.

GENERATING CUSTOMIZED ROUTES

Table 1 presents the navigational model associated with foraging individuals within the honeybee information sharing (IS) model [20]. The Web foragers needed for retrieving Web documents for the Tocorime Apicu IS system require some form of adaptive methodology since each Web scout searches for efficient paths (routes) to an uncongested source of information (Web documents) in order to build HRD ISP router tables. The purpose of periodically collecting time and path information related to the travel of scouts to and from ISPs is to find the most efficient Web

Table 1: Navigational model used by honeybee and Web foragers.

Computation	Honeybee navigational tactile translation	Information ecosystem (network) considerations
The direction of the foraging location	The angle between the direction of foraging location and the sun as measured by foragers	Randomly chosen location (IP address)
The transpose mechanism	Transpose the gravitational stimuli (pull of gravity) in the colony for the visual stimuli (landscape) in the field Vice versa	Customized routing - source to ISP route - ISP to source
Changes in the position of the sun	Transpose the angle in the dance	Network congestion based on time zones, time of day, and/or time of year (movement of the sun)
Basis for the estimate of distance	The time needed to reach the food source The effort needed to reach the food source (estimation of the head and tail winds)	Time-to-live (TTL) Number of hops

Table 2: Cumulative access log summaries for Web probe dispatchers.

Item	Web probe dispatchers		
	Version A	Version B	Version C
Start date	25 Jan 2001	05 Feb 2001	15 Oct 2001
Stop date	01 Feb 2001	05 Mar 2001	28 Jan 2002
Duration (days)	7 days	28 days	105 days
Duration (weeks)	1 Week	4 weeks	15 weeks
Simultaneous probe transmissions	4 to 32	32	128
Total requests	3213939.0	9057559.0	48193332.0
Avg requests per day	459134.1	323484.3	458984.1
HTML servers located	21609	46187	75367

page retrieval routes [11]. Because these routes exist only for short periods of time, the algorithm for the Web scout dispatcher(s) is:

1. Sort the jobs in the ready queue to reflect the scout dispatcher policies.
2. Create periodic tasks and place them in a global FIFO arrival queue.
    (a) Check the feasibility of a specified task set.
    (b) Reject infeasible tasks.
3. Accumulate RS statistical information needed for monitoring network congestion.
4. Generate a set of tasks for the Web forager dispatcher.

EXPERIMENTAL RESULTS

Experimental Environment

The goal of this study was to test the run-time environment associated with probe and scout dispatchers and determine the limitations in executing the HRD network probing software for extended periods of time. Three distinct versions of dispatchers were tested—Versions A, B, and C—and their results will be compared here. The HRD system was tested using HP Pavilions with seven 733MHz (20 Gigabytes of memory) and one 766 MHz (30 Gigabytes of memory) Intel Celeron processors, 128 MB SDRAM, and Intel Pro/100+ Server Adapter Ethernet cards, connected via two D-Link DSH-16 10/100 dual speed hubs with switches through a 144 Kbps router. The dispatcher tests were run using Red Hat Linux release 7.0 (Guiness).

Web Probe Dispatchers

A comparison of the composite log of the four dispatchers for Versions A, B, and C is presented in Table 2. Study results for each of the Web probe dispatchers provide a comparison of the weekly status of the four probe dispatchers for Version A [15] at the end of a week and Version B over a collection period of four distinct seven-day periods. Versions A and B released probes in sets of eight. Version C results were collected over a period of 15 weeks with each dispatcher releasing sets of 32 simultaneous probes. Each one-week period was treated as a collection period for all studies conducted. The goal of each Web probe dispatcher was to release approximately 1 million probes each week [15]—a time period during which cumulative Web probe dispatchers in these studies would release 4 million probes per week. This requirement was based on the results generated by Version A, which was tested for only one week. Version A is treated as the benchmark for all updates of the probe dispatcher.

Releasing the Web Probes in Version B

Table 2. shows the request results of Version B, which reflect the effects of memory leaks [3, 4, 5, 10] associated with each probe dispatcher as the number of collection days for the systems increased. Nodes 2 and 3 ran for the duration of the study of Version B which lasted 28 days. Nodes 1 and 4 were accidentally turned off at the end of the third week but were restarted after a down period of eight hours. This restart process occurred at the beginning of the fourth collection period.

