Geographically Distributed Web Servers under Limited Network Bandwidth[*]

Jirasak Lapanalarploy and Vara Varavithaya
Department of Electrical Engineering
King Mongkut's Institute of Technology North Bangkok
1518 Piboonsongkram Rd., Bangsue Bangkok 10800, Thailand.
Phone (662) 587-0027 Fax 585-7350
E-mail: `vara@hpc.ee.kmitnb.ac.th`

Abstract

A cluster of World Wide Web servers has been used to serve large amount of requests presented today in the Internet. New Web applications have driven even higher demand on communication bandwidth. Server farms and geographically distributed server are considered as two major architectures for high performance WWW services. The dispatched mechanisms used to balance requests among the servers include IP-rewriting, DNS dispatched, HTTP redirection. Due to congested network, quality of WWW services can be very poor. In many case, a crucial bottleneck in the network path is the last-mile lease line connected between an ISP and a Web server. In this paper, we study the geographically distributed web servers under limited network bandwidth. The DNS dispatched mechanism are adopted to balance the Web requests. Together with server utilization, we proposed that the network delay should be considered in the dispatching process. The weighted load sharing algorithm for server utilization and network load is presented. The performance of the system were evaluated through simulation. The results show that, under congested network situation, an average response time of a Web request is improved using geographically distributed web server especially in heterogenous environment.
Keywords: Internet, Distributed Web Server, Load Balancing, Distributed Systems.

1. Introduction

The Internet has emerged as a practical solution for global scale internetworking. The most popular applications run on the Internet are based on the World Wide Web (WWW) technology [1]. The WWW allows information services among remote sites which adopt a client-server service model. As the usage of the WWW explosively increases, popular WWW servers might be overloaded or part of the networks could be overly congested due to large number of WWW

[*]This research was supported in parts by Engineering Research Funds, King Mongkut's Institute of Technology North Bangkok.

transactions [2]. The large number of requests can be serviced using either a single high performance server or multiple servers [3]. A single high-end server is easily managed but do not scale well. Therefore, high throughput Web systems are implemented as distributed servers architecture. The distributed Web servers can be classified as a *server farm* and a *geographically distributed server system* [2, 4].

A server farm consists of a cluster of Web servers resided in the same physical location. In geographically distributed servers, the Web servers are located in different sites. A Web server system at any particular site could be either a single server or a server farm. The web contents are replicated among the servers. The server farm is more convenience in terms of maintaining information consistency, system administration, and security. The geographically distributed servers can provide higher level of availability due to isolated physical locations. The Web server systems is called homogeneous if all Web servers in the distributed system are identical. A homogeneous web server system might not be practical since a higher performance with different capacity server is added to the system when more processing power is needed. Hence, contemporary Web server systems usually compose of heterogenous hardware components which offer different levels of performance.

At the first phase of WWW request, the logical web server name (URL) is translated an IP address using *Domain Name Service* (DNS). The client asks its local name server (LNS) for a specific URL. The LNS checks the request URL in the local name cache. If the URL-IP mapping entry is not presented in the cache, the mapping request is forwarded to the higher level of name servers. If there is no intermediate caching in the process of DNS resolutions, the URL mapping request will eventually arrive the cluster domain name server (CNS) which belong to the Web server systems. The CNS returns the requested IP address to the LNS and then to the client, respectively. In distributed Web server system, the CNS assigns different IP address to a single URL to share workload. Several approaches were proposed to achieve load balancing among the servers. In [2], the CNS assigns the URL-IP mapping to a set of servers in round robin (RR) fashion.

An important performance metric in the Web system is an average user response time. The response time consists two major components, network delay and server processing delay. The server processing delay is the time incurred in preparing web contents. This delay depends on the nature of documents. The document processing times for html file is an order of magnitude less than dynamic pages such as *asp* and *php* documents. A secure transaction require lots of computing resources at the server side. Electronic commerce and personal web services are based on dynamic pages and secure transaction technologies which increase server processing delay substantially. New multimedia Web applications, such as audio and video, consume large amount network bandwidth and keep increasing. The enhancement of the Internet infrastructure will never keep up with the user demand on bandwidth. Therefore server processing power and network bandwidth are the two crucial bottlenecks need to be considered in the high performance Web server systems.

