

Parallel Adaptive Mobile Web Clipping

- Abstract -

We describe a unique approach to improving the performance of a Web clipping portal by exploiting inherent parallelism in the syntax of widely used markup languages, and by employing a parallel computing platform as an in-line proxy between the handheld mobile device and a Web server on the Internet.

1. INTRODUCTION

Web content is now an indispensable tool for businesses and individuals alike. There is an even greater potential benefit for mobile Internet access via handheld cellular communications devices, if only some seemingly insurmountable technical difficulties could be overcome. Users conditioned to accessing the Web from powerful computers, over high-speed Internet connections, often expect to view a tiny Web page on their cellular telephones or their personal digital assistants. This level of service is not possible with today's technology, nor is it likely to be available soon. How can mobile Internet users access Web content from a handheld device and get adequate service?

Generalized in-stream solutions are most often proposed as a separate computer system, called a gateway, portal, proxy system, or *clipping* server. Application-specific processes, such as OreOs [1], run on an in-stream proxy server. Other examples are Mowgli [2], Pythia [3], Glomop [4], Digestor [5], MOWSER [6], eNetwork [7], Power Browser [8], and others [9], [10] and [11].

We discovered one proxy server approach that other researchers seemed to have missed: HTML is used to describe Web server content, while other markup languages (WML and HDML, e.g.) describe how to display similar data on a handheld device. If HTML and other markup languages have a tree-like syntax structure, and converting one language into another can be done on one branch independently of each of the others, then there is parallelism that can be exploited in this process. We propose the use of a parallel computing architecture as a proxy server, and a modular parallel software architecture for the process. Such a cluster server should outperform the equivalent single-system server by performing several

conversions simultaneously. The cost, of course, is the networking overhead. Each branch of an incoming HTML structure must be sent to an available worker node for independent conversion, and sent again to a joiner node for assembly into the outgoing markup language format.

2. EXPERIMENTAL APPROACH

Figure 1 shows our clipping server in relation to support systems on the network. We employ Apache as a local Web server [12]. An external performance monitor gives the researcher at-a-glance information about network resource utilization levels, and a suite of client software processes exercise the system in either a single-query, or a query stream mode. We also have the ability to control both the statistical distribution and the inter-arrival rate of the query stream [13].

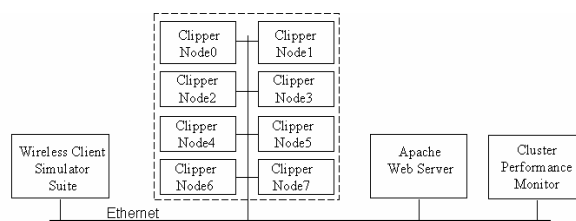


Fig. 1. Our test bed architecture showing the clipping server cluster in relation to external systems.

Clipping server software is designed with the parallel hardware system in mind. If an incoming HTML packet arrives, gets disassembled into its component parts for conversion, then reassembled before passing the result along to a handheld device, the software architecture in Figure 2 may be used.

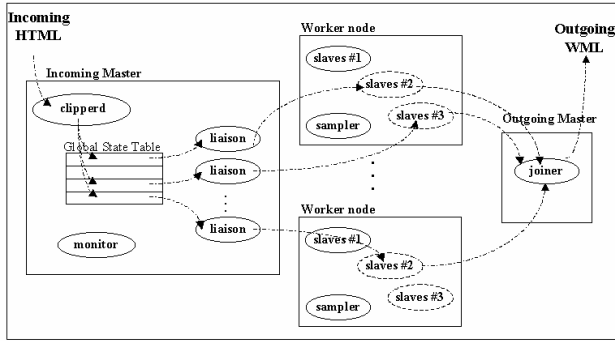


Fig. 2. Our test bed software shows the incoming query handler, several worker processes and a joiner.

Our software allows for multiple *slave* processes on each worker node, which may be adjusted during performance measurements to determine the benefit from an adaptive system, which would change the number dynamically in response to varying loads. Resource utilization data are collected at each node by the *sampler* processes and sent to the *monitor* process, which keeps a summary in a local table. The main process selects a free slave on the least-busy worker node per the Global State Table.

3. EXPERIMENTAL RESULTS

Figure 3 is a plot of system performance in the cluster's various configurations: added processing power is balanced by added system overhead. We developed a methodology allowing us to write software for a single system, while hiding the sophistications of a cluster server environment inside our MSI (Manifold Server Interface), an application program interface [14]. This gave us a *fair* and comparable single-node value.