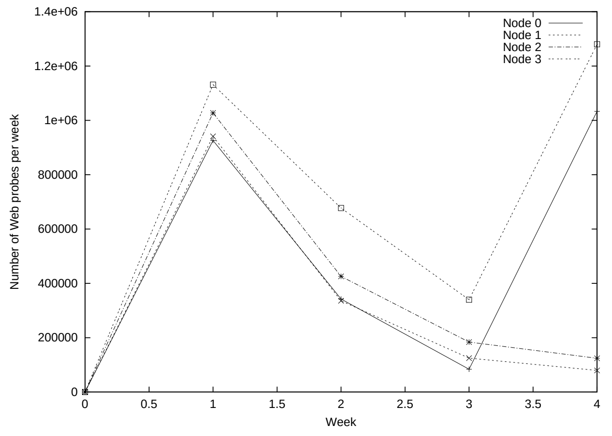

Figure 2: Weekly node access log for Web probes released (Version B).

Initially, the decrease in the number of transmitted probes appeared to have been caused by the ISP which hosted the Internet service associated with this project as shown in Figure 2. This figure presents the weekly performance of each Web dispatcher. The results generated during week 4 indicated that the ISP service worked as expected, but memory leaks appeared on each node (each node (computer) hosts a Web probe dispatcher) causing a denial of service within the LAN due to outstanding probes reflecting their extended TTL.

The nodes were idled at the end of the fourth week in order to allow all outstanding probes to return to their dispatcher, enabling us to determine if each computer system had the ability to stabilize (i.e., all released probes have returned to their dispatcher). After an eight-hour idle period, each node contained outstanding probes that did not terminate—this being a reflection of memory leaks in the probe dispatcher implementation. Node 0 had 1200 outstanding probes, node 1 had 1944, node 2 had 1895 outstanding probes, and node 3 had 888. These numbers reflected the degradation growth rate of the probe dispatcher over a period of weeks. The effect of these outstanding probes was an exponential decrease in the number of probes released per week; this appeared as an exponential backoff by the ISP hosting the Internet service for this application, and resulted in decreased throughput.

Releasing the Web Probes in Version C

The initial week, 15 Oct to 22 Oct, produced results for Web probe dispatchers only. During this week the dispatchers met approximately 88% of their target goal of releasing a composite of 4 million probes. The subsequent weeks of this study reflect a coupling of the probe, scout, and forager dispatchers on the same node (computer). The throughput of the probe dispatchers, reflected in its target ratio, shows the tight coupling of dispatchers on the same computer. Offline results for these dispatchers—loosely coupled—are the focus of this work and are presented for a two-week period, Weeks 16 and 17, using four nodes dedicated to these tasks.

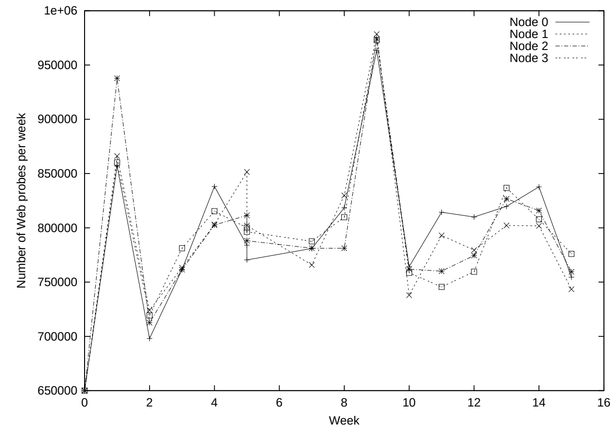

Figure 3: Weekly node access log for Web probes released (Version C).

Week 2 showed throughput fluctuations from the previous week and the week of Halloween. When viewing the first three weeks of this study, it appears as if the HRD system is suffering from performance degradation because of memory leaks. There was approximately an overall decrease of 21% from the target ratio. This shows that Internet user activities increased during the week of Halloween (Week 3).

The next event considered was Thanksgiving with system coverage beginning the week of Halloween. Weeks 4 and 5 show increases in system throughput reflecting possible reductions in user activities because holiday seasons affect U.S. Internet traffic. Week 6 shows a decrease in throughput from Week 5 but an increase over Week 4. The end of this collection period showed approximately a 20.4% difference between the results and the target ratio. The presence of traffic congestion is depicted in the charts by consistent throughput measurements of the number of released probes.

The next major holiday period considered was Christmas/New Years with a collection period beginning Week 7 and terminating Week 12. The starting period reflects a reduction in user traffic following Thanksgiving. The optimal throughput for this period occurred during Week 9, which represents the middle of the collection period. The optimal throughput for the collection period containing Thanksgiving occurred near the middle of the period. The final week of this collection period showed a 19.56% difference from the target ratio.

