An Approximate Analysis of Load Balancing Using Stale State Information for Servers in Parallel

Jianhua Cao and Christian Nyberg
Email: {jcao, cn}@telecom.lth.se
Dept. of Communication Systems
Lund Institute of Technology
SE-221 00 Lund, Sweden

ABSTRACT

That a load balancing strategy using stale information carelessly will incur system performance degradation is easy to verify. However it is not so obvious that routing a customer to the expected shortest queue has the same problem when information for decision is stale. We consider a queueing system with a load balancer and a pool of identical FCFS queues in parallel. The arrival process is assumed to be Poisson and the service times have identical independent exponential distributions. The pool of servers informs the load balancer the number of customers in each server at some regularly spaced time instances. The load balancer routes each customer to the expected shortest queue based on available stale information and elapsed time since the last time instance of system sate information updating. The system performance analysis of this type of model is usually difficult because the involved state space is very large. However when taking the number of servers to the infinite limit, we have a set of differential equations which is easier to handle than the finite case. Using the approximation of infinite number of severs, we show that the average waiting time for the system is not always minimized by routing each customer to the expected shortest queue when information for decision is stale.

KEY WORDS
Queueing Systems, Load Balancing, Stale Information

1 Introduction

Load Balancing (LB) improves network performance by distributing traffic efficiently so that individual servers are not overwhelmed by sudden fluctuations in activity [3, 13]. In a large Internet web site, balancing incoming requests evenly among many computers is a critical and sometimes tricky task [16, 17]. Despite fruitful theoretical results on both static and dynamic LB strategies[19, 18], current engineering practices, such as web server farm/cluster architecting, is calling for robust LB algorithms that can exploit imperfect system state information effectively.

Routing a customer to the shortest queue when some mild constraint on the service time distribution is satisfied is a conventional wisdom in LB for identical servers in parallel. However sticking to the rule when the assumption of perfect information is broken will cause performance troubles as a previous study reveals [14]. In this paper, a seemly "good" LB algorithm which routes customers to the expected shortest queue is studied. The queueing system under consideration consists a pool of identical servers in parallel, each with its own queue. Customers arrive according to a Poisson process and immediately upon arrival must join one of the queues thereafter to be served on a first-come first-served basis, with no jockeying or defection allowed. The service times have identical independent exponential distributions and are independent of the arrival process and the decisions of customers. The pool of servers announces the state of the system, i.e. the number of customers waiting and being served at each server, at some regularly spaced time instance called state information update instance. When a customer arrives, the load balancer of the system know the elapsed time since the last announcement plus the stale state information for decision. The load balancer routs each customer to the expected shortest queue based on available stale information, which is called the expected shortest queue strategy (ESQS) hereafter. Other well known strategies include random selection and round robin.

The problem of deciding which queue to join when the full and exact state information is available has been studied by many authors. Haight [8] and Kingman [12] considered the dynamics of two servers in parallel when arrivals join the shortest queue. Winston [22] and Weber [20] showed that the shortest queue strategy for N servers in parallel is optimal when the service time distribution has a non-decreasing failure rate and arbitrary arrival. In practice, however, the full and exact state information may be difficult to obtain and maintain, see for example Eager [7] and Zhou [23]. Hjalmtysson and Whitt [11] studied the resource sharing of parallel queues by periodically redistribute customers based on fresh state information. Mitzenmacher [14] considered a system with period state information update similar to ours. He showed that joining the

shortest queue based on stale state information is better than the strategy of random selection when the state information is not too old. Motivated by Mitzenmacher's work, Dahlin [5] proposed an algorithm to exploit the stale state information. His simulation results show that the system employing the algorithm gives shorter waiting times than joining queues at random. Even when the state information is very old, his algorithm will not perform worse than the random strategy.

