Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 186 (2008) 681–693
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Stochastics and Statistics

Web server load balancing: A queueing analysis

Zhongju Zhang [a,*], Weiguo Fan [b]

[a] Department of Operations and Information Management, School of Business, University of Connecticut, Storrs, CT 06269, United States
[b] Department of Accounting and Information Systems, Pamplin College of Business, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, United States

Received 21 December 2005; accepted 1 February 2007
Available online 19 March 2007

Abstract

Over the last few years, the Web-based services, more specifically different types of E-Commerce applications, have become quite popular, resulting in exponential growth in the Web traffic. In many situations, this has led to unacceptable response times and unavailability of services, thereby driving away customers. Many companies are trying to address this problem using multiple Web servers with a front-end load balancer. Load balancing has been found to provide an effective and scalable way of managing the ever-increasing Web traffic. However, there has been little attempt to analyze the performance characteristics of a system that uses a load balancer. This paper presents a queuing model for analyzing load balancing with two Web servers. We first analyze the centralized load balancing model, derive the average response time and the rejection rate, and compare three different routing policies at the load balancer. We then extend our analysis to the distributed load balancing and find the optimal routing policy that minimizes the average response time.

Keywords: Load balancing; Parallel queues; Queueing theory; Routing

1. Introduction

The Web traffic has experienced tremendous growth in the last few years. Content providers and E-Commerce merchants are often overwhelmed by the number of request for Web pages and online transactions, resulting in considerable degradation (for example, long waiting time) of the Web server performance. In order to keep the response time within an acceptable limit, Web server administrators often limit the number of simultaneously open connections. When the number of request exceeds this limit, all later user access requests will be denied. A message similar to the one shown in Fig. 1 is quite common. Therefore, often either the waiting time is high, or the user request is not processed at all. Either situation could lead to disgruntled customers and drive many of them away from the site. To complicate things further, the traffic may change

* Corresponding author. Tel.: +1 203 2519583; fax: +1 203 2519541.
E-mail addresses: john.zhang@business.uconn.edu (Z. Zhang), wfan@vt.edu (W. Fan).

0377-2217/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2007.02.011

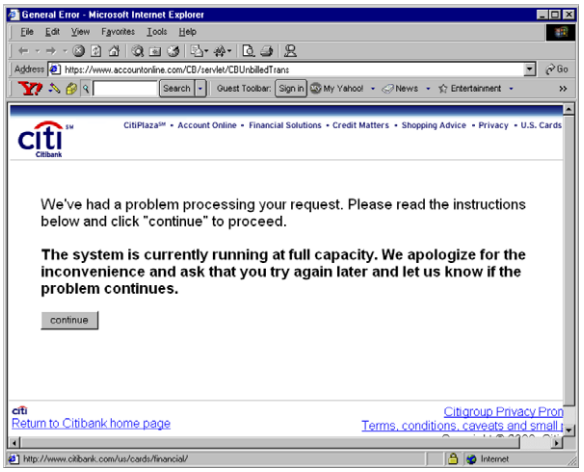

Fig. 1. Service Unavailable at the CitiBank.

with the time of the day, the day of the week, or even the month of the year. These seasonal or periodic fluctuations make it even more difficult to plan for adequate capacity for a site.

There are several possible solutions to this issue, such as adding powerful servers, caching, or outsourcing (Derfler and Freed, 2000). However, these methods of load management have their own problems. For example, a more powerful server may work for a while, but it is not scalable and could cause interruption due to server upgrade and maintenance. Besides, if the server capacity is planned based on the peak load, then the extra capacity is wasted during the off-peak hours. Outsourcing, on the other hand, has a very high price tag, and yet one has limited control over the Quality of Service (QoS). Caches have been used in a number of different ways to address the issue of slow response time resulted from the overloaded servers. Standing somewhere along the path between the Web server and the browsers, the caches intercept requests for Web content, and attempt to respond to the requests whenever possible. When these requests cannot be served from the caches, they are forwarded to the Web server. The presence of *dynamic* content featured on most Web sites raises significant barriers to caching (Datta et al., 2003).

A realistic and effective way is to use multiple Web servers also called "clustered Web servers" or "server farm" and balance the load among these servers. In order to address the network latency delays caused over greater distances, many organizations are also deploying multiple (distributed) Web servers in disparate cities, states, and even countries. User service requests are routed to a server based on some routing algorithms. Of course, the system performance depends critically on these routing algorithms. This method of load management has been shown to improve QoS in practice and is widely used (Derfler and Freed, 2000). One big advantage of using multiple servers is that one need not develop very accurate plans for the server capacity; one can add to the existing capacity in an ad-hoc fashion by either buying new servers or by employing unused capacity from elsewhere.

In this paper, we develop a queueing model to quantitatively analyze the performance characteristics of different routing policies for a clustered Web servers. The main practical contribution of this paper is that it provides network administrators guidelines on how to configure the server farm to achieve maximum efficiency. The main theoretical contributions of this paper are twofold. First, we obtain performance measures (e.g., joint probability distribution, average waiting time, rejection rate) for the shortest-queue load balancing with finite buffer capacity and compare them with the random routing and the round-robin policy. Second, we extend our analysis to the distributed load balancing which could re-route traffic from heavy-loaded regions to light-loaded ones. We consider the proximity which is measured by the network delay due to re-routing, and find the optimal routing policy that minimizes the average response time.

The remainder of the paper is organized as follows. Section 2 discusses related prior research. Section 3 describes the queueing model for the centralized load balancing. Three different routing policies (i.e. the random, the round-robin and the shortest-queue) are considered and numerical results are presented. The distrib-

uted load balancing and its numerical results are presented in Section 4. Section 5 concludes the paper and offers directions for future research.

2. Literature review

Balancing/routing and scheduling issues have been studied quite extensively in the queueing literature. Among others, Flatto and McKean (1977), Haight (1958), Halfin (1985), Kingman (1961), and Koenigsberg (1966) study the shortest-queue policy in two parallel servers with infinite buffer size. In the case of Web servers, there is usually a limit to the number of simultaneous open connections. Hence, the results from those papers cannot be directly used in our problem context. Conolly (1984) later extends the model by Flatto and McKean (1977), Haight (1958) to a case with finite buffer capacity, but assuming the buffer size is an even number. Zheng and Zipkin (1990) analyze a scheduling model, which deals with two parallel queues, each with its own arrival stream, served by a single server. They assume that each queue has an unlimited buffer capacity. Pourbabai and Seidmann (1992) consider routing of requests to parallel servers with finite capacity. However, they assume that a new request will be routed to a server with a fixed probability, and hence the queueing model is similar to the M/M/1/K queue. Sparaggis et al. (1993) show that, under certain conditions, the scheduling and the routing problems are dual. One crucial assumption they made is that the scheduling policy needs to be *preemptive* when the difference of the queue length between two servers is greater than one.