In Thailand, the Internet infrastructure has been slowly developed compared to the user demands. Many organizations have a low speed connection(s), 64 kbps or 128 kbps, to the Internet. The Web server is located behind the low-speed link. It is possible that some of these organizations can cooperated on geographically distributed Web servers to enhance performance and availability. Examples include high schools and government offices. As previously mention, require processing power and amount of information transfer per HTTP request vary widely. The load balancing algorithm need to considered both server load and network load together. In this paper, the parameters used in load sharing in geographically distributed Web servers are studied. We proposed that the network load and the server utilization are equally important parameters in load balancing decision. The server selection algorithm is proposed in the paper. The algorithm is based on server utilization and network load to be used as the server cost of the particular server site. An DNS-based approach is used to distribute requests. This work is different from the works in [5] since the algorithm in [5] uses only a constant network cost in homogeneous environment. The performance of the proposed algorithm is validated using simulation. The results show that the proposed scheme perform significantly better than the Round Robin (RR) and Lowest Utilization (LU) schemes especially in heterogenous environment.

This paper is organized as follows. System model is described in Section 2. Section 3 presents a proposed load balancing algorithm for geographically distributed Web Servers under limited bandwidth. The simulation results are discussed in Section 4 and conclusion remarks and future works are given in Section 5.

2. DNS-Based Distributed Web Server Systems

The simplest form of distributed web servers is the mirror Web sites. The main web page provides multiple Uniform Resource Locators (URLs) that link to mirror sites. A server list can be dynamically updated using dynamic pages. The server is selected based on both server utilization (dynamic web page) and geographical information (user selects). The main drawback of the mirror sites approach is lack of single interface to the Web services. An important problem in distributed Web servers is how to schedule requests to multiple servers while a single URL interface is preserved. The proposed scheduling mechanisms includes DNS-based, IP-Rewriting, Request forwarding (redirection server), and HTTP redirections. An excellent survey on this subject can be found in [6]. To share the workload, HTTP requests are scheduled to multiple web servers in distributed web server systems. Figure 1 (a) shows a server farm architecture where all servers are located at the same site. The server farm can be considered as a subset of geographically distributed server architecture, as shown in Figure 1 (b). System performance, availability, scalability are the main issues need to be considered in designing the distributed web server systems. In addition, a user friendly system requires a single interface to these servers.

An entity in the Internet is usually referred to in form of logical name, such as hostname, domain name, and URL, since a logical name is more suitable for human to remember. In this perspective, the Internet can be viewed as a collection of domains connected to the Internet core. The mapping between the logical name and the IP address is accomplished via DNS mechanism. Each domain has its own local name server (LNS). All mapping requests inside the domain must be processed through the LNS(s). Since the domain name system was designed in hierarchical structure, mapping requests might be forwarded through several intermediate name servers (INS). The distributed web servers maintain URL-IP mapping in their cluster name server (C-NS). A single URL is mapped to multiple IP addresses that belong to the servers in the pool. For the system with N Web servers, the CNS selects appropriate server i and returns IP_i where $(0, \ldots, i, \ldots, N-1)$ for each DNS request.

