Optimization of the Dispatch Probability of Service Requests for Two Servers in Parallel Connection

CHUNG-PING CHEN[*†], YING-WEN BAI[§], HSIANG-HSIU PENG[§]
[*]Graduate Institute of Applied Science and Engineering, Fu Jen Catholic University
[†]Department of Electronic Engineering, National Taipei University of Technology
[§]Department of Electrical Engineering, Fu Jen Catholic University
510 Chung Cheng Rd. Hsinchuang, New Taipei City 24205
TAIWAN, R.O.C.
491598038@mail.fju.edu.tw, bai@ee.fju.edu.tw, 499216210@mail.fju.edu.tw

Abstract: - This paper describes the procedure used to create a queueing model for servers which are currently in parallel connection with different dispatch probabilities. The system response time of parallel Web servers is improved by calculating dispatch probabilities. The system response time is calculated first. The equation for the lowest system response time is obtained by adjusting the equation to meet the dispatch probability requirements. Then the system response time of the servers is measured. Last, the errors in the system response time of the servers are compared by simulating, modeling and measurement.

Key-Words: - Service rate, Optimization of the Dispatch probability, Parallel connection, System response time, Queueing Model, Different Dispatch Probabilities, Equivalent model.

1 Introduction

To simplify the analysis of network performance, packet flow and congestion control, many people use an equivalent queueing model which can deduce the equivalent equation and represent the system response time of the whole network. We classify basic network structures with serial and parallel connections and analyse the error between a queueing model and its measurement counterpart [1]. When several servers are in serial connection, the overall system response time is in proportion to the number of servers [2]. When they have a low CPU load, the system response time shows very few variations [3]. In addition, we aim at two servers both in parallel connection and with different dispatch probabilities. From the measurement results we are able to discern any error in the system response time by means of both measurements and our analysis [4].

This paper is organized as following. In Section 2 was literature review. In Section 3 an M/M/1 queue was used to represent the server system in parallel connection and to simplify the equivalent equations for two servers in parallel connection and with different dispatch probabilities. In Section 4 the queuing network simulation tool was used to simulate the equivalent queues of the aforementioned meanings. In Section 5 the servers were physically install and carried out a performance measurement according to the system setting and make a comparison with simulation results. In Section 6 the conclusion is given and the possible future research projects are previewed.

2 Literature Review

As a network becomes large, the gap in its system response time generates many unpredictable results. To minimize this difference there are many available methods and software algorithms which have been developed to manage multiple Web servers to reach a so-called load balance [5].The DHT (distributed hash table)-based P2P system has been proposed to solve load balancing problems, but one must consider the heterogeneous nature of this system. Zhu presents an efficient proximity-aware load balancing scheme by using the concept of virtual servers. Higher to capacity nodes are employed to carry more loads, thereby with minimizing the load movement cost and allowing efficient load balancing [6]. Chen not only proposes the virtual-server-based load balancing method by systematically using an optimization-based approach but also derives an effective algorithm to rearrange loads among peers. The effect of heterogeneity on the performance of a load balancing algorithm is characterized systematically [7]. Zhang focuses on the load balancing policies established for homogeneously clustered Web servers that tune their parameters on-the-fly to adapt

themselves to both the changes in the arrival rates and the service time of incoming requests [8]. Guo designs, implements and evaluates three scheduling algorithms, namely first fit (FF), stream-based mapping (SM) and adaptive load sharing (ALS) for multimedia transcoding in a cluster environment. He proposes an online prediction algorithm and two new load scheduling algorithms: prediction-based least load first (P-LLF) and prediction-based adaptive partitioning (P-AP). The performance of the system is evaluated in terms of system throughput, the out-of-order rate of outgoing media streams and the load balancing overhead through real measurements using a cluster of computers [9]. Chen shows that the degree of inefficiency in load balancing designs is highly dependent on the scheduling disciplines used by each of the back-end servers which are changed to the shortest remaining processing time (SRPT); thus the degree of inefficiency can be independent of the number of servers. Switching the back-end scheduler to SRPT can provide significant improvements of the overall mean response time of the system as long as the heterogeneity of the server speeds is small [10]. In the RFID-Grid proxy architecture, Chang uses a probabilistic cost estimation model of an M/M/m queuing system to schedule RFID jobs with the system load balance, with both the real-time throughput and deployment cost taken into account [11]. To attain such a balance, different hardware connections which influence the system response time need to be studied.