In the context of Web server load balancing, many different scheduling policies have been proposed. For example, Chen and Choi (2001) study the issue of allocating Web documents among servers so that the load is balanced as equally as possible. Wolf and Yu (2001) study a resource allocation problem and devise algorithms that balance the load on a cluster of servers so as to minimize the response time on the system. Ciardo et al. (2001) present a strategy to achieve a balanced load for a clustered Web servers, based on the size distributions of the requested documents. Comparative analyses (with random allocation and the shortest-queue policy) show the effectiveness of the EquiLoad allocation policy. Teo and Ayani (2001) analyze and compare the round-robin, the least connected, and the least loaded scheduling algorithms for a cluster of centralized Web servers through simulation. Aweya et al. (2002) describe an overload control scheme that integrates both the admission control and the load balancing for Web servers. Colajanni and Yu (2002) and Shi et al. (2005) propose scheduling mechanisms that achieve load balancing among Web servers.

Oates and Corne (2001) discuss distributed load balancing where requests from busy regions could be re-routed to quieter ones. They focus on issues such as algorithm tuning, scalability, and performance by employing a combinatorial optimization technique. Sanghi et al. (2002) propose a load balancing policy of using the proximity information in geographically distributed Web servers. The proximity is measured by the round-trip delay between the Web servers and the clients. Menasce (2002) examine the trade-off of two approaches (i.e. adding more servers or upgrading the capacity of the servers) to scaling Web clusters under different circumstances. Down and Lewis (2006) consider parallel queues where decision makers can move customers from one queue to another so as to minimize the long-run average cost. Karaul et al. (2000), on the other hand, propose a technique that allows clients to make routing decisions so as to achieve the goal of load balancing among servers.

3. The centralized load balancing model

As shown in Fig. 2, we consider two identical Web servers, WS_1 and WS_2. Each server has a finite buffer of K spaces. User requests first come to a central load balancer, which directs them to one of the two queues using a routing policy. Three routing policies are mostly used:

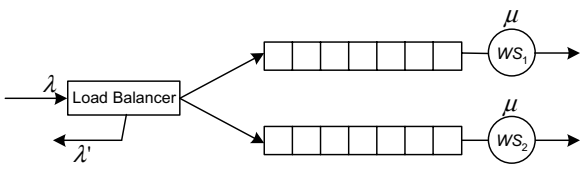

Fig. 2. Queueing model for centralized load balancing.

1. the *random* policy where the server is chosen randomly,
2. the *round-robin* policy where incoming requests alternate between two servers, and
3. the *shortest-queue* policy where the server with the shortest-queue is chosen.

We assume that the arrival of user requests follows a Poisson process with an arrival rate of λ and the service time for each Web server is exponentially distributed with service rate of μ. The load balancer is assumed to have an infinite capacity and know the queue length exactly; it can immediately route a request to one of the queues without delay. Requests at each server are processed on a first-come first-served (FCFS) basis. Incoming requests are rejected only if both queues are full; the rate of rejection is λ'. Note that we do not require $\rho = \frac{\lambda}{2\mu} < 1$ because we have finite buffer in each queue, and requests are rejected if the queues are full.

3.1. The random policy

Analysis of the random policy is straightforward. If the overall arrival follows the Poisson process with rate λ, and if the arrival stream is randomly split into two, then the arrival process to each server is also a Poisson process with rate $\lambda/2$ (Ross, 1993). In other words, we are faced with two identical M/M/1/K queues. Let us consider the first queue. The probability of having n requests in that system (i.e., either in queue or in service) is given by (Gross and Harris, 1998)

$$p_n = \frac{(1-\rho)\rho^n}{1-\rho^{K+1}}, \quad 0 \leqslant n \leqslant K.$$

The normalized rejection rate (NRR) of the whole system can now be calculated easily:

$$\mathrm{NRR} = \frac{\lambda'}{\lambda} = \frac{1}{2}p_K + \frac{1}{2}p_K = \frac{(1-\rho)\rho^K}{1-\rho^{K+1}}.$$

The average number of requests in each server is given by

$$N = \sum_{i=0}^{K} i \cdot p_i = \begin{cases} \frac{\rho}{1-\rho} - \frac{(K+1)\rho^{K+1}}{1-\rho^{K+1}}, & \rho \neq 1, \\ \frac{K}{2}, & \rho = 1. \end{cases}$$

The average response time in the system can then be calculated by dividing $2N$ by the effective arrival rate:

$$W = \frac{2N}{\lambda - \lambda'} = \frac{2N}{\lambda(1-p_K)} = \frac{2N(1-\rho^{K+1})}{\lambda(1-\rho^K)}.$$

3.2. The round-robin policy

Since the arrival process of user requests is a Poisson process, the inter-arrival times of requests are I.I.D exponential random variables with mean $\frac{1}{\lambda}$, so the inter-arrival times to each Web server are Erlang type-2 distributed with a mean of $\frac{2}{\lambda}$. Therefore, we are faced with two identical $E_2/M/1/K$ queues. Let us consider the first queue. Following the analysis of $E_k/M/1$ queue in Gross and Harris (1998), let q_{nj} represent the proba-

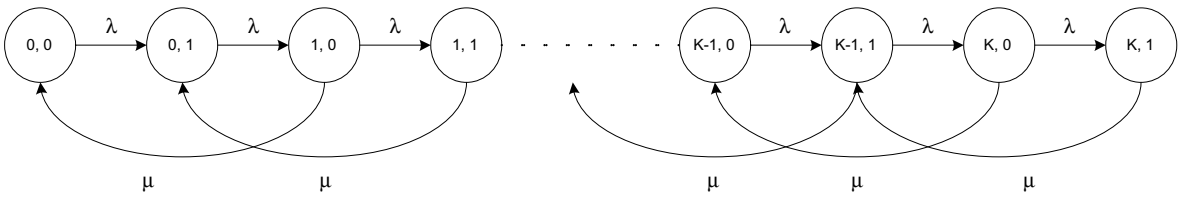

Fig. 3. State transition diagram—round-robin policy.

bility of n customers in the system and an arrival being in phase j ($j = 0, 1$), then it follows that the probability of n customers in the system, p_n, is given by $p_n = q_{n0} + q_{n1}$.

