



Dynamic traffic controls for Web-server networks

Liming Liu ^{a,*}, Yumao Lu ^b

^a *Department of Industrial Engineering and Engineering Management, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong*

^b *Department of Electrical Engineering, University of California at Los Angeles, Los Angeles, CA 90024, USA*

Received 23 May 2002; received in revised form 17 June 2003; accepted 5 November 2003

Available online 20 April 2004

Responsible Editor: G. Pacifici

Abstract

Distributed Web-server systems have been widely used to provide effective Internet services. The management of these systems requires dynamic controls of the Web traffic. With the development of multimedia Web sites and increasingly diversified services, the existing load balancing approaches can no longer satisfy the requirements of either the service providers or the users. In this paper, a new reward-based control mechanism is proposed that can satisfy the dynamic content-based control requirement while avoiding congestion at the dispatcher. The core of the control algorithm is based on an MDP model. To minimize the system overhead, a centralized dispatching with decentralized admission (CDDA) approach is used to distribute the control related computation to each server pool. This cuts down the dimensions of the problem dramatically. We also propose a state-block scheme to further reduce the state space so that the algorithm becomes computationally feasible for on-line implementation. Simulation results demonstrate that the proposed state-block approach can not only reduce the computation time dramatically but also provide a good approximation of power-tailed request interarrival times common for Internet traffic. Finally, an implementation plan with system design is also proposed.

© 2004 Elsevier B.V. All rights reserved.

Keywords: Web server; Reward-based admission; QoS; Customer utility; MDP; Routing control

1. Introduction

The enormous volume and diversity of Internet usage brings new problems and challenges to Internet service providers in the areas of languages, time zones, surfing habits, long latency, and service

requirement differences. Web-server system performance has become increasingly more important to the success of many Internet related companies.

To handle the ever increasing volume and diversity of Web traffic, the distributed Web-server system has become a commonly accepted solution platform, which not only provides higher service capability but also better service locality to Internet users around the world. Of various technical and control issues for distributed systems, the admission and routing control of service requests are

* Corresponding author.

E-mail addresses: liulim@ust.hk (L. Liu), luym@ee.ucla.edu (Y. Lu).

particularly important to the performance of the Web-server system and the quality of service. There exist a number of network routing control mechanisms, including client-based control, DNS-based control, server-based control and dispatcher-based control [6]. In the client-based control, the Web client or client-side proxy server selects the Web server for a request. The DNS-based control authorizes DNS server to route a request to a Web cluster in the permitted time interval and imposes fixed logic-name-to-IP mapping outside that time interval. In the server-based control, each server decides whether to serve a request, direct it to another server, or reject the request. The dispatcher-based control uses a dispatcher machine to route a request to a specific server. The client-based control has limited applicability since it requires a browser to understand the Web-server network structure [16]. The server-based control (e.g. [1,12]) is a distributed control and therefore it cannot generate a global optimal policy. Extensive research has been carried out in searching for effective load balancing control in the DNS system, (e.g. [5,6,9,13]). However, while the DNS-based control is centralized, the controller does not really have full control of the system due to the limitation imposed by the Time-to-Live (TTL) value which is used to decide how long the binding between the server and the client is to remain. Since most servers are configured such that they do not accept a low TTL value, a high TTL value will render most of the requests out of control [9]. The dispatcher-based control is a unique mechanism in which the controller has the full control over the system and, therefore, load balancing can be achieved easily. With full control of the system, more complicated control algorithms can be implemented in the controller so as to satisfy other requirements beyond load balancing. The main drawback of the dispatcher-based control is its use of package forwarding technique. With this mechanism, the client's packages and/or the server's packages have to pass through the centralized dispatcher. This creates a high workload and often causes congestions at the dispatcher, turning it into the bottleneck of the network system [11].

The shortcomings of the existing control mechanisms and new challenges arising from the intro-

duction of new functions on the Web call for the development of more effective new control mechanisms. The emergence of multimedia contents, especially video contents, on the Web presents a significant challenge as well as opportunities to Internet service providers. Good quality of service (QoS) in multimedia such as video-on-demand (VOD) means high resolution, large screen size and low latency. These generate much heavier data flows and storage consumptions, and hence are more costly to operate than other types of services. For a system that provides multiple services, it is necessary to classify and prioritize services so that various QoSs are guaranteed with appropriate compensations, e.g., higher QoS and/or more demanding services requires higher service charges. As such, reward-based controls must be used. Various admission policies have been proposed to maximize the system's rewards (e.g. [7,8,14]). However, simple admission controls cannot provide solutions to all the problems related to file transfers and data movements, such as latency time caused by network and load balancing across distributed servers. Therefore we need a reward-based routing control mechanism, which is not only able to reject some unprofitable requests but also provide good locality (reducing the latency caused by the network).

This research is thus motivated by the challenge and need for a reward-based control mechanism that can also guarantee the appropriate quality of service to different users. In Section 2, we present the CDDA control mechanism, e.g., the centralized dispatching with decentralized admission, and the control algorithm. To implement the control on-line and achieve dynamic control, a faster algorithm is needed. In Section 3, we propose a scheme that can reduce the optimization state space and dramatically speed up the computation. We call the scheme the state-block scheme, or SBS. With SBS, the CDDA mechanism can be implemented for on-line control. The performance of the CDDA is compared with the standard centralized control in Section 4 through numerical studies and simulation. The results show that while the reward with CDDA mechanism is marginally lower than that with the centralized control (which is considered optimal in reward), the quality of

service with CDDA is dramatically better. We also compare performance of CDDA with and without the SBS. The results indicate that with the SBS, the quality of service is improved in most cases. In particular, when the request arrival stream used in the simulation models the real Internet traffic better, the quality of service can be significantly improved without significantly sacrificing the system wide reward. Furthermore, we define a customer utility measure and compute it in the simulation study to examine how customers fare with the proposed control mechanism. The paper concludes in Section 5 with a summary of the results and a brief discussion of further research opportunities.

2. Centralized dispatching with decentralized admission

2.1. Control mechanism

The advantage of a dispatcher-based control is its ability to fully control the system. In a standard dispatcher, an active executor program monitors the number of active connections of the TCP. This executor is able to control a Web-server network using a load-balancing algorithm. In order to achieve a higher system-wide reward and a better QoS, we propose a centralized dispatching with

decentralized admission (CDDA) control mechanism as shown in Fig. 1.