Oshima presents a performance evaluation for a Linux platform under SYN flooding attacks. The SYN flooding attack is a DoS (Denial of Service) method which compels the hosts to retain their half-open state and results in an exhaustion of their memory resources. He implements an attacking and observing program and observes the response packets from the server and the performance of the system under the DoS attacks [12]. The aims of Nakashima's research are to expand the observation of queueing behavior, to explore managing queueing activity for self-similar traffic using the network simulator under different conditions and to clarify the causes of those different features that focus on the bottleneck link. These generate two different types of traffic, "queue-buffering with no full use of bandwidth" and "queue-buffering with full use of bandwidth" [13,14]. Ma offers a queueing-theory-based method and some mathematical models for analyzing the performance of automated storage/retrieval systems (AS/RS) under stochastic demands. His method models the optimized behavior of the AS/RS, establishes a queuing model to determine whether or not it meets the throughput and offers equations to compute the utilization of the AS/RS, the fraction of DC cycles and the mean storage/retrieval time [15].

Some proposals have improved the performance of network equipment. The InterAsterisk eXchange (IAX) and the Real time SWitching (RSW) pose considerable problems for users who have to choose between two solutions offering different advantages and disadvantages [16]. Li investigates the synchronization stability for discrete-time stochastic complex dynamical networks with probabilistic interval-time-varying delays in the network coupling and the dynamical nodes. The solvability of derived conditions depends not only on the size of the delay, but also on the probability of the delay sampled at some intervals [17]. The applications in wireless multimedia sensor networks (WMSNs) must focus on energy saving and the application-level quality of service (QoS). To achieve these goals, Tseng has proposed a new pattern named temporal region requesting pattern (TRRP) and a novel algorithm named TRRP-Mine to mine TRRPs efficiently [18]. Active Queue Management (AQM) routers have been proposed to support the end-to-end congestion control in the Internet. A fuzzy modeling technique is employed to set up a time-delay affined to be the Takagi-Sugeno (T-S) fuzzy model for a Transmission Control Protocol (TCP) network with AQM routers [19].

Some proposals have been made to improve network utilization by using the parallel method. With the improvements in wide-area network performance and powerful computers, it is possible to integrate a large number of distributed machines which belong to different portions of a single system. If a task is processed in parallel in a machine, intra-communication overhead is inevitable and is an important factor affecting the processing time of the task. Thus, instead of a free communication assumption, Lin takes the intra-communication overhead into account [20]. In the Internet networks, packets can be transmitted without loss provided that the network load remains sufficiently low. However, practical networks frequently experience a congestion problem. Chen proposes a generic method for optimizing the end-to-end delay of a single TCP (transmission control protocol) virtual path in the Internet environment [21]. Ku etc. address the problem of improving end-to-end performance while exchanging packets through the Internet by a better management of queuing behavior of network nodes. They present a new network queue control algorithm which may deliver

better end-to-end performance than others at the cost of reduced link utilization at the node [22].

The FTMP (Fault-Tolerant Multiprocessor) is a digital computer architecture which could have varying applications in several life-critical aerospace situations. A dispatch probability model is also presented. Experience with an experimental emulation is described [23]. Solving the probabilistic optimal dispatching problem using fixed steady probability and a full contingency-set as well as an effective contingency-set according to the transmission element outage distribution factors is established [24].

In nuclear fusion experiments, Barana describes the architecture of a general-purpose Java-based tool for action dispatch. The action dispatcher tool must comply with several requirements such as the support for a distributed and heterogeneous environment, a comprehensive user interface for the supervision of the whole sequence and the need for Web-based support [25].

Some papers which improve the network utilization using the dispatcher method are proposed. Power network faults may lead to blackouts in some areas. Dispatchers should accurately diagnose faulted elements as soon as possible and restore those areas to their normal state. Siqing provides the limitation of the conventional expert system (ES) based on the rules for restoring a blacked-out area; moreover, a new random adaptive optimizing method based on a genetic algorithm (GA) is applied to the restoration of the system of the distribution network [26]. Swarup provides a short-term load-forecasting (STLF) program that uses an integrated artificial neural network (ANN) approach which is capable of predicting loads for basic generation scheduling functions, assessing power system security and providing timely dispatcher information [27]. Yan proposes the topological characteristics of scale-free networks and discrete particle swarm optimization, which is, therefore, a skeleton-network reconfiguration strategy [28].

Huang has designed a novel Multi-Core-based Parallel Streaming Mechanism (MCPSM) for concurrently streaming the scalable extension of H.264/AVC videos with the concept both of CPU-scheduling-SJF (Shortest-Job-First) and of multi-core-based parallel programming. Multiple streaming tasks can be executed in parallel using a proposed task dispatcher based on network connection speeds, video bit-rate and user's waiting time [29]. Ohlendorf has presented a solution for a classification problem that is used for optimized packet assignment to different data paths within a network processor system-on-chip (SoC). This solution is concluded with the results of an implementation on our field-programmable gate-array (FPGA)-based prototyping platform [30]. In wireless location privacy protection, computing and communication technologies have finally converged. Today, when a person reports an emergency from a landline phone by dialing 911 in the United States or 112 in Europe, the system displays both the caller's phone number and address to the dispatcher. Another paper focuses on some of the areas affected by this capability, such as privacy risk, economic damage, location-based spam, intermittent connectivity, user interfaces, network privacy and privacy protection [31].