In the quest for the optimal LB strategy with stale state information, one may first consider the optimality of the strategy that minimizes each customer's expected waiting time. When the distribution of service times is exponential, routing a customer to the expected shortest queue will minimize this customer's expected waiting time. However finding the expected shortest queue is usually very computationally demanding as the available state information is stale. Even if a load balancer has access to unlimited computational resources, the overall system performance is still questionable as the following cases suggest. The fact that decentralized routing based on the expected shortest delay delay may result in poor performance has been known for a long time, see for example Cohen and Kelley [4] and Bersekas [2]. There are several other queueing scenarios where individual optimality does not give a system optimum. Bell and Stidham [1] considered the case that customers only know the service time distribution and the cost of waiting in each server upon arrival to N parallel servers. Whitt [21] found that the average waiting time of the system is not minimized by having each customer minimize his expected waiting time upon arrival to N parallel servers with a certain service time distribution. In this paper numerical examples show that the average queue length and average waiting time of the system is not minimized by routing each customer to the expected shortest queue when the available system state information for load balancing is stale. So this is yet another queueing scenario in which the individual optimum does not coincide with the system optimum.

The rest of this paper is organized as follows. In Section 2 we derive differential equations for the system dynamics. Both the case of finite and infinite number of servers are treated. Section 3 contains the numerical results for the case of infinite number of servers and discussions. Discussions of our result from game theoretical perspective are offered in Section 4. We summarize the paper in Section 5.

2 Dynamics of the System

2.1 System model

We consider a system consisting N FCFS servers in parallel and a load balancer that dispatches incoming customers to the servers, as Fig. 1 shows. The service rates are

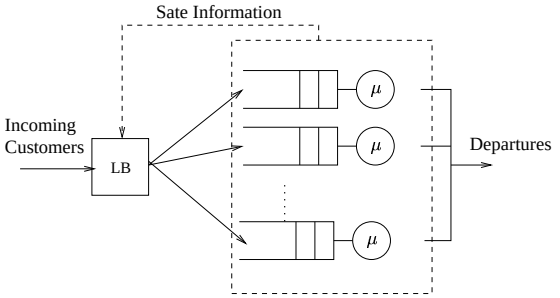


Figure 1. A system consisting of a LB and N identical servers in parallel.

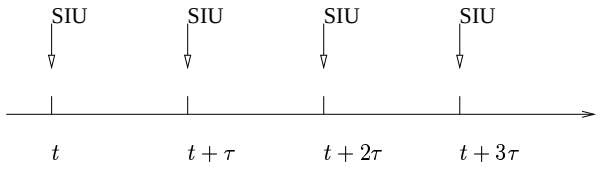


Figure 2. The servers inform the LB their states at regularly space SIU instances: $t, t+\tau, t+2\tau, t+3\tau, \ldots$.

assumed to be identical and exponential distributed. The servers inform the LB their state, i.e. the number of customers in each server, at some regularly space time instance which is called state information update (SIU) instance subsequently, as Fig. 2 shows. The time between two SIU instances is denoted as τ and assumed to be a constant.

We first consider when there are a finite number of servers in the pool. Because the system of differential equations is difficult to solve analytically or numerically especially when the number of server is large, we then consider the limit situation in which the number of servers is infinite. Even though the analytical solution is still difficult to obtain, the numerical result reveals some important proprieties of the system dynamics. The infinite server system approximation seems strange at first since an arrival is guaranteed to find an empty server when the state information is fresh. When the state information is stale, an arrival may join the queue of a busy server due to errors in the predication regardless the number of servers.

2.2 Finite number of servers

Let N be the total number of servers in the server pool; $\mathbf{k} = [k_1, k_2, \ldots, k_N]$ be the vector of the number of customers in each server; $\hat{\mathbf{k}} = [\hat{k}_1, \hat{k}_2, \ldots, \hat{k}_N]$ be the vector of the predicted number of customers in each server. For convenience, let δ_i denote $[\underbrace{0, \ldots, 0, 1}_{i}, \underbrace{0, \ldots, 0}_{N-i}]$. The ar-

rival process is assumed to be Poisson. Let λ be the arrival rate per server to the system. The total arrival rate to the system is $N\lambda$.