The transition state diagram of the queueing model is depicted in Fig. 3, the balancing equations are given by

$$\begin{aligned}
q_{n-1,1} &= \left(1+\frac{1}{2\rho}\right)q_{n0} - \frac{1}{2\rho}q_{n+1,0}, \quad 1 \leqslant n \leqslant K-1,\\
q_{n0} &= \left(1+\frac{1}{2\rho}\right)q_{n1} - \frac{1}{2\rho}q_{n+1,1}, \quad 1 \leqslant n \leqslant K-1,\\
q_{K-1,1} &= \left(1+\frac{1}{2\rho}\right)q_{K0},\\
q_{K,0} &= \frac{1}{2\rho}q_{K1},\\
q_{01} &= q_{00} + \frac{1}{2\rho}q_{11},\\
q_{00} &= \frac{1}{2\rho}q_{10}.
\end{aligned} \tag{1}$$

While it is difficult to obtain the closed-form solutions for q_{nj} from the balancing equations, we can easily calculate q_{nj} numerically, and hence the expected performance measures as $\mathrm{NRR} = p_K$, and $W = \frac{2N}{\lambda(1-p_K)}$, where $N = \sum_{n=0}^{K} n p_n$.

3.3. The shortest-queue policy

The load balancer, upon getting a request, sends it to one of the two Web servers with the least number of requests in queue. If there is a tie, the request is routed randomly. Let N_i be the number of requests in the system WS_i, $i = 1, 2$. The current state of the system can be denoted by (N_1, N_2). Let us define $p_{ij} = \Pr[N_1 = i, N_2 = j]$. Clearly, $p_{ij} = 0$ if either $i, j < 0$ or $i, j > K$. The transition state diagram for this model is shown in Fig. 4. Balancing equations for this model gives us

$$\begin{aligned}
(1+\rho)p_{ij} &= \frac{1}{2}(p_{i+1,j} + p_{i,j+1}) + \rho Q_{ij}, \quad 1 \leqslant i \leqslant K, \;\; 1 \leqslant j \leqslant K,\\
\left(\frac{1}{2}+\rho\right)p_{0j} &= \frac{1}{2}(p_{0,j+1} + p_{1j}), \quad 1 < j \leqslant K,
\end{aligned}$$

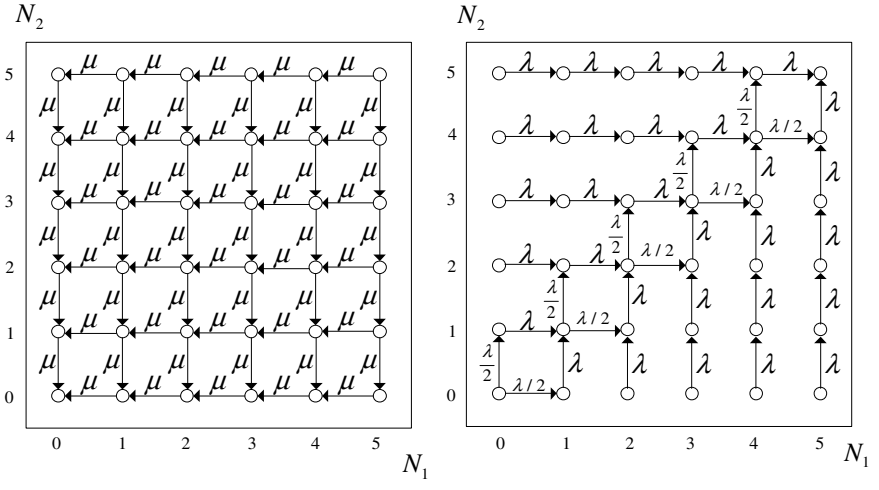

Fig. 4. Schematic of state transition—shortest-queue policy.

$$
\begin{aligned}
&\left(\frac{1}{2}+\rho\right)p_{i0} = \frac{1}{2}(p_{i+1,0}+p_{i1}), \quad 1 < i \leqslant K,\\
&\left(\frac{1}{2}+\rho\right)p_{01} = \frac{1}{2}(p_{02}+p_{11}) + \frac{1}{2}\rho p_{00},\\
&\left(\frac{1}{2}+\rho\right)p_{10} = \frac{1}{2}(p_{11}+p_{20}) + \frac{1}{2}\rho p_{00},\\
&\rho p_{00} = \frac{1}{2}(p_{01}+p_{10}),\\
&p_{KK} = \rho(p_{K-1,K}+p_{K,K-1}),
\end{aligned}
\tag{2}
$$

where Q_{ij} depends on the difference of i and j, and is given by

$$
Q_{ij} = \begin{cases} p_{i,j-1}, & i \geqslant j+2,\\ p_{i-1,j}, & j \geqslant i+2,\\ p_{i,j-1}+\frac{1}{2}p_{i-1,j}, & i = j+1,\\ \frac{1}{2}p_{i,j-1}+p_{i-1,j}, & j = i+1,\\ p_{i-1,j}+p_{i,j-1}, & i = j. \end{cases}
$$

We wish to solve the above simultaneous equations in (2) to obtain p_{ij}, and then calculate the rejection rate and the average response time. The difficulty in finding p_{ij} is that one needs to solve a set of $(K+1)\times(K+1)$ simultaneous equations, which could take a long time for large K. Furthermore, we have no prior knowledge about the distribution of total number of requests in the system.[1]

Our approach of solving p_{ij}'s is to start with $i = K$, and calculate p_{Kj}'s ($\forall j \in [0, K]$) with respect to p_{KK}. We then recursively calculate the other p_{ij}'s where $i \geqslant j$ (because $p_{ij} = p_{ji}$) by decreasing i.