A Web-server network consists of a number of server pools that are located around the world and connected through the Internet. Each server pool (SP) consists of a number of Web servers. A dispatcher-like server is used as a service center (SC) which is located at the same place as one of the SPs. The entire customer population is divided into regions around the SPs. File requests originating from all regions are first directed to the SC. Upon the arrival of a request, the central controller (i.e., the SC) will either select an SP and redirect this request to this SP or reject the request if no SP is selected. The decisions made at the SC are based on a dispatching algorithm, a control policy table and the type of request which is determined by the SC. While the control policy table is maintained at the SC, different information contained in the table is updated in different ways and mostly by the SPs. The SP state information in the policy table is updated automatically by the SPs whenever there is a change of the state. The admission control policies are determined by the SPs individually and are updated periodically. The admission control policy of an SP is a vector with a dimension equal to the number of possible request classes. Each element of the vector is an acceptance–rejection threshold. An optimal

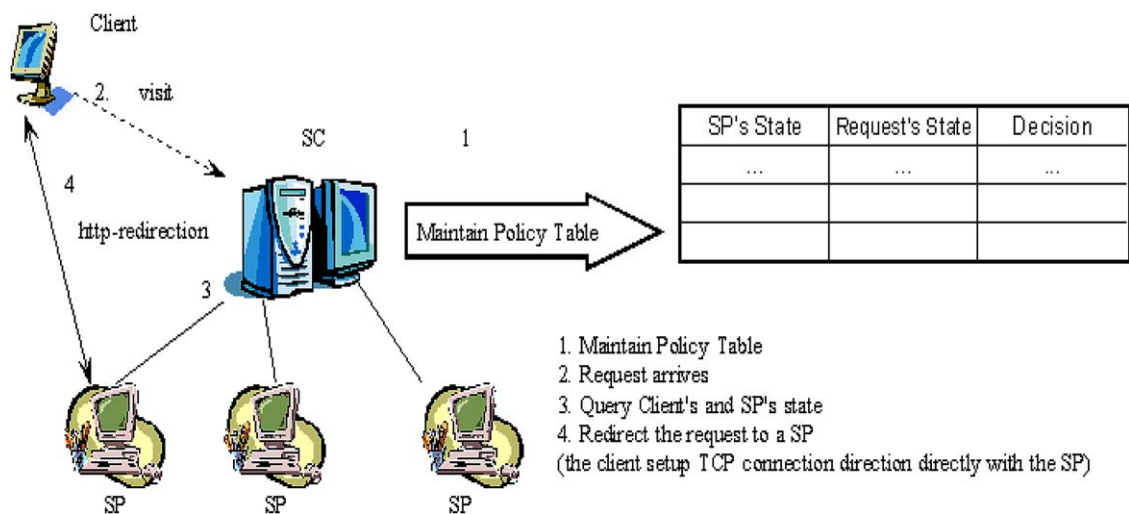


Fig. 1. The CDDA control mechanism.

admission control algorithm is used at the SP to determine the control policy. Details on the control algorithms and the policy table are provided in the next section.

This approach integrates the dispatcher-based approach and the HTTP-redirection techniques. With HTTP-redirection, when the redirection command is executed, the SC sends the assigned SP's IP address back to the browser and from where a new request is automatically made to set up a connection with this SP directly. The difference between this approach and the existing dispatcher-based approach is that the SC does not forward any package. Instead, it only maintains the control policy table and makes admission and routing decisions. By doing this, the controller gains full control of the system without adding excessive overhead so that it will not be congested and become the network bottleneck itself. Technically, it is possible that some users can discover the best server cluster and access it directly. In implementing this control mechanism, we can find a way to either accept or reject such "smart requests" at the SP according to the local rule. Or we can use a password or other security techniques to restrict such independent access.

2.2. Control algorithms

Consider a Web-server network consisting of one Web service center (SC) and J distributed SPs, indexed by $j = 1, 2, \dots, J$. The SPs are connected to the SC through the Internet. The global network is divided into J regions according to the locations of the J SPs. Requests (hits on the Web) are initiated randomly from different regions to the SC. The transmission and propagation delay of a request from region j to SP k is denoted by c_{jk} . Requests are classified into I classes according to the processing times (sizes of the files requested). Let the processing rate of class i requests be μ_i . For convenience, we order the classes in decreasing order, i.e., $\mu_i < \mu_h$ for $i > h$. Within some fixed time interval, the request traffics for various classes can be assumed stable, for example, class i requests is assumed to follow a Poisson process with rate λ_i . This time interval can vary from several minutes to more than 1 h depending on the particular system

[4], but it must be small enough so that the traffic is indeed stable and, at the same time, large enough for off-line policy computation. If a request is assigned to an SP, it joins the queue of the SP when all the servers are occupied. SP j has a maximum connection M_j enforced by a finite buffer. Let k_{ij} be the number of class i requests in SP j and $\mathbf{K}_j = [k_{1j} k_{2j} \dots k_{Ij}]^T$ be a column vector. Clearly, k_{ij} satisfies $\sum_{i=1}^I k_{ij} \leq M_j$, and \mathbf{K}_j provides the full information of the state of SP j . The matrix $\mathbf{K} = [\mathbf{K}_1 \mathbf{K}_2 \dots \mathbf{K}_J]$ defines the state of the entire network system.

We now summarize the notation below:

λ_i	the rate of class i requests
μ_i	the processing rate of class i requests
r_i	the reward from fulfilling a class i request
N_j	the capability of SP j
c_{jk}	the transmission and propagation delay of a request from region j to SP k
M_j	the maximum connections at SP j
I	the number of different request classes
J	the number of SPs
\mathbf{K}	the system state matrix
\mathbf{K}_j	the state vector of SP j
$\beta_{ij}(\mathbf{K}_j)$	the state dependent departure rate of class i requests from SP j
$W_j(\mathbf{K}_j)$	the state dependent expected queueing delay in SP j
$a_j(K)$	the action, i.e., assigning an SP ID number to a request.

We first describe the centralized control algorithm here for comparison with our CDDA algorithm. With a reward-based control, the main control objective is to maximize the rewards received from services provided. Since rewards are different for different contents, it is necessary to identify the departure rates of different classes of requests. Recall that SPs are managed by some local routing algorithms independent of the server network control, like a black box to the Web-server network. Thus we may treat each SP as a single server station when estimating the departure rate. Therefore, we have

$$\beta_{ij}(\mathbf{K}_j) = \frac{k_{ij} N_j}{\sum_{i=1}^I k_{ij}} \mu_i,$$

where $k_{ij}/(\sum_{i=1}^I k_{ij})$ denotes the probability that a class i request is being served in SP j . Clearly, the total event rate is