Figure 2 shows the DNS mechanism in the distributed Web server system. A client first sends a DNS request through a series of INS. The DNS request is then forwarded to the CNS. The load sharing algorithm in the CNS selects server node i in the second step. The IP_i is returned

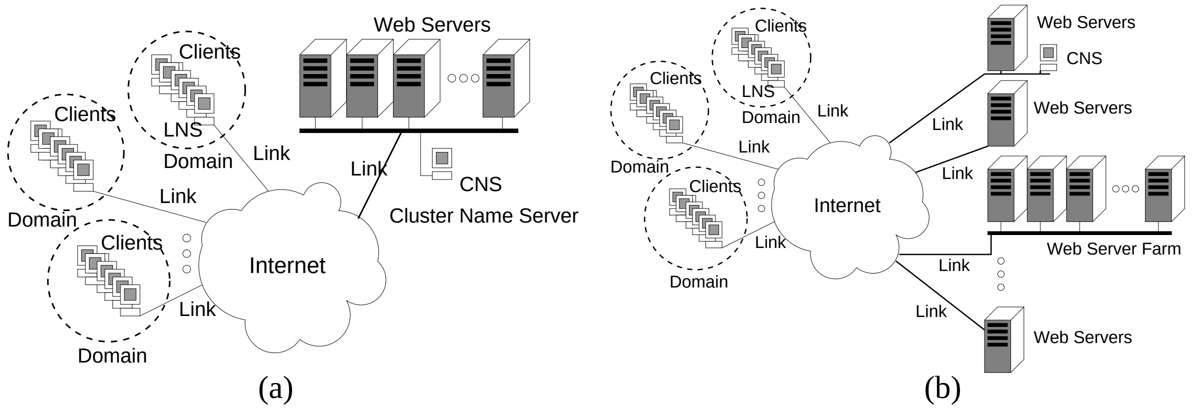


Figure 1. Distributed server architectures: (a) WWW server farm (b) Geographically distributed server.

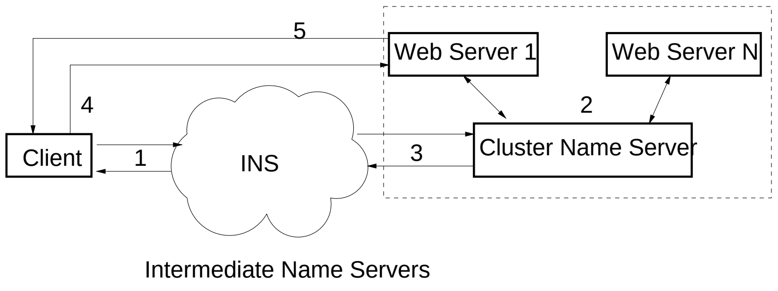


Figure 2. DNS-based load sharing in distributed Web server.

to the client in the third step. After obtaining the IP address of the Web server, the client starts sending HTTP requests to the Web Server and receiving Web contents in the forth step and the fifth step, respectively. The DNS allows INS to cache the name-IP mapping. The caching creates undesirable effects to the load sharing performance. The CNS needs to set the TTL value to a relatively small value to reduce this effect.

In current WWW traffic behavior, it has been observed that almost 75 percent of the requests come from only 10 percent of the domains [2, 7]. This traffic behavior can be represented using the Zipf Distribution [8]. The distributed Web-server system that use domain-based round robin scheduling might have uneven load distribution. Because of domain behavior and caching effects, the DNS-based approaches cannot provide total control on Web request scheduling. Several approaches have been proposed to improve the performance of DNS-based schemes. These approaches include weighted round robin and multi-tier round robin algorithms [8, 9].

Instead of RR scheduling, a more sophisticate scheduling algorithm can be implemented to assist the CNS in URL-IP assignments. To evenly balance the requests, server utilization, overload alarming, server status, number of specific domain's requests were also be considered [8, 9], i.e., Lowest Utilization (LU) algorithm. An DNS reply message consists of URL-IP mapping and TTL value.

The CNS can have total control on the Web request by setting the TTL equal to zero. However, DNS request/reply traffic might overly congest the network and some intermediate name servers might not cooperate in discarding TTL with a very small value. In [4], different value of TTL of the URL-IP mapping is given to each request. The results show that an algorithm with dynamic TTL approach gives better performance.