Let us define $x_i(\delta) = p_{i,i-\delta}$. For $i \geqslant j+2$, let $\delta = i-j$ ($2 \leqslant \delta \leqslant K-1$). Eq. (2) gives us

$$
\rho x_i(\delta+1) - (1+\rho)x_i(\delta) + \frac{1}{2}x_i(\delta-1) = -\frac{1}{2}x_{i+1}(\delta+1). \tag{3}
$$

When $i = K$, the right hand side of Eq. (3) is zero. So we have the following homogeneous difference equations:

$$
\rho x_K(\delta+1) - (1+\rho)x_K(\delta) + \frac{1}{2}x_K(\delta-1) = 0, \quad \text{where } 2 \leqslant \delta \leqslant K-1.
$$

The general solution to the above homogenous equations is

$$
x_K(\delta) = C_1\zeta_1^{\delta-1} + C_2\zeta_2^{\delta-1}, \quad \text{where } \zeta_1 = \frac{1+\rho-\sqrt{1+\rho^2}}{2\rho} \quad \text{and} \quad \zeta_2 = \frac{1+\rho+\sqrt{1+\rho^2}}{2\rho}.
$$

But $\zeta_2 > 1$, so $C_2 = 0$. To solve C_1, we have $(1+\rho)p_{K,K-2} = \frac{1}{2}p_{K,K-1} + \rho p_{K,K-3}$. This gives us (let $\zeta = \zeta_1$):

$$
C_1 = \frac{p_{K,K-1}}{2[(1+\rho)\zeta - \rho\zeta^2]}.
$$

Furthermore, from the boundary condition in (2), we get

$$
p_{K0} = \frac{C_1\zeta^{K-2}}{1+2\rho}.
$$

[1] It should be noted that the distribution of total number of requests in our model is different from that in the single server model (M/M/1/2K).

Putting all of the above together, we have

$$\begin{aligned} &x_K(0) = p_{KK}, \quad x_K(1) = p_{K,K-1} = \frac{p_{KK}}{2\rho}, \quad x_K(K) = p_{K0} = \frac{C_1\zeta^{K-2}}{1+2\rho}, \\ &x_K(\delta) = C_1\zeta^{\delta-1} = \frac{1}{4\rho}\frac{p_{KK}}{(1+\rho)\zeta - \rho\zeta^2}\zeta^{\delta-1}, \quad \text{where } 2 \leqslant \delta \leqslant K-1. \end{aligned} \tag{4}$$

Now, define $\beta = \frac{1}{1-2\rho\zeta^2}$; this is valid as $1 - 2\rho\zeta^2 \neq 0$. We can recursively calculate a set of parameters a_{ij} starting with

$$a_{KK} = \frac{p_{K,K-1}}{2[(1+\rho)\zeta - \rho\zeta^2]},$$

$$a_{in} = \zeta^2 \sum_{m=n}^{K} \beta^{m-n+1} a_{i+1,m}, \quad \text{where } i < n \leqslant K,$$

$$a_{ii} = x_i(1) - \sum_{n=i+1}^{K} a_{in}.$$

From Eq. (2), we get

$$\begin{aligned} &x_i(0) = p_{ii} = \frac{2(1+\rho)p_{i+1,i} - p_{i+2,i} - p_{i+1,i+1} - 2\rho p_{i+1,i-1}}{\rho}, \\ &x_i(1) = p_{i,i-1} = \frac{(1+\rho)p_{ii} - p_{i+1,i}}{2\rho}. \end{aligned} \tag{5}$$

We define

$$x_i(\delta) = \sum_{n=i}^{K} a_{in} \binom{n-i+\delta-1}{n-i} \zeta^{\delta-1}, \quad \text{where } 2 \leqslant \delta \leqslant i-1 \quad \text{and} \quad 0 \leqslant i < K. \tag{6}$$

The boundary condition in (2) gives us

$$x_i(i) = p_{i0} = \frac{p_{i+1,0} + p_{i1}}{1+2\rho}. \tag{7}$$

We can show that $x_i(\delta)$ in (5)–(7) satisfy the set of equations in (2) (see the Appendix for proof).

Therefore, Eqs. (4)–(7) allow us to recursively compute p_{ij}'s ($0 \leqslant i \leqslant K$, $i \geqslant j$) in terms of p_{KK}. Using the boundary condition that $\sum_{i=0}^{K}\sum_{j=0}^{K} p_{ij}$ must sum to 1, we can find p_{KK}; and hence all the probabilities p_{ij}. The performance measures for this model can then be obtained as

$$\text{NRR} = p_{KK}, \quad \text{and} \quad W = \frac{N}{\lambda(1-p_{KK})}, \quad \text{where } N = \sum_{i=0}^{K}\sum_{j=0}^{K}(i+j)p_{ij}. \tag{8}$$

3.4. Numerical results

In this section, we compare the three policies analyzed in the previous section. As a benchmark, we consider the ideal case with the best possible performance: a single server with twice the capacity of the individual servers with a total buffer space of $2K$. We wish to compare the three policies in terms of normalized rejection rate (NRR) and the average response time in system (W).

For a single server with twice the capacity of individual servers, following M/M/1/2K queueing model, the probability of having n requests in the system is given by (Gross and Harris, 1998)

$$p_n = \frac{(1-\rho)\rho^n}{1-\rho^{2K+1}}, \quad 0 \leqslant n \leqslant 2K.$$

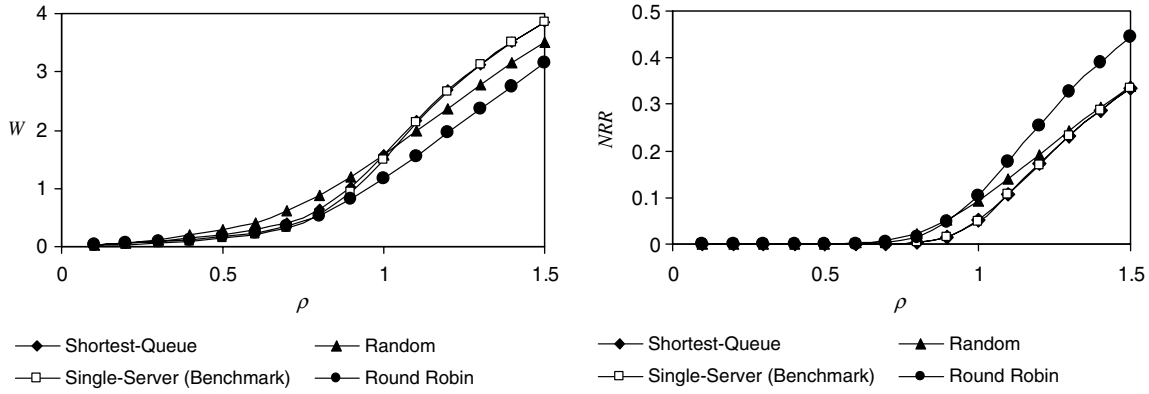

Fig. 5. Comparison of three routing policies.