Since all customers make decisions using prediction, the arrival rate to a server depends not only on how many customers are predicted in this server but also how many customers are predicted in other servers. When the predicted number of customers is distributed as $\hat{\mathbf{k}}$, the arrival rate to the server i at time t is given by,

$$\phi^{(i)}_{\hat{\mathbf{k}}}(t) = \begin{cases} N\lambda/n & \text{if } i \in A \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $A = \{i : k_i = \min(\hat{k}_1, \hat{k}_2, \ldots, \hat{k}_N)\}$ is the set of servers having less customers than the others in the pool and n is the number of elements in the set A. When a few servers are being predicted to have the same number of customers and to have less customers than any other in the pool, the total arrival rate $N\lambda$ is divided into n portions. Each server that is predicted to have the smallest number of customers receives a portion of $N\lambda/n$.

The state of the system is characterized by $\mathbf{k}$ and $\hat{\mathbf{k}}$ jointly because $\mathbf{k}$ and $\hat{\mathbf{k}}$ may differ except at the moment when SIU happens. Let $p_{\mathbf{k},\hat{\mathbf{k}}}(t)$ be the probability that the system is at the state $(\mathbf{k}, \hat{\mathbf{k}})$. When the system is initially empty,

$$p_{\mathbf{k},\hat{\mathbf{k}}}(0) = \begin{cases} 1 & \text{if } \mathbf{k} = \hat{\mathbf{k}} = 0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

For brevity, we only consider states that are not near the boundary, i.e. when $k_i \geq 1$ and $\hat{k}_i \geq 1$, $k = 1, \ldots, N$ in the following discussion.

Conditioned on the current state $(\mathbf{k}, \hat{\mathbf{k}})$, the probability that the state is unchanged in a small time interval Δt is

$$1 - 2\sum_{i=1}^{N} \phi^{(i)}_{\hat{\mathbf{k}}}(t)\Delta t - 2N\mu\Delta t + o(\Delta t).$$

Conditioned on the current state $(\mathbf{k} - \delta_i, \hat{\mathbf{k}})$, the probability that the state changes to $(\mathbf{k}, \hat{\mathbf{k}})$ in a small time interval Δt is

$$\phi^{(i)}_{\hat{\mathbf{k}}}(t)\Delta t + o(\Delta t).$$

Conditioned on the current state $(\mathbf{k}, \hat{\mathbf{k}} - \delta_i)$, the probability that the state changes to $(\mathbf{k}, \hat{\mathbf{k}})$ in a small time interval Δt is

$$\phi^{(i)}_{\hat{\mathbf{k}}-\delta_i}(t)\Delta t + o(\Delta t).$$

Conditioned on the current state $(\mathbf{k} + \delta_i, \hat{\mathbf{k}})$ or $(\mathbf{k}, \hat{\mathbf{k}} + \delta_i)$, the probability that the state changes to $(\mathbf{k}, \hat{\mathbf{k}})$ in a small time interval Δt is

$$\mu\Delta t + o(\Delta t).$$

Conditioned on all other current state, the probability that the state changes to $(\mathbf{k}, \hat{\mathbf{k}})$ in a small time interval Δt is $o(\Delta t)$.

Thus the system evolves from time t to $t + \Delta t$ according to the following rule,

$$\begin{aligned} p_{\mathbf{k},\hat{\mathbf{k}}}(t + \Delta t) =& p_{\mathbf{k},\hat{\mathbf{k}}}(t)\left(1 - 2\sum_{i=1}^{N} \phi^{(i)}_{\hat{\mathbf{k}}}(t)\Delta t - 2N\mu\Delta t\right) \\ &+ \sum_{i=1}^{N} p_{\mathbf{k}-\delta_i,\hat{\mathbf{k}}}(t)\phi^{(i)}_{\hat{\mathbf{k}}}(t)\Delta t \\ &+ \sum_{i=1}^{N} p_{\mathbf{k},\hat{\mathbf{k}}-\delta_i}(t)\phi^{(i)}_{\hat{\mathbf{k}}-\delta_i}(t)\Delta t \\ &+ \sum_{i=1}^{N} p_{\mathbf{k}+\delta_i,\hat{\mathbf{k}}}(t)\mu\Delta t \\ &+ \sum_{i=1}^{N} p_{\mathbf{k},\hat{\mathbf{k}}+\delta_i}(t)\mu\Delta t + o(\Delta t). \end{aligned} \tag{3}$$