$$S(K) = \sum_{i=1}^I \lambda_i + \sum_{i=1}^I \sum_{j=1}^J \beta_{ij}(K).$$

By definition, the system state satisfies the Markovian property and we can construct a Markov decision process (MDP) model for the centralized control algorithm. We can approximate the queueing process of each SP as an $M/M/1/M_j$ process. In any state, the interarrival times and the inter-departure times are exponential, and hence the time between any two events (arrival or departure) is also exponential. Thus, the probability that the next event is a class i arrival is $\lambda_i/S(K)$ and the probability that the next event is a class i departure from server j is $\beta_{ij}/S(K)$. Let $V(K)$ be the value function of the optimal system reward strategy for a given system state K right after an event, i.e., $V(K) = \max\{E[V^{t+1}|V^t(K)]\}$. Then, we have

$$\begin{aligned} V(K) = & \sum_{i=1}^I \frac{\lambda_i}{S(K)} \max \left\{ V(K), \max_j \left\{ V(K + e_{ij}) \right. \right. \\ & \left. \left. + r_i \left| \sum_{i=1}^I k_{ij} < M_j \right. \right\} \right\} \\ & + \sum_{i=1}^I \frac{\beta_{ij}}{S(K)} V(K - e_{ij}) 1[k_{ij} > 0], \end{aligned} \quad (1)$$

where e_{ij} is a matrix of 0s except a 1 at row i and column j . We note that this value function does not require the FCFS assumption for the queueing system.

We may note from (1) that the SC may reject an unprofitable (low reward or long service time) request before all the buffers are full.

There are two problems with the MDP model based on Eq. (1): (1) the QoS is not considered, and (2) the state space is extremely large since the maximum number of total connections can be very large, making the models computationally infeasible. We have to reduce the state space and combine the objectives of low response time and high system reward. The centralized dispatching

with decentralized admission (CDDA) strategy is thus designed to tackle these problems by first optimizing individual SP's rewards using an MDP model and then improving the quality of service using a greedy algorithm. This approach works in the following way.

At the arrival of a request to the SC, each and every SP determines independently whether this request is admissible according to its present state and the reward maximization principle. The SC will then choose an SP from those SPs that are positive to the admission of the request so that the expected average response time is minimized. If an SP is eventually chosen, the request will join the queue when the SP is busy. If no SP will admit this request, the request will be rejected by the SC. The number of class i requests for all $i = 1, \dots, I$ in an SP provides the full information of present state of the SP.

Let λ_{ij} denote the arrival rate of class i to SP j in the current period. An estimate of λ_{ij} will be given in Section 4. We further specify the following elements of the model, for SP j , $1 \leq j \leq J$:

State. $\{\mathbf{K}_j\}$; $\mathbf{K}_j = (k_1 k_2 \dots k_I)^T$ where $k_i \geq 0$ for $1 \leq i \leq I$ and $\sum_{i=1}^I k_i \leq M_j$.

Decision epoch. Right after an event (arrival or departure), i.e., $T = \{1, 2, 3, \dots, \infty\}$.

Actions. $\{\mathbf{A}_j\}$; $\mathbf{A}_j = \{a_1, a_2, \dots, a_I\}$ where $a_i = 0$ or 1 denotes the rejection or acceptance of a class i request.

Transition probabilities

$$\begin{aligned} p[\mathbf{L}_j | \mathbf{K}_j, \mathbf{A}_j] &= \begin{cases} \frac{\lambda_{ij}}{S(\mathbf{K}_j)}, & \mathbf{L}_j = \mathbf{K}_j + \mathbf{e}_i, \\ & \mathbf{A}_j = \{a_1, \dots, a_{i-1}, 1, a_{i+1}, \dots, a_I\} \\ & \text{or } \mathbf{L}_j = \mathbf{K}_j, \\ & \mathbf{A}_j = \{a_1, \dots, a_{i-1}, 0, a_{i+1}, \dots, a_I\} \\ & \text{or } \mathbf{A}_j = \{a_1, \dots, a_{i-1}, 1, a_{i+1}, \dots, a_I\}, \\ & \sum_{i=1}^I k_i = M_j \\ \frac{\beta_{ij}(\mathbf{K}_j)}{S(\mathbf{K}_j)}, & \mathbf{L}_j = \mathbf{K}_j - \mathbf{e}_i, \\ & \mathbf{A}_j = \{0, 0, \dots, 0\}, \\ 0, & \text{otherwise,} \end{cases} \end{aligned} \quad (2)$$

where

$$\beta_i(\mathbf{K}_j) = \frac{k_{ij}N_j}{\sum_{i=1}^I k_i} \mu_i,$$

$$S(\mathbf{K}_j) = \sum_{i=1}^I \lambda_{ij} + \sum_{i=1}^I \beta_i(\mathbf{K}_j).$$

Rewards. Rewards can only be earned when the arrival (such as class i) is accepted, i.e. when $a_i = 1$, the reward is

$$r[\mathbf{K}_j, \mathbf{A}_j] = \sum_{i=1}^I p[\mathbf{K}_j + \mathbf{e}_i | \mathbf{K}_j, \mathbf{A}_j].$$

The value function can then be stated as

$$V(\mathbf{K}_j) = \max_{\mathbf{A}_j} \left\{ r[\mathbf{K}_j, \mathbf{A}_j] + \sum_{L_j} p[L_j | \mathbf{K}_j, \mathbf{A}_j] V(L_j) \right\}.$$

An iterative algorithm can then be constructed to find the optimal solution.

In Section 4, the average response times (ARTs) and system rewards earned with the policies computed with Eq. (1) and the proposed CDDA approach are compared. The numerical results show that the CDDA approach leads to a significantly lower ART with only a slightly lower reward than the centralized control model. The advantage of our approach is obvious.

3. The state-block scheme and control algorithm

Although computational efforts have been distributed to each SP in the CDDA approach, the computation time is still too long for on-line implementation because the state space at each SP can still be too large. To further reduce the state space, we use the *level* instead of the *number* of class i requests in a SP as the state in the MDP model. A level is an index of blocks. Specifically, when the level of class i requests in the SP is l , then there are $l \times B$ to $(l+1) \times B - 1$ class i requests in the SP, where B denotes the block size. Thus a level is a range instead of a fix number of requests in the system. We call this heuristics the state-block scheme, or SBS. Fig. 2 shows a block diagram example.