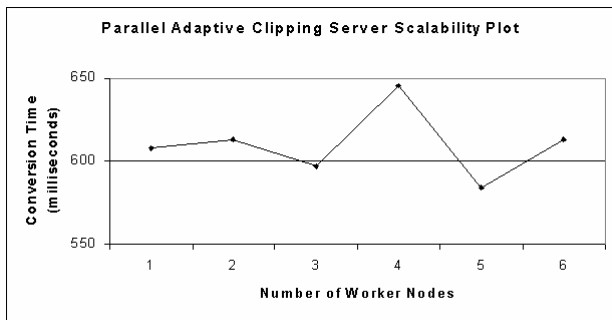


Figure 3. Scalability analysis from our performance data.

A cursory examination of this plot might prove the 5-node configuration as optimal, but the gradual descent of the final points suggests that the response times are headed lower still. This leads us to suspect that a larger cluster may reveal even better scalability than that of our test bed.

4. CONCLUSIONS

Our objective was to draw attention to inherent parallelism in the syntactical structure of popular markup languages, like HTML, WML and HDML. Our data show that the distributed system version of our clipping server outperformed the single system version in most cases; we suspect network loading was a limiting factor in the cases where it did not. Future researchers may want to expand the capabilities of their test bed to extend beyond the optimal point of scalability. They may also want to consider some form of data compression to reduce the network saturation at higher loads, or perhaps consider a faster external network.

5. REFERENCES

- [1] C. Brooks, M. S. Mazur, S. Meeks and J. Miller, "Application-Specific Proxy Servers as HTTP Stream Transducers," in *Proc. 4th Int'l. WWW Conf.*, 1995.
- [2] M. Liljeberg, H. Helin, M. Kojo and K. Raatikainen. (1996, Apr.). Enhanced Services for World-Wide Web in Mobile WAN Environment. University of Helsinki Department of Computer Science. Helsinki, Finland. Tech Report C-1996-28.
- [3] A. Fox and E. A. Brewer, "Reducing WWW Latency and Bandwidth Requirements by Real-Time Distillation," in *Proc. 5th Int'l. WWW Conf.*, 1996.
- [4] A. Fox, S. D. Gribble, E. A. Brewer and E. Amir, "Adapting to Network and Client Variability via On-Demand Dynamic Distillation," *Operating Systems Review*, vol. 30, no. 5, pp. 160-170, Oct. 1996.
- [5] T. W. Bickmore and B. N. Schilit, "Digstor: Device-Independent Access to the World Wide

- Web,” in Proc. 6th WWW Conf., 1997, pp.655-663.
- [6] H. Bharadvaj, A. Joshi and S. Auelphan-wiriyakul, “An Active Transcoding Proxy to Support Mobile Web Access,” in *Proc. 17th IEEE Symp. On Reliable Distributed Systems*, 1998, pp. 118-123.
- [7] B. C. Housel and D. B. Lindquist, “WebExpress: A System for Optimizing Web Browsing in a Wireless Environment,” in *Proc. 2nd Int’l. Conf. On Mobile Computing and Networking*, 1996, pp. 108-116.
- [8] O. Buyukkocuten, H. Garcia-Molina, A. Paepcke and T. Winograd, “Power Browser: Efficient Web Browsing for PDAs,” in *Proc. Conf. on Human Factors in Computing Systems*, Apr. 2000, pp. 430-437.
- [9] K. Ham, S. Jung, S. Yang, H. Lee and K. Chung, “Wireless-Adaptation of WWW Content over CDMA,” in *Proc. 6th IEEE Int’l. Workshop on Mobile Multimedia Communications*, 1999, pp. 368-372.
- [10] K. Kim, H. Lee and K. Chung, “A Distributed Server System for Wireless Mobile Web Service,” in *Proc. 15th Int’l. Conf. on Information Networking*, 2001, pp. 749-754.
- [11] B. Knutsson, H. Lu and J. Mogul, “Architecture and pragmatics of server-directed transcoding,” in *Proc. 7th Workshop on Web Content Caching and Distribution*, 2002.
- [12] G. Holden, N. Wells and M. Keller, *Apache Server Commentary*, Scottsdale, AZ: Coriolis Press, 1999.
- [13] A. Vrenios, *Linux Cluster Architecture*. Indianapolis, IN: Sams Publishing, 2002.
- [14] A. Vrenios, S. Panchanathan, and F. Golshani. “Converting Software from a Single System, Concurrent Server to a Cluster Server Architecture.” *Proc. Int’l Conference on Principles of Distributed Systems – OPODIS 2001*, Dec. 10-12, 2001. Manzanillo, Mexico. (CD-ROM).