Moving $p_{\mathbf{k},\hat{\mathbf{k}}}(t)$ from the right side to the left, dividing both sides by Δt and letting $\Delta t \to 0$, we have the following differential equation,

$$\begin{aligned} \frac{d}{dt}p_{\mathbf{k},\hat{\mathbf{k}}}(t) =& -2p_{\mathbf{k},\hat{\mathbf{k}}}(t)\left(\sum_{i=1}^{N} \phi^{(i)}_{\hat{\mathbf{k}}}(t) + N\mu\right) \\ &+ \sum_{i=1}^{N} p_{\mathbf{k}-\delta_i,\hat{\mathbf{k}}}(t)\phi^{(i)}_{\hat{\mathbf{k}}}(t) \\ &+ \sum_{i=1}^{N} p_{\mathbf{k},\hat{\mathbf{k}}-\delta_i}(t)\phi^{(i)}_{\hat{\mathbf{k}}-\delta_i}(t) \\ &+ \sum_{i=1}^{N} p_{\mathbf{k}+\delta_i,\hat{\mathbf{k}}}(t)\mu + \sum_{i=1}^{N} p_{\mathbf{k},\hat{\mathbf{k}}+\delta_i}(t)\mu. \end{aligned} \tag{4}$$

At the SIU instance t, an arrival will have the full and the exact information about how many customers are in each queue, therefore,

$$p_{\mathbf{k},\hat{\mathbf{k}}}(t) = \begin{cases} \lim_{s\uparrow t} \sum_{\mathbf{h}} p_{\mathbf{k},\mathbf{h}}(s) & \text{if } \hat{\mathbf{k}} = \mathbf{k} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

The system of differential equation is nonlinear because $p_{\mathbf{k},\hat{\mathbf{k}}}$ depends on the state dependent arrival rate. Like most nonlinear differential equations, a closed analytical solution is unlikely to be found. Let us investigate the prospect of a numerical solution. When the number of servers in the pool is large, we have a very large state space. Assume that the system of differential equations is truncated so the maximal number of customers per server is $k_{\max}$. When we have N servers, the total number of states is $(k_{\max} + 1)^{2N}$. The number of states increases exponentially as N increases. Therefore (4) is numerically intractable when N is large.

However when we let $N \to \infty$,we can actually have a set of differential equations that are simpler but yet powerful enough to reveal the dynamics of the system.

2.3 Approximated analysis using infinite number of servers

Let $p_{i,j}(t)$ be the percentage of servers having i customers but being predicted to have j customers at time t. Though we do not explicitly model how many customers, real and predicted, in each queue, knowing $p_{i,j}(t)$ is enough to derive the average number of customers per server in the system.

Let $\psi_j(t)$ be the arrival rate to servers that are predicted to have j customers each. Since new arrivals are routed to servers that are predicted having 0 customers, $\psi_0(t)$ is the ratio between the average arrival rate per server λ and the percentage of servers that are predicted to have 0 customers. Thus,

$$\psi_j(t) = \begin{cases} \lambda / \sum_k p_{k,0}(t) & \text{if } j = 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

When the system is initially empty,

$$p_{i,j}(0) = \begin{cases} 1 & \text{if } i = 0 \text{ and } j = 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