The final event of this composite collection period was the Super Bowl weekend, a major event in the U.S. The time period between Christmas and New Year is short, and this is reflected in the data collection period associated with this system. The collection period started in Week 12 and terminated with Week 15. The optimal throughput

Table 3: Web scout dispatcher results for customized routing (Version B).

ISP response results	Web scout dispatcher Node 0	Node 1	Node 2	Node 3	Totals
Number of QoS requests (based on RS statistic plots)	12012	29392	29610	4248	75262
Periodic tasks (recurring QoS requests)	8094	26985	27340	3814	66233
Number of customized routes (successful retrievals)	1039	1653	1533	319	4544

Table 4: Web scout dispatcher results for customized routing (Version C).

ISP response results	Web scout dispatcher Node 0	Node 1	Node 2	Node 3	Totals
Number of QoS requests (based on RS statistic plots)	9502	9180	9587	9326	37595
Periodic tasks (recurring QoS requests)	240	2406	2766	3456	8868
Number of customized routes (successful retrievals)	1422	1127	1003	1112	4664

occurred near the median point of this collection period which reflected patterns seen in the previous periods. The final week of this period showed a 19.68% deviation from the target ratio.

All of the collection periods contain localized optima that gradually decrease, followed by gradual increases in user activities until the occurrence of the next event is initially acknowledged by Internet users. Figure 3 shows each probe dispatcher's contribution to the aforementioned results. Local maxima occurred during Weeks 1 and 9. Local minima occurred simultaneously across all dispatchers during Weeks 2 and 15.

Web Scout Dispatchers

The ISP responses associated with requesting customized routes reflect: 1) periodic tasks resulting in recurring requests, 2) successful retrieval which is selected by calculated RS statistic measurements per ISP, and 3) time-outs due to congestion and/or no response.

Releasing the Web Scouts

Table 3 provides sets of ISP responses corresponding to the four dispatchers used in Version B. These results are only associated with the Web scouts and their corresponding dispatchers. This uneven distribution of possible ISPs probably is a result of the random selection of tested IP addresses. This disparity is also evident in Figure 2 where the four Web probe dispatchers were traced on a weekly basis. The number of customized routes forms the interface between the scout and forager dispatchers.

The periodic tasks (recurring QoS requests) result from the RS statistic measurements failing to meet the QoS threshold required in this discovery system, which was imposed in order to avoid adverse impacts of probes on available network resources. The number of recurring QoS requests reflects repeated attempts to obtain the desired QoS between the source (dispatcher) and the known/responding HTML servers. The unsuccessful attempts are reinserted at the end of the waiting list.

Application of RS Statistics

Figure 4 depicts a sample of the feasibility test measures of Version B for each of the Web scouts associated with each selected and/or periodic tasks (ISPs). The RS statistic results in these figures differ from the references used as the basis of this approach since the larger RS statistics value implies a smaller degree of self-similarity. RS statistics measurements are periodically used to determine the degree of congestion along the path to a chosen ISP, as well as within the chosen ISP. The retrieval of raw data files removes the ISP from the lists of periodic tasks which are then retested. The feasibility tests are based on the release of scouts to each ISP in order to perform the self-similarity computations. A congestion threshold was used to determine feasibility results, indicating 4544 customized routes. All of the customized routes contained at least one scout that did not return within its allocated TTL. These timeouts reflect possible congestion in the LAN, LAN+WAN, or WAN.

A comparison of the Version B self-similarity figures shows that the scouts associated with dispatchers 0 and 3 perform in a similar manner. Likewise, the scouts associated with dispatchers 1 and 2 produced similar results. The disparity in results stems from the accidental restart of nodes 0 and 3. However, each revealed that several of the periodic QoS requests were unsuccessful. The QoS threshold is shown in the figures as 50000. This value is apparent in the results for dispatchers 1 and 2.

The scout dispatcher results associated with Version C

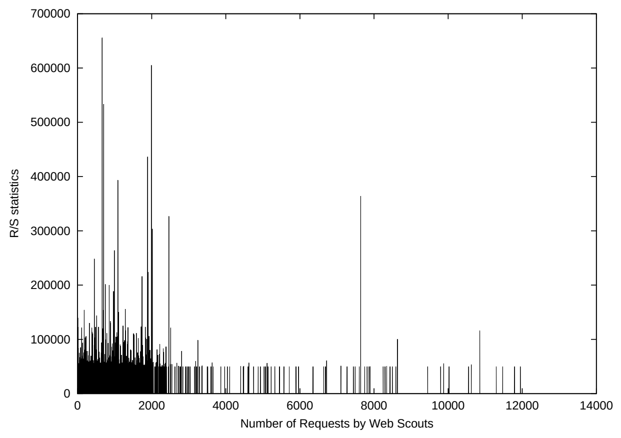

Figure 4: Self-similarity measures for Web scout dispatcher 0—Version B.