Therefore, we can calculate

$$\begin{aligned} \mathrm{NRR} &= \frac{(1-\rho)\rho^{2K}}{1-\rho^{2K+1}}, \\ W &= \frac{N}{\lambda}\frac{1-\rho^{2K+1}}{1-\rho^{2K}}, \end{aligned} \tag{9}$$

where

$$N = \sum_{i=0}^{2K} i \cdot p_i = \begin{cases} \frac{\rho}{1-\rho} - \frac{(2K+1)\rho^{2K+1}}{1-\rho^{2K+1}}, & \rho \neq 1, \\ K, & \rho = 1. \end{cases}$$

Fig. 5 shows the comparison of NRR and W for various values of ρ, where $K = 10$, and $\lambda = 7$. Alternatively, one can plot the performance metrics when the arrival rate (λ) or the service rate (μ) changes. In the queueing literature, however, the traffic intensity ρ determines whether steady-state exists for a system and is often chosen for performance analysis. In addition, choosing ρ allows the flexibility to vary either λ or μ since $\rho = \frac{\lambda}{2\mu}$. It can be seen that, as expected, the shortest-queue policy does better than the random and the round-robin policy, but worse than the single server benchmark under light traffic (around $\rho < 0.7$). However, the performance of these routing policies are quite close to each other when the traffic is light. This implies that, even though the shortest-queue policy is theoretically a better policy, one could still use the random or the round-robin policy since it is easy to implement. When the traffic load is heavy ($\rho > 0.7$), the random and the round-robin policy have a much higher probability of rejecting customers. In this context, the shortest-queue policy is obviously a better policy; it can handle more customers without any serious performance degradation.

4. Distributed load balancing model

The centralized load balancing model has some benefits, such as flexibility and high availability. However, it does not address the network latency issue. For example, user service requests from a distant region have to travel a long distance to the Web server. This creates network congestion and at the same time could result in high latency delays because of the queueing effects on the network. In recent years, content service providers (such as Akamai) are placing Web and/or application servers at network edges and dynamically balance loads among geographically distributed servers. This can improve server selection and load optimization. A general model for the distributed load balancing is depicted in Fig. 6.

4.1. The distributed routing policy

The arrival of user requests at each zone is assumed to follow a Poisson process with an arrival rate of λ. A request that is originated at zone 1 is routed to zone 2 (WS_2) if $(N_1 - N_2) \geqslant \delta$, otherwise it is sent to WS_1.

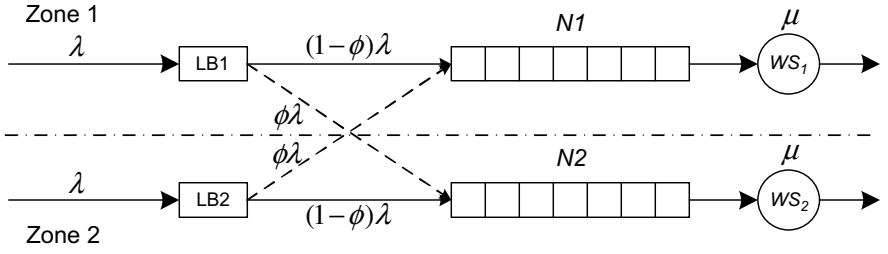

Fig. 6. Distributed load balance model.

Similar policy applies to zone 2. We assume that load balancers know exactly of N_i, the number of customers in each zone. The two Web servers are homogeneous and each has a finite capacity with a buffer size of K. The service time for each server is exponentially distributed with service rate of μ. The queue at each zone is cleared based on an FCFS discipline. The average response time (W) for any given request can be expressed as

$$W = (1-\phi)\left(\frac{1}{\mu} + W_q^{(1)}\right) + \phi\left(\frac{1}{\mu} + W_q^{(2)} + l\right),$$

where $W_q^{(i)}$ is the average waiting time at server WS_i, l is the round-trip latency delay time between the two zones, and ϕ is the probability that a request from one zone is routed to the other zone. Since the two zones are symmetric, $W_q^{(1)} = W_q^{(2)}$. The average response time W simplifies to $W = W_s + \phi l$, where W_s is the average processing time at each zone. Our objective here is to decide a routing policy δ to minimize the average response time

$$\min_{\delta} W = W_s + \phi l,$$

where $\phi = \sum_{i=j+\delta}^{K}\sum_{j=0}^{K} p_{ij}$, $W_s = \frac{N_s}{\lambda(1-\sum_{i=K-\delta+1}^{K} p_{i,K})}$, and $N_s = \sum_{n=0}^{K}[n \cdot \sum_{j=0}^{K} p_{nj}]$.

Notice that there are two special cases for the distributed load balancing model: (i) if $\delta = \infty$, then each zone is an M/M/1 queue with no switching at all, (ii) if $\delta = 1$, then the distributed load balancing model is similar to the shortest-queue policy with total arrival rate of 2λ.

Fig. 7 illustrates a simple example of the state transition diagram when $K = 5$, $\delta = 3$. Using the diagram and equating "flow in" to "flow out" for each state, we can write the steady-state balance equations. Because of symmetry, we only need to consider the lower-right half of the states which have the property $N_1 - N_2 \geqslant 0$. Let $\rho' = \frac{\lambda}{\mu}$ be the server utilization factor at each zone, the balancing equations for the distributed load balancing model can be written as

$$\begin{aligned}
&2\rho' p_{00} = p_{01} + p_{10},\\
&(1+2\rho')p_{i0} = p_{i+1,0} + p_{i,1} + \rho' p_{i-1,0}, \quad 0 < i \leqslant \delta,\\
&(1+2\rho')p_{i0} = p_{i,1} + p_{i+1,0}, \quad \delta < i \leqslant K,\\
&(2+2\rho')p_{ii} = 2\rho' p_{i-1,i} + 2p_{i+1,i}, \quad 0 < i < K,\\
&(2+2\rho')p_{ij} = \rho' p_{i-1,j} + \rho' p_{i,j-1} + p_{i+1,j} + p_{i,j+1}, \quad i = j+m,\ 1 \leqslant m < \delta - 1,\\
&(2+2\rho')p_{ij} = \rho' p_{i-1,j} + 2\rho' p_{i,j-1} + p_{i+1,j} + p_{i,j+1}, \quad i = j+m,\ m = \delta - 1 \text{ or } \delta,\\
&(2+2\rho')p_{ij} = 2\rho' p_{i,j-1} + p_{i,j+1} + p_{i+1,j}, \quad i = j+m,\ \delta < m < K - j,\\
&2p_{KK} = 2\rho' p_{K-1,K}.
\end{aligned} \tag{10}$$