The system evolves from time t to time $t + \Delta t$ according to the following rules:

when $i = 0$ and $j = 0$,

$$\begin{aligned} p_{0,0}(t + \Delta t) =& p_{0,0}(t)\,(1 - 2\psi_0(t)\Delta t) \\ &+ p_{1,0}(t)\mu\Delta t + p_{0,1}(t)\mu\Delta t + o(\Delta t); \end{aligned} \tag{8}$$

when $i = 0$ and $j \geq 1$,

$$\begin{aligned} p_{0,j}(t + \Delta t) =& p_{0,j}(t)\,(1 - 2\psi_j(t)\Delta t - \mu\Delta t) \\ &+ p_{0,j-1}(t)\psi_{j-1}(t)\Delta t \\ &+ p_{1,j}(t)\mu\Delta t + p_{0,j+1}(t)\mu\Delta t + o(\Delta t); \end{aligned} \tag{9}$$

when $i \geq 1$ and $j = 0$,

$$\begin{aligned} p_{i,0}(t + \Delta t) =& p_{i,0}(t)\,(1 - 2\psi_0(t)\Delta t - \mu\Delta t) \\ &+ p_{i-1,0}(t)\psi_0(t)\Delta t \\ &+ p_{i+1,0}(t)\mu\Delta t + p_{i,1}(t)\mu\Delta t + o(\Delta t); \end{aligned} \tag{10}$$

when $i \geq 1$ and $j \geq 1$,

$$\begin{aligned} p_{i,j}(t + \Delta t) =& p_{i,j}(t)\,(1 - 2\psi_j(t)\Delta t - 2\mu\Delta t) \\ &+ p_{i-1,j}(t)\psi_j(t)\Delta t + p_{i,j-1}(t)\psi_{j-1}(t)\Delta t \\ &+ p_{i+1,j}(t)\mu\Delta t + p_{i,j+1}(t)\mu\Delta t + o(\Delta t). \end{aligned} \tag{11}$$

Moving $p_{0,0}(t)$, $p_{0,j}(t)$, $p_{i,0}(t)$ and $p_{i,j}(t)$ from the right sides of (8),(9), (10) and (11) to the left sides, dividing both sides by Δt and letting $\Delta t \to 0$, we have the following system of differential equations, when $i = 0$ and $j = 0$,

$$\frac{d}{dt}p_{0,0}(t) = -2p_{0,0}(t)\psi_0(t) + p_{1,0}(t)\mu + p_{0,1}(t)\mu; \tag{12}$$

when $i = 0$ and $j = 1$,

$$\begin{aligned} \frac{d}{dt}p_{0,j}(t) =& -p_{0,j}(t)(2\psi_j(t) + \mu) \\ &+ p_{0,j-1}(t)\psi_{j-1}(t) + p_{1,j}(t)\mu + p_{0,j+1}(t)\mu; \end{aligned} \tag{13}$$

when $i = 1$ and $j = 0$,

$$\begin{aligned} \frac{d}{dt}p_{i,0}(t) =& -p_{i,0}(t)(2\psi_0(t) + \mu) \\ &+ p_{i-1,0}(t)\psi_0(t) + p_{i+1,0}(t)\mu + p_{i,1}(t)\mu; \end{aligned} \tag{14}$$

when $i \geq 1$ and $j \geq 1$,

$$\begin{aligned} \frac{d}{dt}p_{i,j}(t) =& -2p_{i,j}(t)(\psi_j(t) + \mu) \\ &+ p_{i-1,j}(t)\psi_j(t) + p_{i,j-1}(t)\psi_{j-1}(t) \\ &+ p_{i+1,j}(t)\mu + p_{i,j+1}(t)\mu. \end{aligned} \tag{15}$$

At the SIU instance t, an arrival knows the percentage of servers having i customers in the system. So,

$$p_{i,j}(t) = \begin{cases} \lim_{s \uparrow t} \sum_k p_{i,k}(s) & \text{if } j = i \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

The system of differential equations for the case of an infinite number of servers is still nonlinear and analytical intractable. But now it is solvable by numerical methods.

3 Numerical Results

We first truncate the system of differential equations (12), (13), (14) and (15) to a finite number of i:s and j:s, i.e. $0 \leq i \leq i_{\max} = 100$ and $0 \leq j \leq j_{\max} = 100$. The truncated differential equations are then solved numerically using ODEPACK [10]. For convenience, we assume the service rate of each server $\mu = 1$ in the following discussion. For stability reasons, both for the system itself and

the numerical solutions, we choose to show the result when $\lambda = 0.5$.