An interesting part of HTTP protocol is a redirection mechanism. HTTP redirection can be used to forward the requests to a set of servers which almost innoticable to the user. The main Web server can function as redirection server. All the client requests are directed to this main server and redirected to the servers in the cluster pool. The server selection is based on the server utilization together with some other parameters [10]. In [5], the proposed scheduling algorithms use the combination of DNS-based and HTTP redirection servers. The IP-Rewriting approach [11, 6] requires additional IP header modification at the router. The IP destination address of the HTTP request is modified to the specified Web server. The router needs to keep track on the relationship between clients and servers in the systems. Although this method has full control on the load sharing, the router can be a bottleneck and be considered as a single point of failure. In this paper, we consider DNS-based approach in distributing the Web requests.

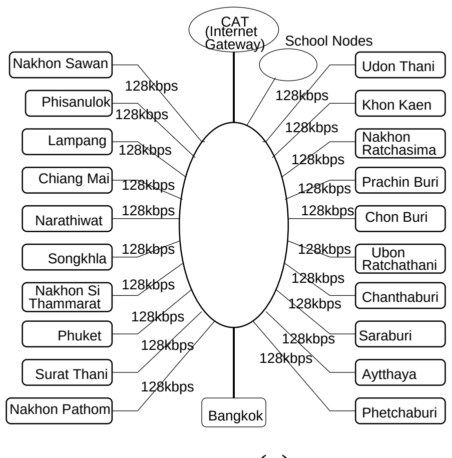
3. A Load Balancing Algorithm under Limited Bandwidth

Most organizations are connected to the Internet via leased lines. The monthly cost of the leased line is a great expense. Therefore the speed of the leased line is limited. Figure 3 (a) shows the network of 1998's Schoolnet which has evolved from the Golden Jubilee Access network[1]. A group of School in the provinces are connected using 128kbps links to the Internet. All Internet Traffic relies on these low speed links. If each school owns the Web server, not only does the Web server could be a system bottleneck but also the communication link. Some might argue that the network technology is improved in a rapid rate, however history has shown that the available bandwidth is never keep up with the user demand. As the communication bandwidth improves, the new applications demand even higher speed. In Thailand, many organizations have this class of Internet architecture, i.e., Schoolnet, GInet, UniNet.

The general model of limited bandwidth scenario of the Internet is presented in Figure 3 (b). The Internet backbone comprises of a set of high speed switches and fiber optic infrastructure. The edge routers connect the Internet Services Providers (ISP) to the backbone. The ISPs serve as a distribution point to customers. The customer of ISP can be either a part of ISP network system or an autonomous system (AS). The dashed-line represents the low speed link that is a bottleneck in many present Web services systems. From the ISP, the leased line connects the ISP to the organization's gateway. All client's traffic and all web traffic has to share this link. This amount of traffic is easily overload the link which will lead to useless state where most of the connections do not success (timeout).

From above reason, if organizations that belong to the same domain or loosely administrate by the same entity can cooperated on the Web services, the performance and also availability of the systems can be improved significantly. The participate organizations will initially sign up for the services. The software that responsible for replicating the web objects and managing the distributed system is installed to each server. The CNS(s) has responsibility in distributing the HTTP requests among these servers. The geographically distributed servers will help sustaining the quality of services when some major web events occur.

[1]Schoolnet's information was obtained from National Electronics and Computer Technology Center `http://www.nectec.or.th`.