In general, if the buffer size is K, there will be a total of $\frac{(K+1)\cdot(K+2)}{2}$ equations, among which one of them is redundant (i.e. can be obtained from the other equations). Using any $\left[\frac{(K+1)\cdot(K+2)}{2} - 1\right]$ equations in (10), along

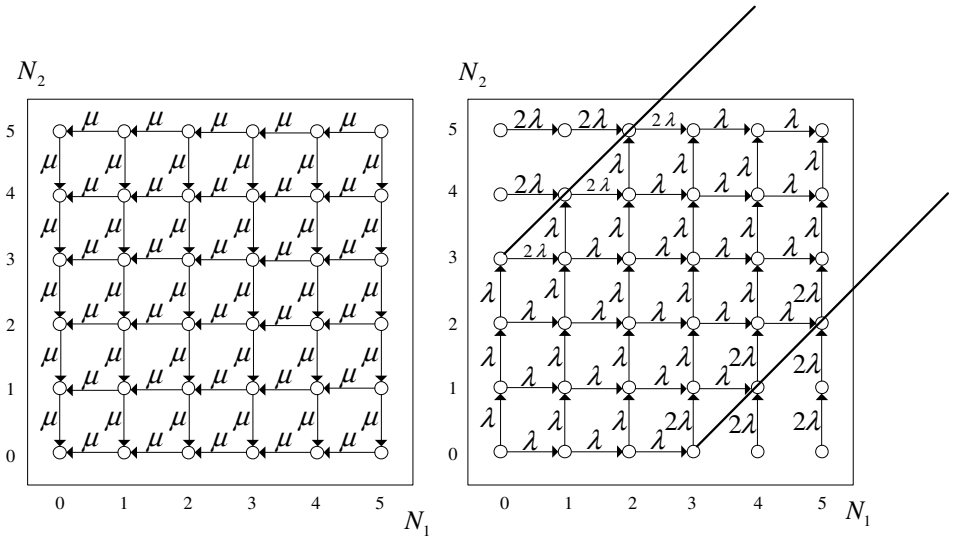

Fig. 7. Schematic of state transition when $K = 5$, $\delta = 3$.

with the unitary boundary condition $\sum_{i=0}^{K} p_{ii} + 2\sum_{i>j}^{K}\sum_{j=0}^{K} p_{ij} = 1$, we can solve for the $\frac{(K+1)\cdot(K+2)}{2}$ steady-state probabilities p_{ij} $(i \geqslant j)$ from the following system of linear equations:

$$AP = B,$$

where A is a $\frac{(K+1)(K+2)}{2} \times \frac{(K+1)(K+2)}{2}$ coefficient matrix, P is a $\frac{(K+1)\cdot(K+2)}{2} \times 1$ column vector denoting probabilities, and $B = (0, 0, \ldots, 0, 1)^{\mathrm{T}}$. Therefore, $P = A^{-1}B$. For instance, when $K = 3$ and $\delta = 2$,

$$\begin{pmatrix} p_{00} \\ p_{10} \\ p_{20} \\ p_{30} \\ p_{11} \\ p_{21} \\ p_{31} \\ p_{22} \\ p_{32} \\ p_{33} \end{pmatrix} = \begin{pmatrix} \rho' & -1 & & & & & & & & \\ -\rho' & 1+2\rho' & -1 & & -1 & & & & & \\ & -\rho' & 1+2\rho' & -1 & & -1 & & & & \\ & & & 1+2\rho' & & & -1 & & & \\ & -2\rho' & & & 2+2\rho' & -2 & & & & \\ & & -2\rho' & & -\rho' & 2+2\rho' & -1 & -1 & & \\ & & & -2\rho' & & -\rho' & 2+2\rho' & & -1 & \\ & & & & & -2\rho' & & 2+2\rho' & -2 & \\ & & & & & & -2\rho' & -\rho' & 2+\rho' & -1 \\ 1 & 2 & 2 & 2 & 1 & 2 & 2 & 1 & 2 & 1 \end{pmatrix}^{-1} \cdot \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

As the size of A grows, it would be cumbersome if not impossible to obtain explicit solutions for p_{ij}. Hence a numerical solution approach is employed (Loan, 2000). Once we obtain p_{ij}'s, we can compute the system performance measures such as $W_s(\delta)$, $W(\delta)$, $\phi(\delta)$. Finally, we iterate δ from 1 to K to find the optimal δ that minimizes the average response time W.

4.2. Numerical results

In this section, we present numerical results for the distributed load balancing model (DLB) and compare it with the single server and centralized load balancing with the shortest-queue routing policy (SQ CLB).

If there is only one single powerful server located in zone 1, then the requests from both zones are routed to that server. Therefore, the total arrival of user requests to the single server follows a Poisson distribution with rate of 2λ. In addition, all requests from zone 2 will experience an additional latency delay of l. So, the average response time $W = \frac{l}{2} + W_1$ where W_1 is given by Eq. (9) with λ being replaced by 2λ and ρ being replaced by

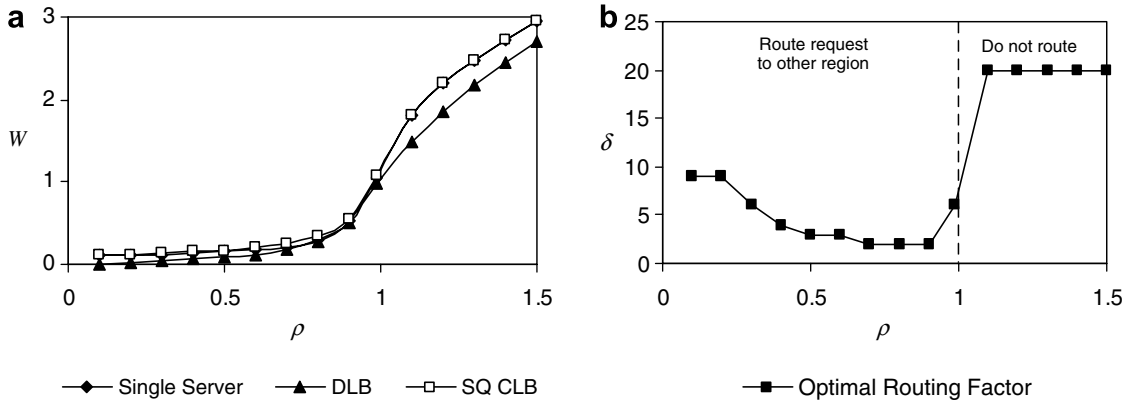

Fig. 8. Comparison of distributed LB, centralized LB and single server model. (a) Average response time W and (b) optimal routing factor δ.