We are interested in the average number of customers per server $\bar{n}(t)$ which can be calculated as follows once the distribution of the system states $[p_{i,j}(t)]$ is known,

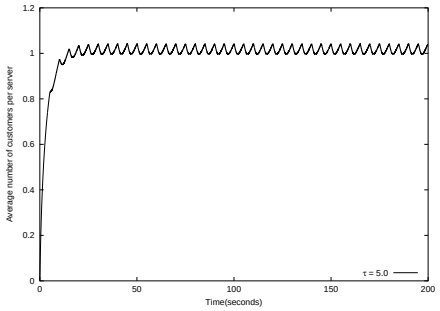
$$\bar{n}(t) = \sum_{i=0}^{i_{\max}} i \sum_{j=0}^{j_{\max}} p_{i,j}(t) \tag{17}$$

Fig. 3 shows the transient behavior of $\bar{n}(t)$ for different SIU intervals τ when the system is initially empty. From Fig. 3, we see that $\bar{n}(t)$ oscillates. The period of oscillation is equal to the period of the SIU updates. The amplitude of oscillation increases as SIU interval increases.

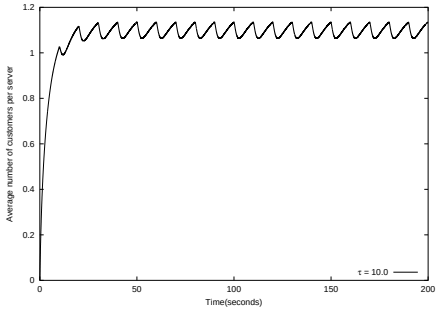
Fig. 4 shows the transient behavior of $\bar{n}(t)$ between two SIU updates for different SIU intervals τ when the influence of the initial state can be neglected. We scale different SIU intervals to 1 in order to reveal the trend in the curves. From Fig. 4 we see that just before each SIU update instance $\bar{n}(t)$ reaches maximum, and $\bar{n}(t)$ starts to drop right after each SIU update instance. Fig. 4 also shows that for the fixed arrival rate $\lambda = 0.5$, the maximum of $\bar{n}(t)$ increases, but not indefinitely, as the SIU update interval increases.

When the SIU interval is large, the average number of customers in the system is not minimized by routing each customer to the expected shortest queue. In Fig. 4(b) the minimum of the average queue length per server $\bar{n}(t)$ between two SIU instances, which is about 1.11 , is greater than 1.0 which can be obtained if random selection strategy is used. Hence, by Little's law, the average waiting time for the system is not minimized by routing each customer to the expected shortest queue.

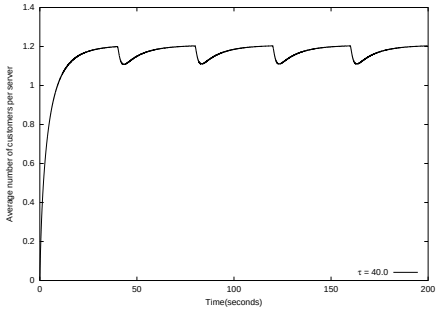
Fig. 5 shows how the maximum of $\bar{n}(t)$ increases with arrival rate λ when the SIU interval τ is fixed to 50 seconds. In the same figure, we also plot the average number of customers per server in steady state for the random selection strategy [1]. For the ESQS, the maximum of $\bar{n}(t)$ increases faster and is always greater than the average number of customers per server for the random selection. Therefore it suggests that the system using the ESQS becomes unstable earlier than the system using the random strategy when the arrival rate per server λ is close to 1 or the utilization of servers is close to 1.