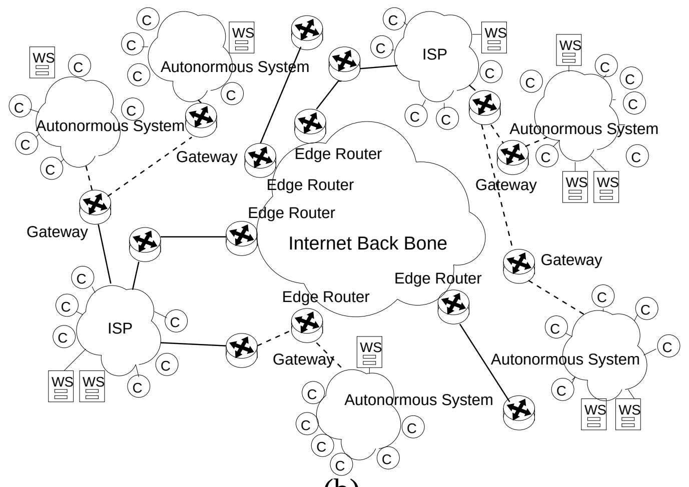


Figure 3. (a) Thailand Schoolnet architecture (b) Geographically distributed server under limited bandwidth: dash-lines represent communication links with limited bandwidth that connect ISP to autonomous systems (AS).

Under limited bandwidth environment, both server and communication link are considered as the performance bottleneck. Previous works on distributed web server did not take the communication link utilization in the server cost calculation. We propose a new server cost shown in the following equation.

$$ServerCost = (\alpha U_s + \beta U_l) \tag{1}$$

where U_s is the server utilization and U_l is the link utilization. α and β are the constant weight values. Figure 4 describes the load sharing algorithm. The CNS first probe the server utilization. Both push and poll paradigms can be implemented in the CNS and the Web servers to obtain utilization information. The utilization for each communication links is fetched using standard SNMP services. After all relevant information is gather, CNS calculates the cost of each server. The server with the lowest cost is selected and its IP address is returned to the requester. With slight modification, the new server cost can work together with other scheduling algorithms such as multi-tier round robin, hidden load weight, HTTP redirection [8, 6, 12]. α is intended to adjust the effects of server load and β is used to weight the network load. Figure 4 (b) specifies 6 cases of server cost. In case a ($\alpha = 1, \beta = 0$)), only server utilization is incorporated in the server cost. Similarly, only communication link utilization is considered in case b. Effects of server load and network load are enumerated in the case c, d, e. Case c gives equally weight to the server load and network load. Unbalance effects of the two loads are presented in case d and e.

In heterogenous environment, the performance of each Web server site is different in both processing power and communication bandwidth. The α and β values can be set accordingly. Server throughput and network throughput are two parameters that represent the system capacity. The server throughput T_s is equal to inverse of average request service time and the T_l is given as the average file size divided by the link bandwidth. To better predict the server cost, the server cost is set to $(T_1 U_s + T_s U_l)$ in the case t which is equivalent to $(U_s/T_s + U_l/T_l)$. U_x/T_x is essentially average a mean service time of the system x. Based on new servers cost, extensive experiments

```
Load Sharing Algorithm
Input:
  1. Utilization of Server $i$ for all $i = (0, \dots, N)$
  2. Utilization of Communication Link $i$ for all $i$
Output: IP address of the selected Web server.
Procedure:
  Begin
    1. Probe server utilization for all $i = (0, \dots, N)$.
    2. Probe link utilization for all $i = (0, \dots, N)$.
    3. Calculate the $servercost_i$ = $(\alpha U_{server_i} + \beta U_{link_i})$
       for all $i = (0, \dots, N)$.
    4. Select $i$ with the lowest cost
    5. Return the $IP_i$ address.
  End
```

(a)

Description	α	β
case a	1.0	0.0
case b	0.0	1.0
case c	0.5	0.5
case d	0.75	0.25
case e	0.25	0.75
case t	T_l	T_s

(b)