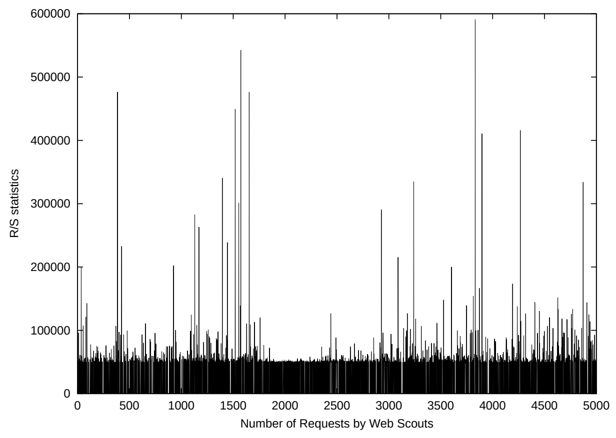

Figure 5: Self-similarity measures for Web scout dispatcher 0—Version C (Week 16).

are shown in Table 4. The increase—compared to Version B results—is due to the extended collection period and dispatcher improvements. The two-week collection period resulted in a limited number of periodic tasks for scout dispatcher 0 only. The periodic task of each dispatcher was not a significant contributor to the number of customized routes.

The corresponding number of QoS requests was approximately equivalent for all of the dispatchers despite the disparity in the number of periodic tasks. Dispatchers 0 and 2 submitted the most QoS requests followed by dispatcher 1 and subsequently, dispatcher 3. Dispatcher 0 generated the largest number of customized routes. The other dispatchers generated results that were approximately equivalent.

The RS statistic results for each dispatcher were partitioned into two figures, with the first figure of each pair depicting the first 5000 QoS request sets as shown in Figures 5 and 6. The accompanying figures in each pair present the remaining QoS requests. The combination of these two figures reveal that a large number of requests resulted in an ISP responding to a QoS below the required threshold. All of the figures associated with the detection of congestion (RS statistic measurements) contain areas which reflect no response from a selected ISP. This indicates a HTML server which provided a slow response due to congestion.

RELATED WORK

An admission control function [8] was used to estimate the "available bandwidth" between two end points of a network. This quantity was estimated using RS statistics so to determine whether requested telecommunication services could be established.

[7] developed a methodology using the simple network management protocol (SNMP) to query the status of networking hardware. An adaptive network performance prediction function was used to take a bottleneck link of the Internet as input and predict future performance over the link. SNMP provided a methodology with the IP routing topology/tables using software similar to the *traceroute*() program [12].

SUMMARY

Web prototypes were developed as active processes using honeybee search strategies based on the honeybee information discovery model. These prototypes are required to supply the Tocorime Apicu ISI system with Web pages generated by foraging the Internet in search of ISPs hosting HTML services. Locating these ISPs constitutes one aspect of fulfilling the request/needs of the Tocorime Apicu IS system with Web pages to index the WWW. Congestion and avoidance mechanisms required for efficient retrieval of remote pages of varying size were incorporated in the second phase of the Web foraging process. Customized routes are implicitly generated by the routers and switches in a effort to increase traffic flow for requested Web pages. The final component of the retrieval process is delayed in favor of generating customized routes that meet the required QoS that is used as a threshold in the second foraging phase. The actual retrieval of Web pages in their raw format is conducted using information generated in the first two phases of this foraging methodology. The retrieval of raw pages is the final component of the HRD system but constitutes the initial component of the ISI system by implementing the first pass of a two-pass document parser.