About optimizing aspects, find the following few papers. Orthogonal Frequency Division Multiplexing (OFDM) is a suitable multicarrier modulation scheme for high speed broadband communication systems. Palanivelan proposes an efficient Reduced Complexity Max Norm (RCMN) algorithm for optimizing the PAPR of the OFDM signals [32]. Hua Hou proposes 4 new schemes of cross-layer packet dependent OFDM scheduling for heterogeneous classes of traffic based on proportional fairness [33]. A computationally efficient global optimization method, the adaptive particle swarm optimization (APSO), is proposed for the synthesis of uniform and Gaussian amplitude arrays of two cases [34]. Milan Tuba present a modified algorithm which integrates artificial bee colony (ABC) algorithm with adaptive guidance adjusted for engineering optimization problems [35]. Manjusha Pandey proposed privacy provisioning scheme aim to optimize energy consumption for privacy provisioning in WSN [36]. Vincent Roberge presented a parallel implementation of the Particle Swarm Optimization (PSO) on GPU using CUDA [37].

3 The Queueing Model for Servers in Parallel Connection and of Different Dispatch Probabilities

We use an equivalent model to represent the servers in parallel connection. We find the system response time of the equivalent model and verify it by measuring the system response time. Table 1 shows the system definition and the model parameters. We choose 1 ms as the measurement unit for the system response time.

Table 1 System Parameters and Units of the Model

Parameter	Description	Definition
λ	Web request arrival rate	Requests/sec
μ_m	Service rate of the m^{th} server	Requests/sec
D_m	Dispatch probability of the m^{th} parallel queue	None
ρ_m	λ / μ_m	None
n	μ_1 / μ_2	None
$E_{D_m}(T)$	System response time of the m^{th} parallel queue	msec
$E_{D_{eq}}(T)$	Equivalent system response time of the servers in parallel connection	msec
$E_{Peq_n}(T)$	Equivalent system response time of the n^{th} parallel connection servers	msec

We use the idea of an equivalent model for an electric circuit in parallel connection as our analytical basis and verify the circuit's performance experimentally. In this paper we use approximate equations for the different dispatch probabilities of the servers in parallel connection. The model representation is shown in Fig.1.

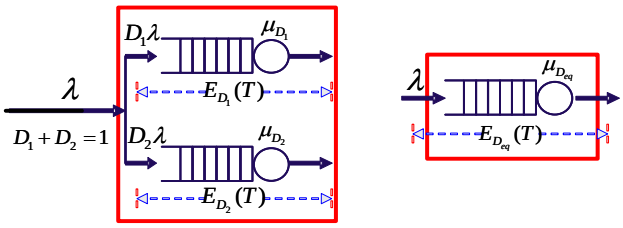

Fig. 1 Equivalent model of a parallel queue

At the beginning we use queueing models and Markov Chains [4] to analyze the servers in parallel connection with the following assumptions:

- All requests in the system are on a first in first out basis.
- The total of the requests in the system is unlimited.
- Each node in the system can have Poisson arrivals from an outside node.
- A request in the system can leave the system for another node.
- All service time is exponentially distributed.

The arrival rate λ enters the node of each server in parallel connection as shown in Fig. 1. We analyze the performance of the servers in parallel connection and the major factors μ_1, μ_2, …, μ_m, D_1, D_2, …, D_m and λ. The equivalent system response time is shown to be $E_{D_{eq}}(T) = \sum_{m=1}^{m} D_m E_{D_m}(T)$, where $\sum_{m=1}^{m} D_m = 1$ and D_m's are the dispatch probabilities, and $E_{D_m}(T)$ are the system response time of queues. Therefore the equivalent system response time is the sum of the system response times of all queues. Ten sets of different service rates and nine different dispatch probabilities are used for a total of 90 cases. The system response time of different dispatch probabilities is measured.

The system response time of the first server is shown in Eq. (1)

$$E_{D_1}(T) = \frac{1}{\mu_1 - D_1\lambda} \quad (1)$$

The system response time of the second server is shown in Eq. (2)

$$E_{D_2}(T) = \frac{1}{\mu_2 - D_2\lambda} \quad (2)$$

The dispatch probability of the second server is shown in Eq. (3)

$$D_2 = 1 - D_1 \quad (3)$$

The service rate of the first server is equal to n times that of the second server, as shown in Eq. (4)

$$\mu_1 = n\mu_2 \quad (4)$$

The system arrival rate is equal to ρ_2 times the service rate of the second server, as shown in Eq. (5)

$$\lambda = \rho_2\mu_2 \quad (5)$$

The equivalent system response time of the servers in parallel connection is shown in Eq. (6)

$$E_{eq}(T) = D_1E_{D_1}(T) + D_2E_{D_2}(T) \quad (6)$$

By applying (1), (2), (3), (4) and (5) to Eq. (6), Eq. (7) is obtained as follows.

$$E_{eq}(T) = \frac{D_1}{n\mu_2 - D_1\rho_2\mu_2} + \frac{1 - D_1}{\mu_2 - (1 - D_1)\rho_2\mu_2} \quad (7)$$

The value of the arrival rate varies from 40 to 70; the service rate of the second server is 100. The value of n in (4) ranges from 0.7 to 1.4. The dispatch probability D_1 of the first server varies from 0.1 to 1.0. The service rate of the second server μ_2 is set to be 100. Given the proper ranges and values for n, ρ_2, μ_2 and D_1 in Eq. (7), the system response time can be obtained as shown in Fig. 2.