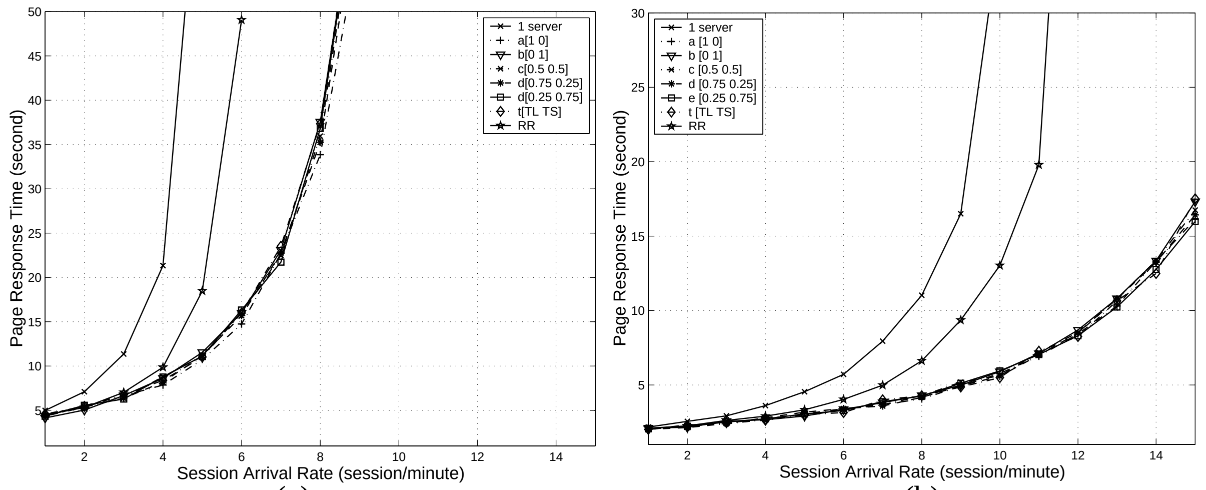
Figure 4. Load sharing algorithm for Web server system under limited bandwidth (a) Formal load sharing algorithm (b) six experiment cases of parameters setting for α and β.

were simulated to investigate the performance of the new algorithms. The simulation results are presented in the next section.

4. Performance Evaluation

A file-level Web simulation is developed to evaluate the performance of the distributed load sharing algorithms. A discrete event-driven simulator was built using C++ based on the CSIM simulation platform. The simulation model consists of ten domains. The 64 kbps and 128 kbps bandwidth are selected as the speed of limited bandwidth links. The delay of the Internet Backbone is assumed to be equal to a constant value of $3ms$. Hit service time is set to 22.5 ms. and URL translation time is set to 4.5 ms. The TTL value is selectable between 60 and 255 seconds. The number of clients in the system is equal to 1500. The distribution of the client from each domains is represented using Zipf's distribution [13]. The probability of selecting the domain rank i^{th} is equal to $\frac{1}{i^{(1-x)}}, (0 \geq x \geq 1)$ where a lower rank domain has more probability of generating requests. The traffic model used in the simulation is based on [14]. The request size, reply length, page length, consecutive document retrievals are formulated based on real experiments. Each page consists of 2.8 to 3.2 files and an average of 4 web pages are browsed in each session. Average user thinking time is equal to 15 seconds. For more information on the traffic model, the reader is encouraged to find detail information in [14].

A web page response time, measured in ms, is used as a performance metric since it is most perceptable by a user. The page response time is defined as the elapsed time between the start up of the first TCP connection and the time that the client terminates the last TCP connection. Several files are transfered and multiple TCP connections can be established during the Web page fetching. The session generation time was assumed to be exponentially distributed. The utiliza-



(a) (b)

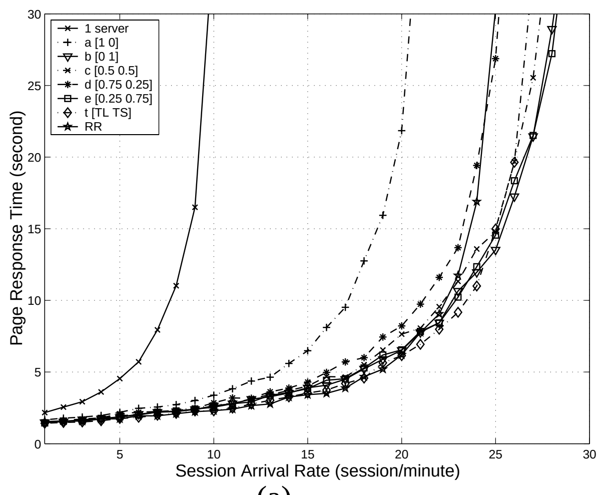
Figure 5. Session arrival time versus page response time of 2-server homogeneous distributed Web servers (a) 64 kbps limited bandwidth link (b) 128 kbps limited bandwidth link.

tion update time is programmable in the simulator. Statistical results of the first 100,000 sessions are not included in the result to reduce the transient effects of the network. The simulation results stay within a confident interval of 95%.