The SBS is motivated by the following facts: the state space without using SBS is too large; rejected

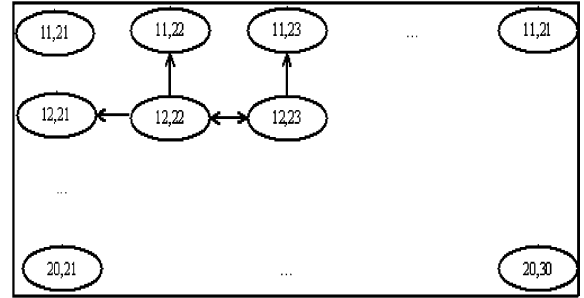


Fig. 2. State block ($B = 10$).

customer requests may return immediately creating lumpier traffic; and, in the Internet traffic environment, the state transitions in an SP are more likely to jump from one block to another. The first fact is obvious. Regarding the second and third facts, we note that Internet traffic has four main characteristics: self-similar traffic traces, long-rang dependence, bursty on multiple scales, and long or heavy-tailed packet inter-arrival time [10]. Returned requests cause lumpiness in the request traffic. Hence, it is reasonable to suggest that if the number of a class of requests in an SP is to change, it is more likely to change continuously on one direction in a very short interval, e.g., the number increases by 10 before a decrease. This scenario can be interpreted in another way: when the variance increases, it is easier for the state to transit to the next block in a short interval than to the next state. At the same time, when we block the states and let the edges of the block be absorbing states, the probability that the state transit to the edge also increases. So, the state-block method captures this Internet state transition behavior nicely. Our numerical results support this observation.

We now present this approach through a problem with two request classes. We note that each block is isolated and the states on the edge of the block are absorbing states, as shown in Fig. 2. Again, considering only SP j , the state transition probability for the state on the edges of the block then satisfies

$$p[L_j | \mathbf{K}_j, \mathbf{A}_j] = \begin{cases} 1, & \text{if } L_j = \mathbf{K}_j, \\ 0, & \text{otherwise,} \end{cases} \quad (3)$$

if $k_1 = l \times B$ or $(l+1) \times B - 1$ and $k_2 = m \times B$ or $(l+1) \times B - 1$, where k_1 and k_2 denote the num-

bers of class 1 and class 2 arrivals, respectively in the SP and m and n are integers.

The block transition probabilities are computed by assuming that the system is stationary within each block. Let p^B denote the block transition probabilities. We have

$$p^B[\mathbf{L}_j | \mathbf{K}_j, \mathbf{A}_j] = \begin{cases} \frac{\sum_{j=l_2 \times B}^{(l_2+1) \times B-1} \pi_{(l_1+1) \times B-1, j}}{\sum_{i=l_1 \times B}^{(l_1+1) \times B-1} \sum_{j=l_2 \times B}^{(l_2+1) \times B-1} \pi_{ij}}, & \mathbf{L}_j = \mathbf{K}_j + \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \mathbf{A}_j = \begin{bmatrix} 1 \\ a_2 \end{bmatrix} \\ \frac{\sum_{i=l_1 \times B}^{(l_1+1) \times B-1} \pi_{i, (l_2+1) \times B-1}}{\sum_{i=l_1 \times B}^{(l_1+1) \times B-1} \sum_{j=l_2 \times B}^{(l_2+1) \times B-1} \pi_{ij}}, & \mathbf{L}_j = \mathbf{K}_j + \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \mathbf{A}_j = \begin{bmatrix} a_1 \\ 1 \end{bmatrix} \\ \frac{\sum_{j=l_2 \times B}^{(l_2+1) \times B-1} \pi_{l_1 \times B, j}}{\sum_{i=l_1 \times B}^{(l_1+1) \times B-1} \sum_{j=l_2 \times B}^{(l_2+1) \times B-1} \pi_{ij}}, & \mathbf{L}_j = \mathbf{K}_j - \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \mathbf{A}_j = \begin{bmatrix} 0 \\ a_2 \end{bmatrix} \\ \frac{\sum_{i=l_1 \times B}^{(l_1+1) \times B-1} \pi_{i, l_2 \times B}}{\sum_{i=l_1 \times B}^{(l_1+1) \times B-1} \sum_{j=l_2 \times B}^{(l_2+1) \times B-1} \pi_{ij}}, & \mathbf{L}_j = \mathbf{K}_j - \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \mathbf{A}_j = \begin{bmatrix} a_1 \\ 0 \end{bmatrix} \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

The marginal distribution in each block is computed by using

$$\pi = \pi P, \quad (5)$$

where P is state transition probability matrix of each block modified by (3). With the transition probabilities, the MDP model becomes:

State. $\mathbf{L}_j = [l_1, l_2]^T$ where l_i denotes the level of class i requests in the system.

Decision epoch. Each time, the system changes from one block to another: $T = \{1, 2, \dots, \infty\}$.

Actions. $\mathbf{A}_j = \{a_1, a_2\}$, with $a_i = 0$ or 1 for rejection or acceptance of a class i request.

Rewards. Obtained using the departure rate

$$r[\mathbf{K}_j, \mathbf{A}_j] = \sum_{i=1}^I \beta_i(\mathbf{K}_j) r_i. \quad (6)$$

The solution is obtained from the procedure outlined in the following algorithm. For every fixed time interval, each SP performs a few computation steps as follows:

- Step 1. Update the arrival rates of different classes of requests;
- Step 2. Compute the transition probabilities by (2);
- Step 3. Compute the marginal distribution in each block by (5);
- Step 4. Compute the block transition probabilities by (4);
- Step 5. Use value iteration to calculate the policy [17];
- Step 6. Report the admission policy (see Fig. 3–6) to the SC.

After every event, each SP reports its state to the SC.

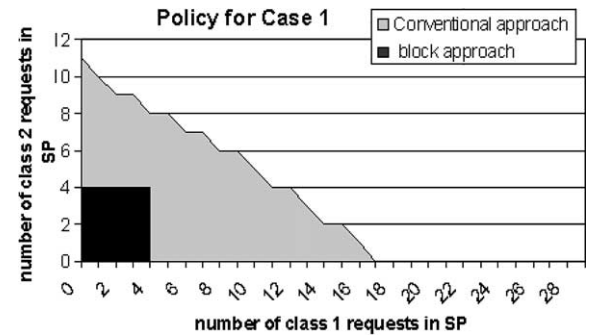


Fig. 3. Policy for Case 1.

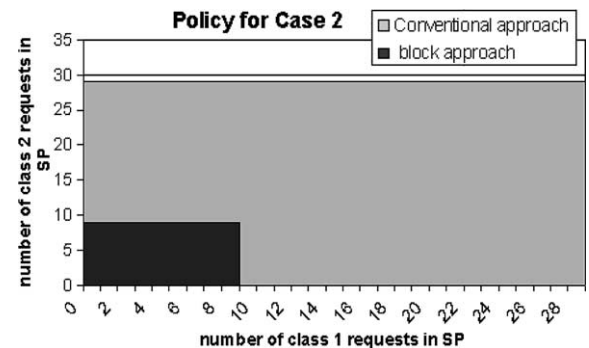


Fig. 4. Policy for Case 2.