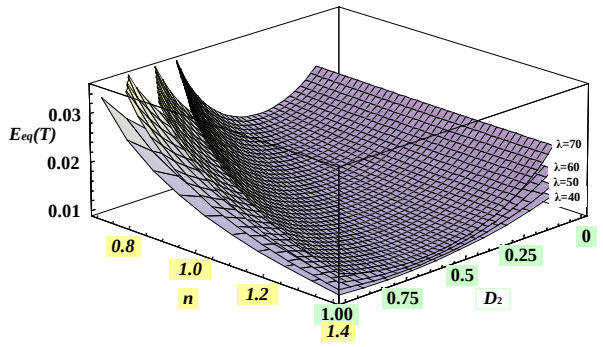

Fig. 2 System response time of different dispatch probabilities for λ= 40 - 70, *n* = 0.7 - 1.4.

The best system response time can be obtained by differentiating Eq. (7) with respect to D_1,

$$\frac{\partial E_{eq}(T)}{\partial D_1} = \frac{n\mu_2}{(n\mu_2 - \lambda D_1)^2} + \frac{\mu_2 - \lambda}{(\mu_2 - \lambda(1-D_1))^2} \quad (8)$$

and then setting the derivative equal at 0. The value of the dispatch probability is D_1 for the least system response time and is given by

$$D_1 = \frac{\lambda\sqrt{n} - \mu_2 n - \mu_2\sqrt{n}}{\lambda(\sqrt{n}-1)} \quad (9)$$

$$D_1 = \frac{\lambda\sqrt{n} + \mu_2 n - \mu_2\sqrt{n}}{\lambda(\sqrt{n}+1)} \quad (10)$$

In Eq. (9), the condition of *n*=1 would give an infinite value for D_1, which is of no concern to us. Therefore Eq. (10) is the solution. We assume $D_2 = 1 - D_1$ (7) can be rewritten as

$$E_{eq}(T) = \frac{1-D_2}{n\mu_2 - (1-D_2)\lambda} + \frac{D_2}{\mu_2 - D_2\lambda} \quad (11)$$

By differentiting the above equation with respect to D_2, the following expression is obtained.

$$\frac{\partial E_{eq}(T)}{\partial D_2} = \frac{-n\mu_2}{(n\mu_2 - \lambda(1-D_2))^2} + \frac{\mu_2}{(\mu_2 - \lambda D_2)^2} \quad (12)$$

By setting the expression equal at 0, the value of the dispatch probability D_2, which would give the minimum system response time, is given by

$$D_2 = \frac{\mu_2 n + \mu_2\sqrt{n} - \lambda}{\lambda(\sqrt{n}-1)} \quad (13)$$

$$D_2 = \frac{\lambda - \mu_2 n + \mu_2\sqrt{n}}{\lambda(\sqrt{n}+1)} \quad (14)$$

In Eq. (13), with *n*=1, the D_1 value is an infinity which we don't use. Therefore Eq. (14) is the solution.

By applying (5) to Eqs. (10) and (14) the influence of the arrival rate and the service rate on the dispatch probability is obtained.

$$D_1 = \frac{\rho_2\sqrt{n} + n - \sqrt{n}}{\rho_2(\sqrt{n}+1)} \quad (15)$$

$$D_2 = \frac{\rho_2 - n + \sqrt{n}}{\rho_2(\sqrt{n}+1)} \quad (16)$$

In keeping with Eqs.(15) and (16), when two server service rates are equal, their dispatch probabilities are equal to 0.5, and the minimum system response time can be obtained. Therefore, the combination of Eqs.(4) and (5) would lead to the following:

$$\mu_1 = \frac{n}{\rho_2}\lambda \quad (17)$$

When the system service rate is greater than the arrival rate, the system is stable. When the service rate is less than the arrival rate, the system is unstable and Eqs. (1) and (2) are not valid. The system response time is not discussed in this paper.

When the two service rates of the servers in parallel connection are not equal, and the arrival rate approaches a service rate, the dispatch probabilities of the servers affect the system response time. From Eq. (7) we learn that when $\mu_1 = n\mu_2$ is the specific value (*n*=0.1-0.9) of two service rates, and assuming the performance of Server 2 is better and the dispatch probability of Server 1 is lower than 0.5, the system response time for a minimum value can be obtained. When $\mu_1 = n\mu_2$ is the specific ratio (*n*=1.1-2.0) of two service rates, and assuming the performance of server 1 is better, its dispatch probability is consequently higher than 0.5, and the system response time can reach the lowest possible value.