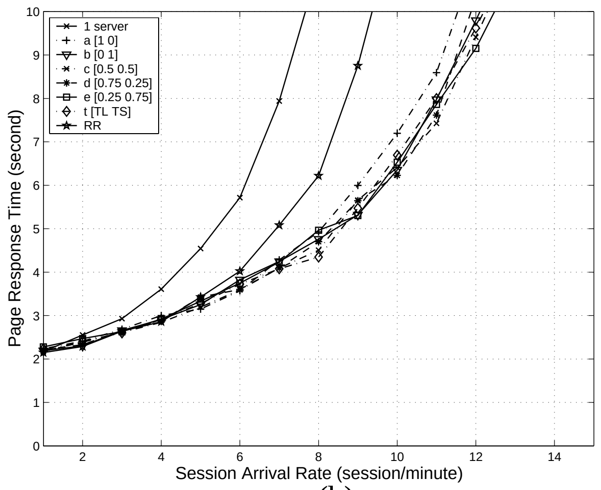
Figure 5 (a) shows the session arrival time versus page response time of the homogeneous systems with 64 kbps. TTL and utilization update time is set to 255 seconds and 8 seconds, respectively. The algorithms that aware of server load and network load perform 40% better than the round robin algorithm at the workload of 5 sessions per minute. The performance of cases (a-e) is comparable in this case. The results show that in the limited bandwidth situation, the utilization factors have strong effect on the performance. Similar trend is observed in Figure 5 (b) with 128 kbps. Because of homogeneity, the case t is equivalent to the case c. The simulation results of link heterogeneity is given in Figure 6 (a). The system consists of two servers. One server connects to a 128 kbps link and the other connects to 256 kbps links. In this case, the algorithm that considers only server utilization gives poor performance which is worst than the round robin approach. The round robin scheme give good performance at the low load but saturate early at the heavy session arrival. The page response time of the test cases that taken link utilization into account is grouping in the lowest value. The delay incurred in the limited bandwidth links strongly effects the performance of the systems.

To observe the effects of difference in server processing power, one server was programmed to be two time faster than the second server as shown in the Figure 6 (b). Both servers are connected to the same 128 kbps links. Case b to t perform better than the round robin algorithm and case $a[1, 0]$ that considers only server utilization. The case t that considers minimum response time for both server side and network side balance out the page response time.

From the simulation results, changing the server and network utilization from 8 seconds to 1 seconds does not have strong effects to the performance results. Reducing the TTL value form 255 seconds to 60 seconds improve the overall performance by 5.68-31.24% for the proposed schemes and 12.98-97.07% for the round robin algorithm. In the Web applications, doubling the bandwidth from 64kbps to 128kbps results in reducing the page response time by a factor



(a)



(b)

Figure 6. Session arrival time versus page response time of 2-server distributed Web servers. (a) heterogenous links of 128 kbps and 256 kbps bandwidth (b) heterogenous distributed Web servers with 128 kbps limited bandwidth links. The first server is two times faster than the other.

of 2 at the workload five sessions per minute. The proposed approaches give good performance compared to the round robin and server utilization especially under heterogenous environment. The network load factor in the efficient load balancing algorithm is worth consideration. The case t will help reducing effects for exterior sources such as heterogenous links and other Internet traffic.