2ρ. If we use the centralized load balancing with the shortest-queue routing policy in zone 1. (Suppose there is no Web server in zone two, all consumers originated in zone 2 have to travel to zone 1, which has two identical servers.) then the average response time $W = \frac{l}{2} + W_2$ where W_2 is given by Eq. (8) with λ being replaced by 2λ and ρ being replaced by 2ρ.

Fig. 8a shows the comparison of W for various values of ρl where $l = 0.2$ s, $K = 20$, and $\lambda = 10$. It can be seen that distributed load balancing performs consistently better than the centralized load balancing and the single server model if the latency (l) is fairly large. (In this case, $l = 0.2$ s.) Fig. 8b shows how the optimal routing factor (δ) changes when the utilization factor (ρ') changes. It is intuitively clear that when both servers are very busy (for example $\rho' > 1$ due to fluctuation of λ), it is better not to route requests to another region. Rather it implies that the server needs to be upgraded or more server added so that the traffic intensity is maintained at a relatively low level. On the other hand, if the load in one zone is heavy and the differences in queue length between the two zones is above a certain level, it might be better off to route some of the requests to the other zone because the latency delay caused by routing may be offset by the faster service in the other zone.

5. Conclusions

This paper analyzes the centralized and distributed load balancing model with two identical servers. The underlying queueing model differs from previous research in that each server has a finite buffer space. We first consider the centralized load balancing with three different routing policies and compare them with the single server benchmark. Our results indicate that the performance of all the routing policies are quite close if the traffic load is light. But when the traffic is heavy, the shortest-queue performs better than other policies; it serves more customers without any serious service degradation.

We then consider situations where multiple Web servers are distributed across two regions. The results show that distributed load balancing may greatly improve the Web server performance especially in the light of continuing network congestion and globalization of business operations. The optimal routing factor δ provides network administrators managerial insight on Web server configuration. It indicates that when the load is heavy and there is a big difference in queue length between the two zones, routing some of the requests may improve the overall system performance. Our routing decision is based on the difference in queue lengths and is easy to implement. The two servers just need to periodically notify each other the load as well as the queue length in each region.

There are several directions for future research. In the paper, we analyzed load balancing among only two Web servers. While one can conceptually extend our techniques to multiple servers, it might be mathematically intractable to obtain analytical results. In that case, one could resort to simulation techniques to obtain a more realistic comparison. Another possible extension is to use processor sharing in place of FCFS queueing

discipline. Finally one can include a cost-benefit analysis among different routing policies, and trade-off the lost sales (due to rejection of customer service) against the better service to find the optimal routing policy.

Appendix

Proof of Eqs. (5)–(7) *in* Section 3.

We want to show that the joint probabilities calculated in Eqs. (5)–(7) satisfy the set of balancing equations in (2). This is equivalent to prove that Eq. (6) satisfies (3) as (5) and (7) are calculated directly from the balancing equations in (2). Let us prove Eq. (6) satisfies (3) by induction on $i \geqslant 1$.

Observe that when $i = K$, (6) reduces to (4). Suppose that the result holds for $i+1$, let us show it also holds for i. Notice that $\binom{i-1}{n-1} + \binom{i-1}{n} = \binom{i}{n}$ and $\rho\zeta^2 - (1+\rho)\zeta + \frac{1}{2} = 0$. Now substitute (6) into the left hand side of (3), we have

$$
\begin{aligned}
&\rho x_i(\delta+1) - (1+\rho)x_i(\delta) + \frac{1}{2}x_i(\delta-1) \\
&= \sum_{n=i}^{K} a_{in}\left[\rho\binom{n-i+\delta}{n-i}\zeta^2 - (1+\rho)\binom{n-i+\delta-1}{n-i}\zeta + \frac{1}{2}\binom{n-i+\delta-2}{n-i}\right]\zeta^{\delta-2} \\
&= \sum_{n=i}^{K} a_{in}\left\{\rho\left[\binom{n-i+\delta-1}{n-i-1} + \binom{n-i+\delta-1}{n-i}\right]\zeta^2 - (1+\rho)\binom{n-i+\delta-1}{n-i}\zeta\right. \\
&\quad \left. + \frac{1}{2}\left[\binom{n-i+\delta-1}{n-i} - \binom{n-i+\delta-2}{n-i-1}\right]\right\}\zeta^{\delta-2} \\
&= \sum_{n=i}^{K} a_{in}\left[\rho\binom{n-i+\delta-1}{n-i-1}\zeta^2 - \frac{1}{2}\binom{n-i+\delta-2}{n-i-1}\right]\zeta^{\delta-2} \\
&= \sum_{n=i+1}^{K} a_{in}\left\{\rho\binom{n-i+\delta-1}{n-i-1}\zeta^2 - \frac{1}{2}\left[\binom{n-i+\delta-1}{n-i-1} - \binom{n-i+\delta-2}{n-i-2}\right]\right\}\zeta^{\delta-2} \\
&= \sum_{n=i+1}^{K} a_{in}\binom{n-i+\delta-1}{n-i-1}\left(\rho\zeta^2 - \frac{1}{2}\right)\zeta^{\delta-2} + \frac{1}{2}\sum_{n=i+2}^{K} a_{in}\binom{n-i+\delta-2}{n-i-2}\zeta^{\delta-2} \\
&= a_{iK}\binom{K-i+\delta-1}{K-i-1}\left(\rho\zeta^2 - \frac{1}{2}\right)\zeta^{\delta-2} + \sum_{n=i+1}^{K-1} a_{in}\binom{n-i+\delta-1}{n-i-1}\left(\rho\zeta^2 - \frac{1}{2}\right)\zeta^{\delta-2} \\
&\quad + \frac{1}{2}\sum_{n=i+2}^{K} a_{in}\binom{n-i+\delta-2}{n-i-2}\zeta^{\delta-2} \\
&= a_{iK}\binom{K-i+\delta-1}{K-i-1}\left(\rho\zeta^2 - \frac{1}{2}\right)\zeta^{\delta-2} + \sum_{n=i+1}^{K-1}\left[a_{in}\left(\rho\zeta^2 - \frac{1}{2}\right) + \frac{1}{2}a_{i,n+1}\right]\binom{n-i+\delta-1}{n-i-1}\zeta^{\delta-2}.
\end{aligned}
$$