To verify the lowest value of the system response time, Eq. (7) is employed with ρ_2 increasing from 0.1 to 0.9 in steps of 0.1 and *n* varying between 0.1 and 2.0. For *n* ranging between 2.0 and 10.0, the step size is increased to 1.0. Because the space is limited we choose an arrival rate of 0.9 by way of explanation and design from Fig. 3 to Fig. 7. *n* in (4) is set to range from 0.1 to 1.0, 1.1 to 2.0 and 1.0 to 10.0, three intervals. The variations of the system response time in these intervals are observed.

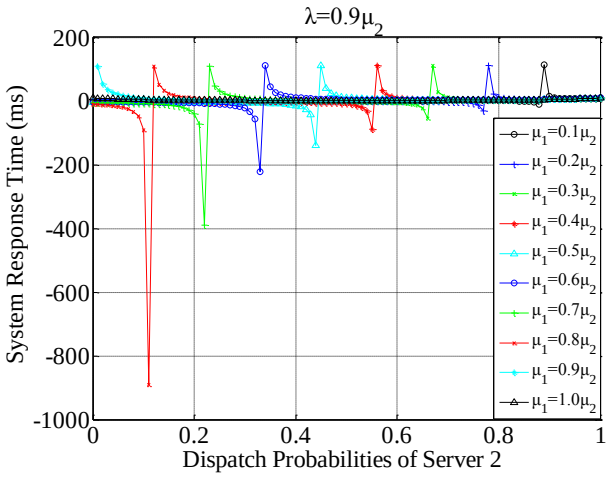

Fig. 3 System response time of different dispatch probabilities for ρ_2 = 0.9, n = 0.1 - 1.0.

In Fig. 3 nine singular points are observed. This is, as previously discussed in this paper, because when μ_2 = 1, the Server 1 service rate is less than the arrival rate. In Fig. 3 the system response time ranges from -1000 ms to 200 ms. The variation of the system response time is zoomed in and displayed in Fig. 4.

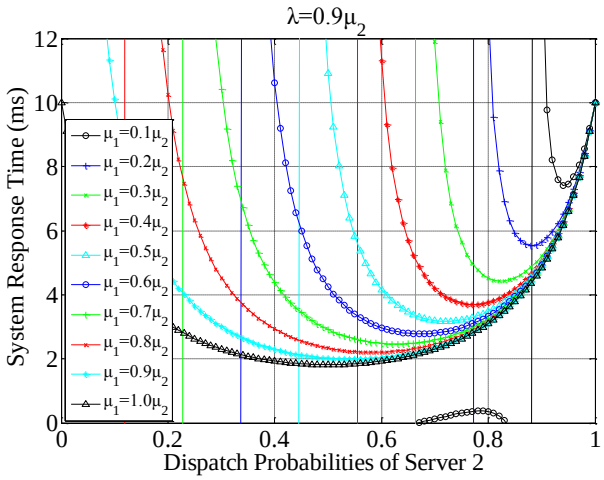

Fig. 4 System response time of different dispatch probabilities for ρ_2 = 0.9, n = 0.1 - 1.0.

In Fig. 4 the service rate of Server 2 is higher than that of Server 1. When μ_2 = 1 and μ_1 = 0.1, the system response time has a lowest value of 7.4082 ms with a dispatch probability of 0.07. At μ_1 = 0.2 the system response time has a lowest value of 5.5351 ms with a dispatch probability of 0.13. By progressing one by one in order to attain μ_1 = 1, the system response time has a lowest value of 1.8182 ms with a dispatch probability of 0.51. Therefore the performance of the two servers is the same if their dispatch probabilities are the same and the system response time has a lowest value.

In Fig. 5 no singular point is observed, because the service rates of both servers are greater than the arrival rate. Since the range of the system response time is from 0 to 12 ms, the system response time varies significantly. To set the specific value of 200%, these variations need to be zoomed in, as shown in Fig. 6.

The value of n is increased as shown in Fig. 4. In Fig. 6 the service rate of Server 1 is higher than that of Server 2. When μ_2 = 1 and μ_1 = 1.1, the lowest value of the system response time is 1.6645 ms, and correspondingly the dispatch probability is 0.55. At μ_1 = 1.2 the lowest value of the system response time is 1.5307 ms, and the corresponding dispatch probability is 0.59. The value of μ_1 is increased up to a value of 2, the system response time has a lowest value of 0.86165 ms, and its corresponding dispatch probability is 0.87. If the performance of the two servers is not the same, their dispatch probabilities need to be adjusted. For the one with a higher service rate, its dispatch probability will be higher, decreasing the system response time.