5. Concluding Remarks and Future Works

The communication bandwidth for accessing information on the Internet is expensive. Many organizations can only afford a low speed link. The quality of services in term of page response time might not be acceptable to the user. Cooperative Web services could help improve the performance of the WWW systems. These organizations can establish the geographically distributed Web server systems in which the web contents are replicated to all sites. Previously proposed load sharing algorithms are well fit under limited bandwidth links. A load sharing algorithm that aware of communication link utilization is proposed. The DNS-based algorithm using the new server cost is described. Our proposed approach focuses on the effect of network delay in the distributed WWW systems. The simulation results shows that the proposed approach performs significantly better than the round robin scheme and the algorithm that consider only the server utilization. The proposed approach is also well adapted in the heterogeneity environments. The simulation results of processing power heterogeneity and communication heterogeneity are shown. We are working on the large scale simulation where the number of servers and degree of heterogeneity increase.

References

[1] T. Berners-Lee, R. Calliau, A. Luotonen, H. Frystyk, and A. Secret, “The World-Wide Web,” *Communication of ACM*, vol. 37, no. 8, pp. 76–82, 1994.

[2] T. T. Kwan, R. E. McGrath, and D. A. Reed, “NCSA’s World Wide Web server: Design and performance,” *Computer*, vol. 28, pp. 68–74, November 1995.

[3] C. Wu, C. Yeh, and J. Juang, “A World-Wide Web server on a multicomputer system,” in *the Proceeding of the 1996 International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN’96)*, pp. 522–528, 1996.

[4] M. Colajannni, P. S. Yu, and V. Cardelliini, “Dynamic load balancing in geographically distributed heterogeneous Web servers,” in *the Proceeding of the 18th International Conference on Distributed Computing Systems*, (Amsterdam), May 1998.

[5] D. Andresen, T. Yang, V. Hoedahl, and O. H. Ibarra, “SWEB: Toward a scalable World Wide Web-server on multicomputer,” in *the Proceeding of the 10th IEEE International Symposium on Parallel Processing*, (Los Alamitos, Calif.), pp. 850–856, IEEE Computer Soc. Press, 1996.

[6] V. Cardellni, M. Colajanni, and P. S. Yu, “Dynamic load balancing on Web-server systems,” *IEEE Internet Computing*, pp. 28–37, May-June 1999.

[7] M. F. Arli and C. L. Williamson, “Web server workload characterization: The search for invariants,” in *the Proceeding of ACM Sigmetrics ’96*, (Philadelphia), pp. 126–137, May 1996.

[8] M. Colajanni, P. S. Yu, and D. M. Dias, “Analysis of task assignment policies in scalable distributed Web-sever systems,” *IEEE Transaction on Parallel and Distributed System*, vol. 9, pp. 585–600, June 1998.

[9] M. Colajannni, D. M. Dias, and P. S. Yu, “Scheduling algorithms for distributed Web servers,” in *the Proceeding of the 17th International Conference on Distributed Computing Systems (ICDCS’97)*, pp. 169–176, IEEE, 1997.

[10] V. Cardellini, M. Colajanni, and P. S. Yu, “Redirection algorithms for load sharing in distributed Web-server systems,” in *the Proceeding of the 19th IEEE International Conference on Distributed Computing Systems*, May 1999.

[11] G. Hunt and et al., “Network dispatcher: A connection router for scalable internet services,” *J. Computer Networks and ISDN Systems*, vol. 30, 1998.

[12] V. Cardellini, M. Colajanni, and P. S. Yu, “DNS dispatching algorithms with state estimators for scalable Web-server clusters,” *World Wide Web J.*, vol. 2, July 1999.

[13] K. Y. Leung and K. H. Yeung, “The design and implementation of a WWW traffic generator,” in *the Proceeding of the 7th Intl. Conf. on Paral. and Dist. Systems*, 2000.

[14] B. A. Mah, “An empirical model of HTTP network traffic,” in *the Proceeding of INFOCOM’97*, (Kobe, Japan), April 1997.