From the definition of a_{in} (see Section 3), we have

$$
\begin{aligned}
&-\frac{1}{2}\zeta^2 a_{i+1,K} = a_{iK}\left(\rho\zeta^2 - \frac{1}{2}\right), \\
&a_{in} = \zeta^2\sum_{m=n}^{K}\beta^{m-n+1}a_{i+1,m}, \\
&a_{i,n+1} = \zeta^2\sum_{m=n+1}^{K}\beta^{m-n}a_{i+1,m}.
\end{aligned}
$$

This implies that $a_{in} = \zeta^2\beta a_{i+1,n} + \beta a_{i,n+1}$. So $\frac{a_{in}}{\beta} = \zeta^2 a_{i+1,n} + a_{i,n+1}$ and $(\rho\zeta^2 - \frac{1}{2})a_{in} + \frac{1}{2}a_{i,n+1} = -\frac{1}{2}\zeta^2 a_{i+1,n}$ (because $\beta = \frac{1}{1-2\rho\zeta^2}$). The left hand side of Eq. (6) can then be simplified as

$$= \binom{K-i+\delta-1}{K-i-1}\left(-\frac{1}{2}\zeta^2\right)a_{i+1,K}\zeta^{\delta-2} + \sum_{n=i+1}^{K-1}\left[-\frac{1}{2}\zeta^2 a_{i+1,n}\binom{n-i+\delta-1}{n-i-1}\zeta^{\delta-2}\right]$$

$$= -\frac{1}{2}\sum_{n=i+1}^{K} a_{i+1,n}\zeta^{\delta}\binom{n-i+\delta-1}{n-i-1}.$$

This is exactly the right-hand side of Eq. (3).

References

Aweya, J., Ouellette, M., Montuno, D.Y., Doray, B., Felske, K., 2002. An adaptive load balancing scheme for web servers. International Journal of Network Management 12 (1), 3–39.

Chen, L., Choi, H., 2001. Approximation algorithms for data distribution with load balancing of web servers. In: Proceedings of the IEEE International Conference on Cluster Computing, October, pp. 274—281.

Ciardo, G., Riska, A., Smirni, E., 2001. EquiLoad: A load balancing policy for clustered web servers. Performance Evaluation 46 (2), 101–124.

Colajanni, M., Yu, P.S., 2002. A performance study of robust load sharing strategies for distributed heterogeneous web server systems. IEEE Transactions on Knowledge and Data Engineering 14 (2), 198–414.

Conolly, B.W., 1984. The autostrada queueing problem. Journal of Applied Probability 21 (2), 394–403.

Datta, A., Dutta, K., Thomas, H., VanderMeer, D., 2003. World wide wait: A study of internet scalability and cache-based approaches to alleviate it. Management Science 49 (10), 1425–1444.

Derfler, F.J., Freed, L., 2000. Crash-proof your web site. PC Magazine 19 (7), 135–150.

Down, D.G., Lewis, M.E., 2006. Dynamic load balancing in parallel queueing systems: Stability and optimal control. European Journal of Operational Research 168 (2), 509–519.

Flatto, L., McKean, H.P., 1977. Two queues in parallel. Communications of Pure and Applied Mathematics 30, 255–263.

Gross, D., Harris, C.M., 1998. Fundamentals of Queueing Theory. John Wiley & Sons, New York.

Haight, F.A., 1958. Two queues in parallel. Biometrika 45 (3/4), 401–410.

Halfin, S., 1985. The shortest queue problem. Journal of Applied Probability 22, 865–878.

Karaul, M., Korilis, Y.A., Orda, A., 2000. A market-based architecture for management of geographically dispersed, replicated web servers. Decision Support Systems 28, 191–204.

Kingman, J.F.C., 1961. Two similar queues in parallel. Annals of Mathematical Statistics 32 (4), 1314–1323.

Koenigsberg, E., 1966. On jockeying in queues. Management Science 12 (5), 412–436.

Loan, C.V., 2000. Introduction to Scientific Computing: A Matrix–vector Approaching Using MATLAB. Prentice Hall, Upper Saddle River, NJ.

Menasce, D.A., 2002. Trade-offs in designing web clusters. IEEE Internet Computing 6 (5), 76–80.

Oates, M.J., Corne, D.W., 2001. Global web server load balancing using evolutionary computational techniques. Software Computing 5 (4), 297–312.

Pourbabai, B., Seidmann, A., 1992. Routing and buffer allocation models for a telecommunication system with heterogeneous devices. European Journal of Operational Research 63 (3), 423–431.

Ross, S., 1993. Introduction to Probability Models. Academic Press, Boston.

Sanghi, D., Jalote, P., Agarwal, P., 2002. Using proximity information for load balancing in geographically distributed web server systems. In: Proceedings of 1st Eurasian Conference on Advances in ICT, Tehran, India, October.

Shi, W., MacGregor, M.H., Gburzynski, P., 2005. Load balancing for parallel forwarding. ACM Transactions on Networking 13 (4).

Sparaggis, P.D., Cassandras, C.G., Towsley, D., 1993. On the duality between routing and scheduling systems with finite buffer space. IEEE Transactions on Automatic Control 38 (9), 1440–1446.

Teo, Y.M., Ayani, R., 2001. Comparison of load balancing strategies on cluster-based web servers. The International Journal of the Society for Modeling and Simulation 77 (5–6), 185–195.

Wolf, J.L., Yu, P.S., 2001. On balancing the load in a clustered web farm. ACM Transactions on Internet Technology 1 (2), 231–261.

Zheng, Y.S., Zipkin, P., 1990. A queueing model to analyze the value of centralized inventory information. Operations Research 38 (2), 296–307.