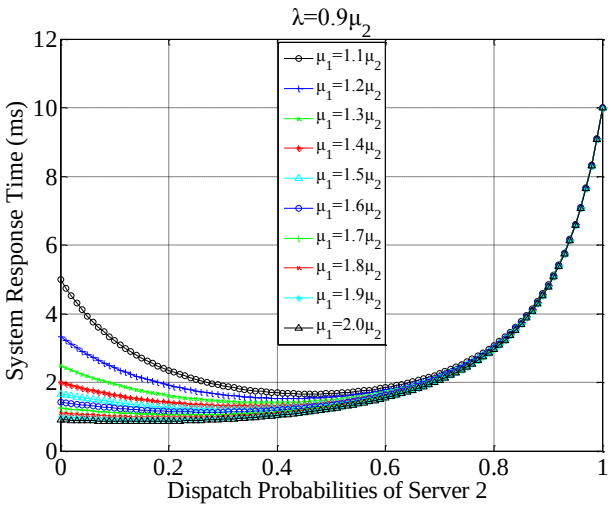

Fig. 5 System response time of different dispatch probabilities for ρ_2 = 0.9, n = 1.1 - 2.0.

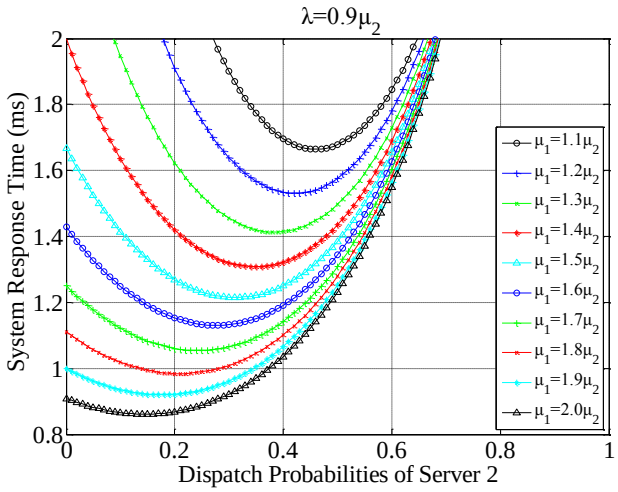

Fig. 6 System response time of different dispatch probabilities for ρ_2 = 0.9, n = 1.1 - 2.0

Singular points could not be found because the service rates of both servers were greater than the arrival rate and the range of the system response time was from 0 to 10 ms. When μ_1 = 3, the system response time will reach its lowest value which is 0.47619 ms.

This section was found in the Eqs.(15) and (16). When n is 0.1-2.0, system response time can be improved and the ratio of optimized dispatch probabilities can be gathered.

4 The Simulation of Two Servers in Parallel Connection and of Different Dispatch Probabilities

To analyze the error margins of two servers in parallel connection, the accuracy of the previous equivalent parallel equation must be examined. After modeling we employ the software network simulation tool to support our study during the simulation stage. For our simulation the Queuing Network Analysis Tool (QNAT) is chosen [38], which features a closed or open queueing network under the simulation.

This research from the simulation developed for networks in parallel connection was focused. To find the arrival rate a measurement is carried out with the rate increment from 10 requests/sec to 90 requests/sec in steps of 10 requests/sec. The service rate of Server 2 is set at 100 requests/sec. The service rate of Server 1 is set within a range storing from 10 requests/sec to 200 requests/sec in steps of 10 requests/sec and a range storing from 200 requests/sec to 1000 requests/sec in steps of 100 requests/sec. The dispatch probability D_1 ranges from 0.1 to 0.9, in steps of 0.1, while D_2 is decreased from 0.9 to 0.1 in steps of 0.1. As D_1 and D_2 vary, their sum is equal to 1.

When the arrival rates of two servers are similar to a slight ratio change of the service rate, the variation of the dispatch probability also affects the system response time. We adjust and set the service rate of Server 2 at 100 and set the value for the arrival rate, the dispatch probability and the service rate of Server 1 including all variables and use QNAT to simulate the system response time. Fig. 7 shows the cases with n varying from 0.8 to 1.6 and λ equal to 80. Let's consider the case with an arrival rate of 80 and n equal to 0.8. The system response time has a lowest value of 19.303 ms, with the dispatch probability ratio equal to 4:6. For n ranging between 0.9 and 1.1, the lowest system response time is 15.135 ms, with the dispatch probability ratio equal to 5:5. With the case of n equal to 1.2 and 1.3, the lowest system response time is 12.933 ms, with the dispatch probability ratio equal to 6:4. For the case of n=1.4, the lowest system response time is 12.017 ms, with the dispatch probability ratio equal to 7:3. For the case of n=1.5 and 1.6, the system response time has a lowest value of 10.63 ms, with the dispatch probability ratio equal to 8:2, as shown in Fig. 7.