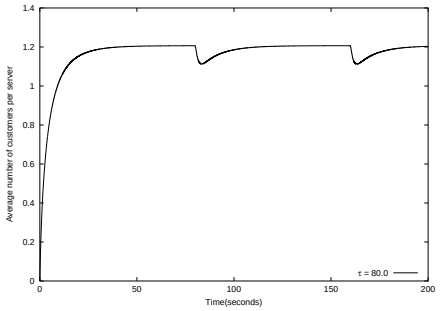
(a) $\tau = 5.0$



(b) $\tau = 10.0$



(c) $\tau = 40.0$



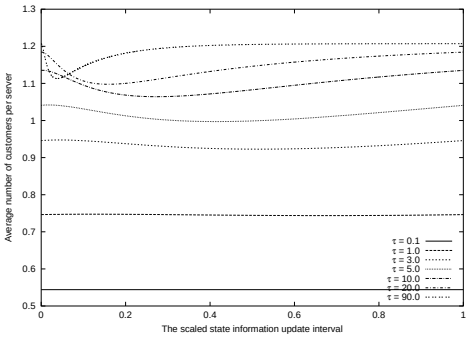
(d) $\tau = 90.0$

Figure 3. Transient behavior of the average number of customers per server between time 0 second and 200 seconds for different SIU intervals τ when the system is initially empty.

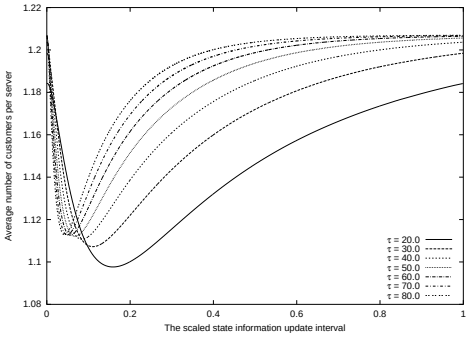
4 Discussions

The system under consideration can also be viewed from another perspective. Let us call the system described in

[1] When the random selection is used, the arrival process to each server is Poisson. Therefore the average number of customers per server $\bar{n} = \lambda/(1-\lambda)$, where λ is the arrival rate per server.



(a)



(b)

Figure 4. Transient behavior of the average number of customers per server between two SIU instance for different SIU intervals τ when the influence of the initial state can be neglected. We scales different SIU intervals to 1 in order to reveal the trend in curves.

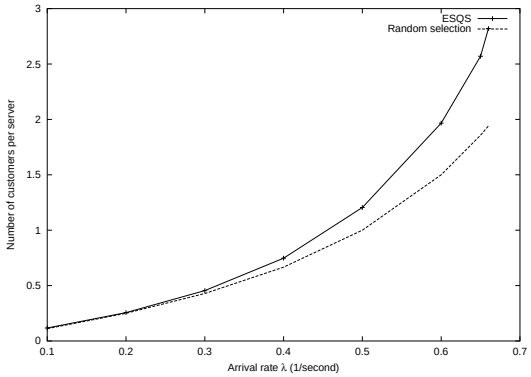


Figure 5. The maximum of the average number of customers per server for the ESQS vs. arrival rate per server when the influence of initial state can be neglected. We also plot the average number of customers per server in steady state when the random selection strategy is used.

Section 2.1 system A. Consider a similar system B. In system B, the load balancer is removed but each arrival decides which queue to join based on stale state information and the elapsed time since last SIU update instance. Now we have a queueing game that will be played by all customers. Every customer tries to minimize his own waiting time. Clearly the optimal decision of each customer also depends on all early arrivals. When almost everyone joins the expected shortest queue based on stale information, no one will benefit a shorter waiting time by deviating the common strategy. In game theory literatures [9, 6, 15], such a situation is refereed as Wardrop equilibrium (for infinite numer of players) or Nash equilibrium (for finite number of players). By definition, system B in Wardrop equilibrium is equivalent to System A.

From game theory point of view, that minimizing the expected waiting time of each customer is not equivalent to minimizing the average waiting time of the system is clear. However when there are finite numer of servers in our system, this fact is difficult to verify through equations that governs the system dynamics. The approximation using infinite number of servers turns out be effective to reveal that in our system Wardop equilibrium does not bring the system optimal solution.