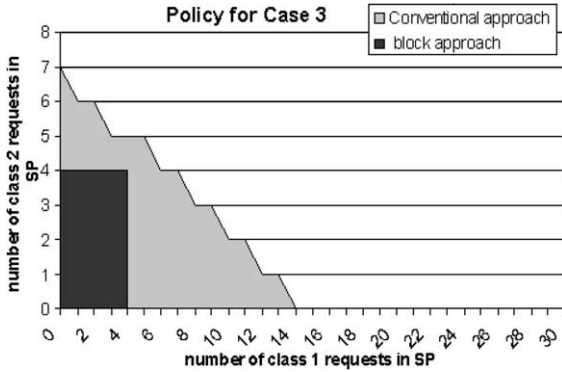


Fig. 5. Policy for Case 3.

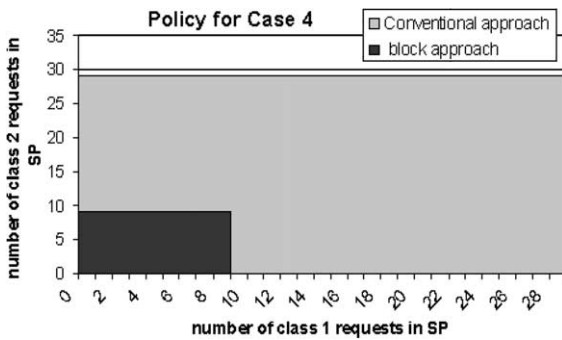


Fig. 6. Policy for Case 4.

Upon the arrival of a request, the SC searches, with a greedy algorithm, from the set of admissible SPs to choose one SP to receive the request so as to minimize the expected response time. An admissible SP is one that can accept a new request according its admission policy table and current state. Suppose, for example, both SP1 and SP2 can accept a class 2 arrival originating from region 3, the SC will assign the request to SP 1 if $c_{13} < c_{23}$, and vice versa.

4. Simulation, numerical studies and discussion

4.1. Comparison of CDDA with the centralized control

By the dynamic programming convention and noting that the arrival rate is updated periodically, an exponential weight may be applied to each

period to formulate an infinite-horizon dynamic programming problem. This weight discounts the rewards received in the future periods. Given the arrival rates and processing rates, the weight can be estimated as follows: suppose the value earned in a period is 10% less than the value earned from the first event of the previous period, the exponential weight α satisfies

$$\alpha^x \leq \frac{\alpha}{10},$$

in which x is the estimated the number of events in one period. Hence,

$$\alpha = 0.1^{1/(x-1)}.$$

In this section, we approximate λ_{ij} with the following formula:

$$\lambda_{ij} = \lambda_i \frac{N_j}{\sum_j N_j}. \tag{7}$$

The arrival process is split into j sub-processes virtually to each SP according to its capability. Eq. (7) gives the arrival rate of SP j which is referred to as the virtual arrival rate. It is assumed that the sub-process to each SP is still Poisson and thus the MDP formulation for each SP is reasonable. Our simulation results support this claim.

The comparison of the standard centralized control and the proposed CDDA is made in the following fashion: for the same system definition, the two approaches are used to generate the control policies, respectively; the policies are then used in simulation of an identical system; finally, the performance measures of each approach are compared.

The performance measures include service provider’s reward and customer’s utility. The service provider’s reward is defined according to its real business profit. For example, some class of multimedia file (usually with higher quality and therefore larger size) may need membership to access which requires an extra fee. In this case, we can define the reward of fulfilling a more profitable request larger than that of fulfilling a less profitable one. On the other hand, if all customers pay the same price, the provider may define the equal reward for all classes. In this case, the small file service seems more profitable than a larger file service.

For different service providers or different Web-server systems, the definition is different. In our simulation, the reward is defined according to the latter case, i.e., a unit reward will be earned when a request is fulfilled. Therefore, the total rewards are the total number of served requests.

In general, the customer utility is a concave function of the customer arrival rate in such a service system. Here, we do not need a precise definition, as we only need to compare, from the customer's perspective, the relative performance of the system under different control policies. Since the customer utility is negatively affected by the waiting time and the rejection rate, the following utility function is suffice for our purpose

$$u = 1/[aC(t) + (1 - a)p], \quad (8)$$

where t is the average response time, C is an increasing bounded function, p is the rejection cost to the customer, and a is the probability that a request is accepted. The total customer utility is therefore decreasing with the average response time (ART) and the number of rejected requested, where the request's ART is given by

$$\begin{aligned} \text{ART} = & \text{Processing time} + \text{Queueing delay} \\ & + \text{Transmission time} + \text{Propagation time.} \end{aligned}$$

We built a discrete-event simulator that models the traditional centralized control and our CDDA mechanism. The network component setting in our simulator is guided by empirical measurements of a similar Web-file service system [2,3]. We assume that the two classes of files are 1 Mbytes and 0.2 Kbytes. The file of 1 Mbytes stands for high resolution file and the file of 0.2 Mbytes stands for normal resolution file. When effective connection speed is 1.6 Mbytes per second, which is realistic through common LAN connection, the service time (processing time plus transmission time) therefore is 0.2 and 0.5 s, respectively. The corresponding SP service rates are $\mu_1 = 5$ and $\mu_2 = 2$, respectively. Propagation delay is transmission media dependent and can vary from several milliseconds to several seconds or even longer [3]. If the SP is local, the propagation time is usually several milliseconds and is therefore omitted in our simulations. If the SP is close, say within US, the

propagation time is set to a typical value of 0.2 s [18]. If the SP is very far away, the propagation delay is set to 2 s.

In the standard centralized control, the SC makes dispatching decisions based on the policy computed by (1). There are nine possible actions: reject all arrivals, direct class 1 arrivals to SP1 but reject class 2 arrivals, direct class 1 arrivals to SP2 but reject class 2 arrivals, direct class 2 arrivals to SP1 but reject class 1 arrivals, direct class 2 arrivals to SP2 but reject class 1 arrivals, direct all arrivals to SP1, direct all arrivals to SP2, direct class 1 arrivals to SP1 and class 2 arrivals to SP 2, and direct class 1 arrivals to SP2 and class 2 arrivals to SP1.

In the CDDA control, each SP makes its own admission decision and the SC makes dispatching decisions based on the decision tables from the SPs and the propagation delays. There are four possible actions for each SP: reject all arrivals; accept class 1 arrivals but reject class 2 arrivals; accept class 2 arrivals but reject class 1 arrivals; and accept all arrivals.