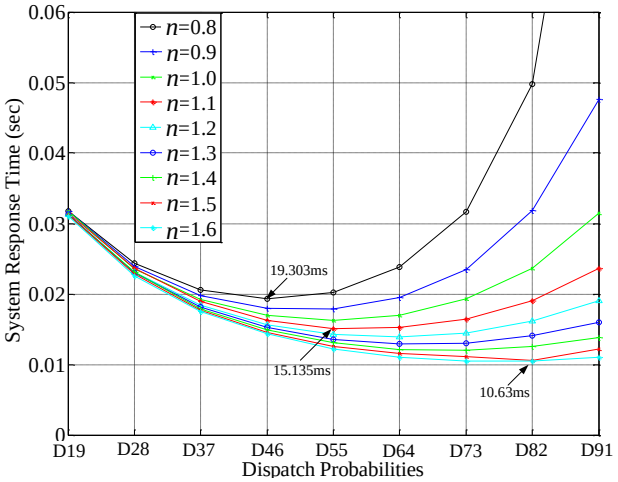

Fig. 7 System response time by simulation for λ= 80, n = 0.8 - 1.6.

5 Measurement of the Performance of Two Servers in Parallel Connection and with Different Dispatch Probabilities.

To verify the performance representation of the servers in the parallel connection of a real network, a local network is employed as the measuring environment and both the Webserver Stress Tool [28, 39] and the Active Server Pages (ASP) Web are selected for the measurements. The Server 1 IP address is set at 192.168.0.1, the Server 2 IP address at 192.168.0.2 and the Client IP address at 192.168.0.10. For the investigation of the effect on the parallel connection, the Webserver Stress Tool measurement software is chosen to carry out the measurement of the system response time of the servers in parallel connection, as shown in Fig. 8.

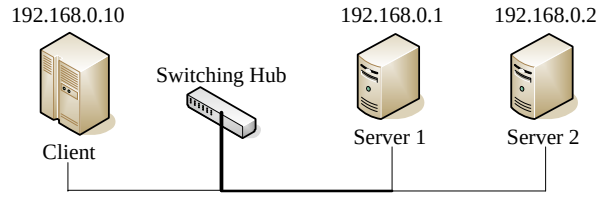

Fig. 8 Measurements of servers in parallel connection and with different dispatch probabilities.

The Webserver Stress Tool is installed to simulate the users accessing the Internet and the Client. Simulation servers 1 and 2 install IIS. In the simulation each user selects a server, and then the response waits for a server. After receiving service, the user makes a new request. Testing parameters are adjusted to run a test for 20 minutes. When the system response time increases, the number of requests from users is reduced. The number of users for the user simulation is between 1 and 100 and increases in steps of 5 each time, which is the mode that all users adopt at random request. The system records Click Time, Time to First Byte, Time to Connect and Time for DNS every 10 seconds. In the URLs parameter we descend to the bottom, selecting one link after another on the Web page. We set 10 sets of Web pages to link the service that arrives at the first server. For the other nine sets the Web page is combined with the second server. We use the dispatch probability ratio that links two servers as the basis for allocation and other ratio, 1:9, 2:8, … , 9:1.

To reduce the measurement error, the setting of the computer configurations of all Web servers is similar to the service requests for using our ASP Web page to create a similar CPU load which can be controlled by adjustable multiplication loops. We fix one ASP Multiplication loop set from 10 K to 100 K and another from 10 K to 200 K, both with incremental steps of 10 K.

The client is responsible for creating the Web service request to the Web servers while the Web servers then aim to provide a Web page service to meet the client's request. To increase the accuracy of the measurements we provide the Web page service with regular operations including a large number of mathematical operations. We adjust the two servers to different service rates. Their dispatch probability ratio is 1:9, and the number of the Web users ranges from 1 to 100. We then carry out the performance measurement. The measurement results change when we adjust the dispatch probabilities, from 1:9, 2:8, 3:7,… and then to 9:1. The number of Web users ranges from 1 to 100 with incremental steps of 10.

We use the ASP Multiplication loop which, increasing from 100 K to 200 K, consists of a total of 20 different servers in parallel connection with different service rates and with dispatch probability ratios at 1:9, 2:8, 3:7, …, 9:1, a total of nine measurements, thus comparing 180 sets of data in all. Each set of data is used by 1 to 100 users. By adding 5 users for each set, our total is 21 measurements, which gives us 3780 measurements altogether. From our experiment we find that the service rate of two parallel servers with ASP Multiplication loop =100 K and 90 K, 100 K and 110 K and the dispatch probability ratio of 5:5 has the lowest system response time, as shown in Fig. 9.