5 Conclusions

This article studies a LB strategy of routing an arrival to the expected shortest queue among a pool of identical servers in parallel when the available state information is stale. When the number of servers is finite, we derive differential equations that reflect the system dynamics. Because of the analytical and numerical intractability of these differential equations, we consider the limiting situation in which the number of servers is infinite. It turns out that the set of differential equations for an infinite number of servers can be solved numerically. The numerical solution presents some expected behaviors of the system dynamics such as the oscillation and some unexpected behaviors such as the extreme of the average number of customers per server increases as the SIU interval grows. The numerical solution also shows that LB through routing each customer to the expected shortest queue is not always a good strategy in order to achieve the minimum of average waiting time when the available information for decision is stale. However for the system under consideration the optimum load balancing strategy that minimizes the average waiting time is still unresolved and is a subject of future work.

References

[1] Colin E. Bell and Shaler Stidham Jr. Individual versus social optimization in the allocation of customers to alternative servers. *Management Science*, 29(7):831–839, July 1983.

[2] Dimitri P. Bertsekas. Optimal routing and flow control methods for communication networks. In A. Bensoussan and J. L. Lions, editors, *Analysis and Optimization of Systems*, volume 44 of *Lecture Notes in Control and Information Science*, pages 615–643. Springer-Verlag, 1982.

[3] Tony Bourke. *Server Load Balancing*. O'Reilly & Associates, 2001. 2nd edition.

[4] Joel E. Cohen and Frank P. Kelly. A paradox of congestion in a queuing network. *Journal of Applied Proabbility*, 27:730–734, 1990.

[5] Michael Dahlin. Interpreting stale load information. *IEEE Transactions on Parallel and Distributed System*, 11(10):1033–1047, October 2001.

[6] Pradeep Dubey. Inefficiency of nash equilibria. *Mathematics of Operation Research*, 11:1–8, 1986.

[7] Derek L. Eager, Edward D. Lazowska, and John Zahorjan. Adaptive load sharing in homogeneous distributed systems. *IEEE Transactions on Software Engineering*, SE-12(5):662–675, May 1986.

[8] Frank A. Haight. Two queues in parallel. *Biometrika*, 45(3/4):401–410, December 1958.

[9] Refael Hassin and Moshe Haviv. *To Queue or Not to Queue: Equilibrium Behavior in Queueing Systems*. Kluwer Academic Publishers, 2003.

[10] Alan C. Hindmarsh. *Scientific Computing*, chapter ODEPACK, A Systematized Collection of ODE Solvers. North-Holland, Amsterdam, 1983.

[11] Gisli Hjalmtysson and Ward Whitt. Periodic load balancing. *Queueing Systems*, 30:203–250, 1998.

[12] John F. C. Kingman. Two similar queues in parallel. *Annals of Mathematical Statistics*, 32(4):1314–1323, December 1961.

[13] Chandra Kopparapu. *Load Balancing Servers, Firewalls, * Caches*. John Wiley & Sons, 2002.

[14] Michael Mitzenmacher. How useful is old information? *IEEE Transactions On Parallel and Distributed Systems*, 11(1):6–20, January 2000.

[15] Roger B. Myerson. *Game Theory, Analysis of Conflicit*. Harvard University Press, 1991.

[16] IBM Redbooks. *Load-Balancing Internet Servers*. IBM Corp., 1998.

[17] IBM Redbooks. *Load Balancing for eNetwork Communications Servers*. IBM Corp, 1999.

[18] Keith W. Ross and David D. Yao. Optimal load balancing and scheduling in a distributed computer system. *Journal of the ACM*, 38(3):676–690, July 1991.

[19] Asser N. Tantawi and Don Towsley. Optimal static load balancing in distributed computer systems. *Journal of the ACM*, 32(2):445–465, April 1985.

[20] Richard R. Weber. On the optimal assignment of customers to parallel servers. *Journal of Applied Probability*, 15:406–413, 1978.

[21] Ward Whitt. Deciding which queue to join: Some counterexamples. *Operations Research*, 34(1):55–62, January–February 1986.

[22] Wayne Winston. Optimality of the shortest line discipline. *Journal of Applied Probability*, 14:181–189, 1977.

[23] Songnian Zhou. A trace-driven simulation study of dynamic load balancing. *IEEE Transactions On Software Engineering*, 14(9):1327–1341, September 1988.