The maximum connection for SP1 is set to 4 so as to limit the simulation burden for the standard centralized control. A total of 32 cases of different system configurations are considered. In Cases 1–16, the propagation delays are quite low, at $c_{12} = c_{21} = 0.2$ s. This accounts for less than 20% of the total ART in the results shown in Table 1. In Cases 17–32, the propagation delays are relatively high, at $c_{12} = c_{21} = 2$ s. This network delay is significant and accounts for more than 50% of the total ART as shown in Table 1. In Cases 1–8 and 17–24, the two SPs have the same capability. In Cases 9–16 and 25–32, SP2's capability is two times of SP1's, that is $N_2 = 2$ and $M_2 = 8$. The arrival rates are given below:

- Cases 1, 9, 17, 25: $\lambda_{11} = \lambda_{12} = \lambda_{21} = \lambda_{22} = 1.0$.
- Cases 2, 10, 18, 26: $\lambda_{11} = \lambda_{22} = 2.0$, $\lambda_{12} = \lambda_{21} = 2.0$.
- Cases 3, 11, 19, 27: $\lambda_{11} = \lambda_{12} = \lambda_{21} = \lambda_{22} = 3.0$.
- Cases 4, 12, 20, 28: $\lambda_{11} = \lambda_{21} = 5.0$, $\lambda_{12} = \lambda_{22} = 2.5$.
- Cases 5, 13, 21, 29: $\lambda_{11} = 0.5$, $\lambda_{21} = 1.5$, $\lambda_{22} = \lambda_{12} = 1.0$.
- Cases 6, 14, 22, 30: $\lambda_{11} = \lambda_{21} = 2.0$, $\lambda_{12} = 0.5$, $\lambda_{22} = 1.5$.

Table 1
Comparison of centralized control and CDDA

SP	Result	$c_{12} = c_{21} = 0.2$								
		Balanced origins				Unbalanced origins				
		Light load		Heavy load		Light load		Heavy load		
		Case1	Case 2	Case 3	Case 4	Case 5	Case 6	Case7	Case 8	
BS	M1	ART	1.263	1.112	1.138	0.905	1.269	1.123	1.023	0.886
		R^*	23,216	32,493	42,287	53,118	23,270	32,416	41,942	52,847
	M2	ART	1.243	1.082	1.087	0.895	1.244	1.086	1.055	0.870
		ART less (%)	-1.61	-2.73	-4.45	-1.11	-2.01	-3.30	3.10	-1.73
		R	23,209	32,394	41,774	52,334	23,255	32,335	41,646	52,208
		R/R^*	1.00	1.00	0.99	0.99	1.00	1.00	0.99	0.99
		$W1/W2$	1.0	1.0	1.0	1.0	0.8	1.0	0.9	0.9
UBS	M1	ART	0.957	0.812	1.184	0.887	0.954	0.799	1.160	0.867
		R^*	24,068	36,076	56,885	69,695	24,095	35,969	56,363	68,757
	M2	ART	0.739	0.758	1.158	0.879	0.882	0.750	1.138	0.859
		ART less (%)	-22.74	-6.57	-2.24	-0.85	-7.55	-6.16	-1.89	-0.86
		Reward	23,958	35,853	55,210	68,907	24,025	35,919	56,050	68,541
		R/R^*	1.00	0.99	0.97	0.99	1.00	1.00	0.99	1.00
		$W1/W2$	0.6	0.5	0.5	0.5	0.5	0.6	0.5	0.5
	$c_{12} = c_{21} = 2$									
			Balanced origins				Unbalanced origins			
			Light load		Heavy load		Light load		Heavy load	
		Case 17	Case 18	Case 19	Case 20	Case 21	Case 22	Case 23	Case 24	
BS	M1	ART	2.211	2.134	2.162	2.097	2.203	2.163	2.004	1.924
		R^*	23,177	32,628	42,146	53,075	23,221	32,679	41,925	52,817
	M2	ART	1.518	1.451	1.623	1.433	1.542	1.478	1.697	1.555
		ART less (%)	-31.32	-32.02	-24.92	-31.67	-30.00	-31.66	-15.31	-19.21
		R	23,000	32,125	41,270	51,565	22,977	32,441	41,177	51,873
		R/R^*	0.99	0.98	0.98	0.97	0.99	0.99	0.98	0.98
		$W1/W2$	1.0	1.0	1.0	1.0	0.7	1.0	0.8	0.9
UBS	M1	ART	1.860	1.720	2.152	1.882	1.834	1.660	1.949	1.669
		R^*	24,003	35,912	55,140	69,597	24,021	35,934	55,234	68,544
	M2	ART	1.114	1.032	1.613	1.333	1.025	0.887	1.492	1.218
		ART less (%)	-40.09	-39.97	-25.07	-29.16	-44.11	-46.59	-23.47	-27.01
		R	23,994	35,860	55,056	68,708	24,045	35,953	55,151	68,016
		R/R^*	1.00	1.00	1.00	0.99	1.00	1.00	1.00	0.99
		$W1/W2$	0.8	0.7	0.5	0.5	0.5	0.6	0.4	0.4

- Cases 7, 15, 23, 31: $\lambda_{11} = 1.0$, $\lambda_{21} = 5.0$, $\lambda_{12} = \lambda_{22} = 3.0$.

- Cases 8, 16, 24, 32: $\lambda_{11} = \lambda_{22} = 2.5$, $\lambda_{12} = 2.5$, $\lambda_{21} = 7.5$.

The numerical results are shown in Table 1, where M1 denotes the standard centralized control approach, M2 denotes the CDDA approach, R^* stands for the optimal reward, BS stands for balanced SPs and UBS stands for unbalanced SPs, $W1$ and $W2$ stand for the workloads of SP1 and SP2, respectively. The system-wide reward obtained with the standard centralized approach is the optimal value.

First, one may observe that

$$\frac{W1}{W2} \approx \frac{N_1}{N_2}.$$

This result verifies the validity of the virtual arrival rate (see Eq. (7)) assumption, that is, under the dispatching policy with decentralized admissions, the workloads are well balanced across the SPs.

Second, from Table 1, it is clear that for all the cases, the system-wide rewards earned by using the proposed CDDA approach are more than 97% of the corresponding optimal values. This shows that the CDDA approach performs very well in all cases in terms of rewards. The average ratio of the reward earned with the CDDA approach to the optimal value for the 32 cases is 99%. For the ART, the CDDA approach clearly dominates the centralized control approach.

Third, one may observe that the ART reductions by using the CDDA approach in Cases 17–32 are more than the corresponding ART reductions in Cases 1–16. This shows that the CDDA approach leads to a better QoS especially when the propagation delay is high. The average improvement in QoS (reduction of ART) for the 32 cases is 17%. In the extreme cases when the propagation delay accounts for more than 50% of the ART, the average improvement of QoS for the Cases 17–32 is 31%.