ACKNOWLEDGEMENTS

The author wishes to thank D. Stott Parker and Gary B. Fogel for their direction and suggestions. The author

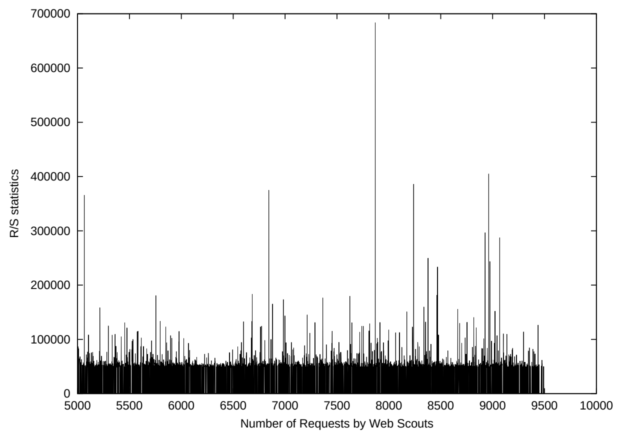

Figure 6: Self-similarity measures for Web scout dispatcher 0—Version C(Week 17).

wishes to express his gratitude to the reviewers whose detailed and useful comments helped tremendously to improve the quality of this paper. This work was supported by Honeybee Technologies and Tapicu, Inc.

References

[1] M.E. Crovella and A. Bestavros. Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes. *IEEE/ACM Transactions on Networking*, pages 1–25, 1997.

[2] P. Fritsch. Five Mellow Guys Follow Their Dream: A 'Tall Ship' in Brazil. *The Wall Street Journal*, CXLII(35):1, Friday, February 18, 2000.

[3] L. Gillie. Conversations on network-related topics. Raytheon Corporation, El Segundo, CA, 2001.

[4] C.S. Kennedy. Conversations on network-related topics. Raytheon Corporation, El Segundo, CA, 2001.

[5] V.C. Kirk. Conversations on network-related topics. Raytheon Corporation, El Segundo, CA, 2001.

[6] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson. On the Self-Similar Nature of Ethernet Traffic. In *Proceedings of ACM SIGComm '93*, pages 1–11, 1993.

[7] B. Lowekamp, D. O'Hallaron, and T. Gross. Direct Queries for Discovering Network Resource Properties in a Distributed Environment. *Clustering Computing*, 3:281–291, 2000.

[8] P. Maryni and F. Davoli. Load Estimation and Control in Best-Effort Network Domains. *Journal of Network and Systems Management*, 8(4):527–541, 2000.

[9] H.G. Sandalidis, K. Mavromoustakis, and P. Stavroulakis. Performance Measures of an Ant Based Decentralised Routing Scheme for Circuit Switching Communication Networks. *Soft Computing*, 5:313–317, 2001.

[10] J. Simpson. Conversations on network-related topics. Raytheon Corporation, El Segundo, CA, 2001.

[11] M.V. Srinivasan, S. Zhang, M. Altwein, and J. Tautz. Honeybee Navigation: Nature and Calibration of the 'Odometer'. *Science*, 287:851–853, 2000.

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

[13] A.V. Vasilakos, K.G. Anagnostakis, and W. Pedrycz. Application of Computational Intelligence Techniques in Active Networks. *Soft Computing*, 5:264–271, 2001.

[14] R.L. Walker. Dynamic Load Balancing Model: Preliminary Results for Parallel Pseudo-Search Engine Indexers/Crawler Mechanisms Using MPI and Genetic Programming. In *VECPAR'2000 , LNCS 1981*, pages 61–74. Springer-Verlag, Berlin, 2000.

[15] R.L. Walker. Preliminary Study of Web Scouts/Foragers for a Bioinformatic Application: A Parallel Approach. In Y.V. Esteve, G.M. Carlomagno, and C.A. Brebbia, editors, *Computational Methods and Experimental Measurements X*, pages 967–976. WIT Press, 2001.

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

[17] R.L. Walker. Applying Evolutionary Computation Methodologies for Search Engine Development. In L. Wang, K.C. Tan, T. Furhashi, J. Kim, and X. Yao, editors, *SEAL'02: Proceedings of the 2002 Asia-Pacific Conference on Simulated Evolution and Learning*, pages 208–213, Singapore, November 2002. Nanyang Technological University Press.

[18] R.L. Walker. Simulating an Information Ecosystem within the WWW. In A. Abraham, J. Ruiz del Solar, and M. Koppen, editors, *Soft Computing Systems: Design, Management and Applications*, pages 891–900, IOS Press, Amsterdam, December 2002.

[19] R.L. Walker. Using Nearest Neighbors to Discover Web Page Similarities. In H.R. Arabnia, editor, *PDPTA'02: Proceedings of the 2002 International Conference on Parallel and Distributed Processing Techniques and Applications*, pages 157–163. CSREA Press, June 2002.

[20] R.L. Walker. *Tocorime Apicu: Design of an Experimental Search Engine using an Information Sharing Model.* Dissertation, Publication Pending, ProQuest Information and Learning (formerly UMI), Ann Arbor, MI, 2003.