Finally, we can estimate the customer utility from Eq. (8). We assume that $C(t) = t$, and $p = 2$ for impatient customers and $C = 10$ for patient customers. We set $a = 1$ for approach M1 and $a = R/R^*$ for approach M2, after normalization to the optimal acceptance rate. From Table 1, the customer utilities are given by: for Cases 1–8, (0.792, 0.805), (0.899, 0.924), (0.879, 0.850, 0.903), (1.105, 1.014, 1.211), (**0.804, 0.812**), (0.890, 0.921), (**0.978, 0.874, 0.939**) (1.129, 1.040, 1.135); for

Cases 9–16, (1.045, 1.353), (1.232, 1.176, 1.298), (**0.845, 0.703, 0.845**), (**1.127, 1.031, 1.123**), (1.048, 1.134), (1.252, 1.333), (0.862, 0.815, 0.872); (1.153, 1.164); for Cases 17–24, (0.452, 0.624, 0.657), (0.469, 0.671, 0.684), (0.463, 0.558, 0.613), (0.477, 0.592, 0.690), (0.454, 0.615, 0.647), (0.462, 0.640, 0.674), (0.499, 0.537, 0.594), (0.520, 0.580, 0.639); and for Cases 25–32, (0.538, 0.898), (0.581, 0.969), (0.465, 0.620), (0.531, 0.704, 0.746), (0.545, 0.952), (0.602, 1.127), (0.513, 0.670), (0.599, 0.766, 0.816). Here, the first number in the triplets is the utility for approach M1, the second and third numbers are the utilities for approach M2 when the customers are patient or impatient, respectively. When there are only two numbers, the third number is equal to the second and is omitted. The absolute values are not important. What is important is the comparison of the three numbers in a triplet, which indicates the relative performance of different approaches. We note that except for a few cases (in bold), the CDDA approach almost always gives better customer utilities and for cases 17–32, the improvements of the CDDA approach over the centralized approach are dramatic.

Our simulation results show that the number of accepted requests is near optimal, which means that the number of rejected requests for the CDDA approach is near the minimum. The ART decreases significantly for most cases. The customer utility also improves significantly in most cases with the CDDA approach. Therefore, our algorithm leads to much better customer satisfaction than standard Centralized control mechanism at a minimum cost in terms of the system reward. This demonstrates the significant advantage of the CDDA approach when customer satisfaction is as important as the system reward. We note that in practice the fulfillment of a more profitable request usually earns a higher reward. Under such reward scheme, more unprofitable requests will be rejected. The total customer utility will therefore decrease, but the operator's profit will be maximized. The above analysis shows that our algorithm has enough flexibility to achieve customers' satisfaction but it is able to go beyond that point by always achieving maximum service provider's profit.

4.2. Comparison of standard CDDA and the SBS

In this subsection, the performance of the SBS is examined using one SP. To guarantee the minimum QoS, the maximum connections are limited to 30 per server in the numerical example (At the HKUST library, the Web reservation system used 30 as the maximum connections. But for different systems, this number is different). Again, there are two classes of arrivals with the corresponding service rates $\mu_1 = 5$ and $\mu_2 = 2$. The discount rate is set to $\alpha = 0.9999$. Four arrival rates combinations are used to compute the optimal policies with both the standard CDDA and the SBS. The following figures show the corresponding optimal policies. In each figure, the area below the curve is the acceptance area. It is obvious that the policy computed using the SBS is much tighter towards larger files than the corresponding policy computed using the standard CDDA approach. It should be pointed out that the ratio of the acceptance regions from the two approaches is different from the realized ratio of the numbers of requests accepted using the two approaches. Since the maximum connections are finite, a tighter control towards one type of requests may actually lead to higher number of serviced requests. Such more proactive control (rejection before buffer overflow)

benefits more when the request arrival stream has a higher variability, as demonstrated in Table 2.

Case 1. Heavy workloads and balanced arrival rates: $\lambda_{1j} = \lambda_{2j} = 3$.

Case 2. Light workloads and unbalanced arrival rates: $\lambda_{1j} = 1$ and $\lambda_{2j} = 0.5$.

Case 3. Heavy workloads and unbalanced arrival rates: $\lambda_{1j} = 4$ and $\lambda_{2j} = 1$.

Case 4. Light workloads and balanced arrival rates: $\lambda_{1j} = \lambda_{2j} = 1$.

In the numerical results reported below, both exponential and Weibull interarrival and service times are considered. For the Weibull distribution, the location parameter and shape parameter are set to 0 and 0.5, respectively. By doing this, the Internet-type power-tail interarrival/service time is generated. The optimal solutions for the four cases are summarized in Table 2, where M2 denotes the standard CDDA approach, M3 denotes the SBS, R denotes the rewards, and C1, C2, C3, and C4 denote the four cases, respectively. Here, the simulation is conducted on a Pentium III 733 and the control policies are computed using the standard value iteration approach.

Obviously, when the interarrival time follows an exponential distribution and the policy is

Table 2
Comparison of the standard CDDA approach and the SBS approach

C	M	Arrival type	R (unit)	ART (min)	Percentage of R^*
C1	M2	Expo	22,968	4.728	100.0
		Weibull	21,128	5.403	92.0
	M3	Expo	18,000	0.490	78.4
		Weibull	21,356	1.778	93.0
C2	M2	Expo	8951	0.578	100.0
		Weibull	8887	1.620	99.3
	M3	Expo	8927	0.588	99.7
		Weibull	8875	1.498	99.2
C3	M2	Expo	26,204	3.469	100.0
		Weibull	23,598	3.628	90.1
	M3	Expo	23,996	1.124	91.6
		Weibull	23,924	2.438	91.3
C4	M2	Expo	12,106	1.308	100.0
		Weibull	11,480	3.880	94.8
	M3	Expo	12,086	1.201	99.8
		Weibull	11,206	2.379	92.6

computed by conventional approach, the reward obtained should be optimal. When the variance of the interarrival time increases, the optimal reward should be lower. From Table 2, it is clear that the reduction of the reward caused by higher variance of the interarrival time can be partly neutralized by applying SBS when the workload is heavy. If the workload is light, the rewards earned by the two approaches are nearly the same. Notice that the computation time required for the standard CDDA approach could be as long as 6.2×10^3 min, far more than the maximum computation time available in a real environment. On the other hand, the time required for completing a policy iteration with the SBS approach is only a few minutes, shorter than the fix time interval for the new policy generation.

5. Conclusion

Web-server network control problem is an important research area for emerging e-commerce and e-business applications. Most works on this subject have appeared in the literature over the last five years. With the rapid development of the Internet, new problems and challenges have been emerging continuously. This paper addresses one of the important issues: reward-based admission and routing control.

In order to implement the reward-based control, a content-based controller with redirect dispatching and decentralized admission is proposed. This mechanism is especially suitable for multimedia Web sites. Without using additional hardware, this mechanism achieves content-based control with the HTTP-redirect technique and CGI program (see details of implementation plan in [15]). Because of the relatively long queueing and transmission delay for multimedia service, the control overhead due to the CGI computing and redirection can be ignored. This control mechanism has five main advantages: (1) It is a reward-based control. Since different services have different profits, the reward-based control can help service providers to maximize profits. (2) The server side overhead is quite low. Decentralized mechanism provides an agile control with lower

server side overhead. Moreover, with the HTTP-redirect technique the central dispatcher does not need to forward packages and therefore it will not become the system bottleneck. (3) This mechanism leads to a lower response time. Since the propagation delay is considered by the dispatcher, the average response time is shorter than that achieved by the traditional load balancing control, which means better quality of service (QoS). (4) This mechanism is open to other local control algorithms. Any local routing algorithms can be combined with the proposed control approach since the proposed control mechanism considers each LAN as a SP. (5) There are no additional hardware requirements.

The reward-based control approach is developed from an MDP model. This approach combines a distributed control strategy that maximizes individual SP's reward and a greedy algorithm that optimizes the system quality of service. Simulation results show that this approach has much better QoS than pure centralized control with a minimum reward loss in most of the test cases. A state-block numerical scheme is proposed to reduce the state space. This scheme is also motivated by the observation that in the Internet environment, if the number of a class of requests in an SP changes, it is more likely to change continuously on one direction in a very short interval. The numerical results show that the policy computed by this approach leads to higher reward and lower response time than the standard MDP solving approach when the inter-event time follows the Internet type distribution.

We do not specifically address the issue of service quality verses fee for the service. But such a relation is implicitly included in the model in the sense that when a service is rendered (realized), the customer is guaranteed a waiting time smaller than a maximum waiting time enforced by the buffer size. Otherwise, the request is rejected and no fee is collected. The expected waiting time is also within a certain level.

Further studies of the SBS are needed to investigate how to determine the appropriate block sizes for different network systems and operation environments. Also, the block size might need to be determined with the class of requests.

This may involve extensive numerical and simulation experiments. The control algorithm may be modified to extend the reward-based control to common Web-server networks. In this case, the number of classes will increase and the definition of reward becomes fuzzy. New approaches are also needed to handle problems with large request class space.

Acknowledgements

This research is supported in part by Hong Kong Research Grant Council through a grant N_HKUST023/00. The authors would like to thank two anonymous referees. Their comments and suggestions on an earlier version of this paper have helped us to improve both the content and style of the paper.

References

- [1] D. Andresen, T. Yang, O.H. Ibarra, Toward a scalable distributed WWW server on workstation clusters, *Journal of Parallel and Distributed Computing* 42 (1) (1997) 91–100.
- [2] M. Arlitt, C. Williamson, Internet Web servers: workload characterization and performance implications, *IEEE/ACM Transactions on Networking* 5 (5) (1997) 631–645.
- [3] P. Barford, M. Crovella, Critical path analysis of TCP transactions, *IEEE/ACM Transactions on Networking* 9 (3) (2001) 238–248.
- [4] N. Brownlee, K.C. Claffy, Understanding Internet traffic streams: dragonflies and tortoises, *IEEE Communications Magazine* 40 (10) (2002) 110–117.
- [5] V. Cardellini, M. Colajanni, S.P. Yu, DNS dispatching algorithms with state estimators for scalable Web-server clusters, *World Wide Web* 2 (3) (1999) 101–113.
- [6] V. Cardellini, M. Colajanni, P.S. Yu, Dynamic load balancing on Web-server systems, *IEEE Internet Computing* 3 (3) (1999) 28–39.
- [7] S.-H.G. Chan, F.A. Tobagi, Threshold-based admission policies for video services, in: *Seamless Interconnection for Universal Services Global Telecommunications Conference GLOBECOM'99* (Cat. No.99 CH37042), Vol. 4, IEEE, Piscataway, NJ, USA, 1999, pp. 2076–2080.
- [8] I.R. Chen, C. Chen, Threshold-based admission control policies for multimedia servers, *The Computer Journal* 39 (9) (1997) 757–766.
- [9] M. Colajanni, P.S. Yu, D.M. Dias, Analysis of task assignment policies in scalable distributed Web-server systems, *IEEE Transactions and Distributed Systems* 9 (6) (1998) 585–600.
- [10] C.M. Harris, P.H. Brill, M.J. Fischer, Internet-type queues with power-tailed interarrival times and computational methods for their analysis, *INFORMS Journal on Computing* 12 (14) (2000) 261–271.
- [11] G.D.H. Hunt et al., Network dispatcher: a connection router for scalable Internet service, *Computer Networks and ISDN Systems* 30 (1998) 347–357.
- [12] E.D. Katz, M. Butler, R. McGrath, A scalable HTTP server: the NCSA prototype, *Computer Networks and ISDN Systems* 27 (1994) 155–164.
- [13] T.T. Kwan, R.E. McGrath, D.A. Reed, NCSA's World Wide Web server: design and performance, *Computer* 28 (11) (1995) 68–74.
- [14] F.Y-S. Lin, Optimal real-time admission control algorithms for the video-on-demand (VOD) service, *IEEE Transactions on Broadcasting* 44 (4) (1998) 402–408.
- [15] L. Liu, Y. Lu, Dynamic traffic control for Web-server networks, Department of Industrial Engineering and Engineering Management, HKUST, 2001.
- [16] D. Mosedale, W. Foss, R. Mccool, Lessons learned administering Netscape's Internet site, *IEEE Internet Computing* 1 (2) (1997) 28–35.
- [17] M.L. Puterman, *Markov Decision Process*, Wiley-Interscience, New York, 1994.
- [18] S. Steinke, Network delay and signal propagation, *Network Magazine* (May 2001).



Liming Liu received his B.Eng. in Shipbuilding Engineering and M.Sc. in Management from Huazhong University of Science and Technology (HUST). He received his Ph.D. in operations research from University of Toronto. He taught at HUST and Toronto before joining Hong Kong University of Science and Technology in 1993, where he is now an associate professor in the department of industrial engineering and engineering management. His research interests include queueing networks, supply chains, e-commerce, Internet application systems, and inventory control.



Yumao Lu received a B.Eng. in Automatic Control from the Huazhong University of Science and Technology, Wuhan, China in 1999, and an M.Phil. in Industrial Engineering and Engineering Management from the Hong Kong University of Science and Technology, Hong Kong, China in 2001. He is currently a Ph.D. candidate in Electrical Engineering at University of California—Los Angeles (UCLA), Los Angeles, CA. From 1999 to 2001, he was a research assistant under the supervision of Prof. Liming

Liu. Since 2001, he has been working as a research assistant and teaching assistant/associate in Electrical Engineering at UCLA. His research interests include stochastic operations research in communications, global optimization in machine learning, statistic signal processing and pattern recognition.