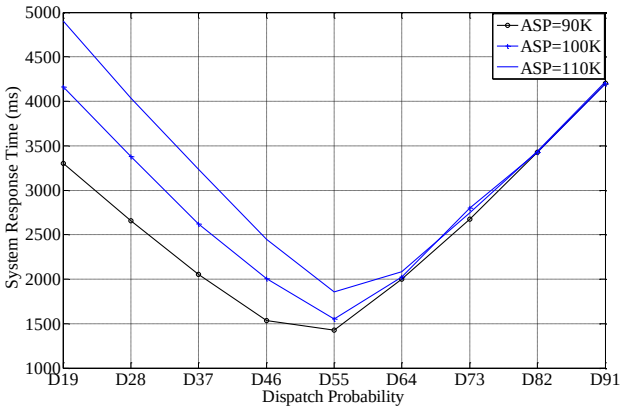

Fig. 9 ASP=100 K: Parallel system response time of different dispatch probabilities with CPU load ASP=90 K to ASP=110 K with incremental steps of ASP=10 K.

When the service rates of a single server are less than the ASP Multiplication loop =100 K, the overall system response time of two servers in parallel connection is improved as shown in Fig. 10.

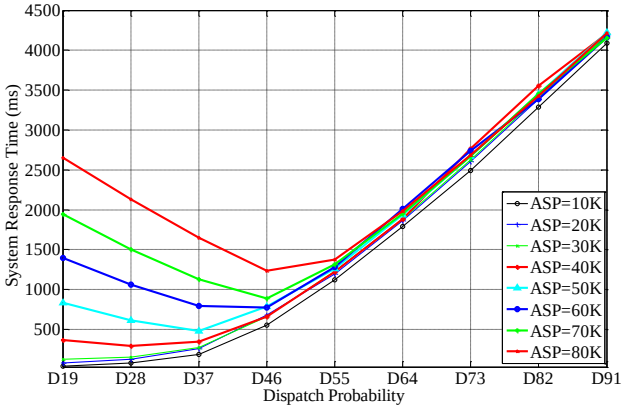

Fig. 10 ASP=100 K: Parallel system response time of different dispatch probabilities with a CPU load of ASP=10 K to ASP=80 K with incremental steps of ASP=10 K.

By using two servers in parallel connection with a similar service rate one can obtain the lowest system response time from the ratio of their different dispatch probabilities. Fig. 11 shows two servers in parallel connection with the loadings ASP=100 K and ASP=50 K and the dispatch probability ratio of 1:9, but the system response time isn't the lowest; however, it will be the lowest when the ratio is 3:7 (D37).

When the service rate of two servers in parallel connection is the same, the system response time is the lowest at the dispatch probability ratio of 5:5 (D55). The system response time is the lowest when the loading of two servers in parallel connection is ASP=100 K with a dispatch probability ratio of 5:5.

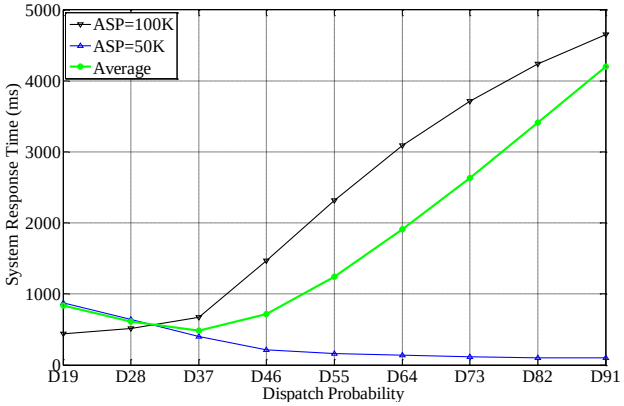

Fig. 11 Three system response times of servers in parallel connection and two different dispatch probabilities with CPU load ASP = 100 K, ASP = 50K and their average.

Eq. (7) was used with a low blocking probability, since the buffer size was not taken into consideration, we therefore had to add the buffer size to Eq. (7). The buffer size is decided as a result of both the measurement time and the system response time of Eq. (7), because comparison with the service rate, the different arrival rate would lead to a different buffer size. When the arrival rate gradually becomes greater than the service rate, the size of the buffer will also gradually increase. With an actual measurement time of 20 minutes, the arrival rate is consequently subjected to a limit. Therefore most of the system response time of the measurement is higher, 27501.32 ms, with ASP = 100 K vs. 200 K, and the dispatch probability ratio is 1:9.

The difference between the measurement and the correction of Eq. (7) is shown in Fig. 12 with CPU load ASP=100 K vs. 10 K … 200 K and the number of users within the range of 1 to 100. If the ASP ratio is n=0.1, the error can reach as high as 606.45%, and if it is n=0.4, the error can still reach 152.41%. When the ASP ratio n is above 2.5, the service rate of two servers varies. One has a light load, and the other has a heavy load. The buffer occupancy of the server with a light load has very few requests, and the buffer of the server with a heavy load requires a larger buffer size. Among the dispatch probabilities, the buffer occupancy of a heavy load server is different. We use an average value of the buffer size in Eq. (7) which causes a big error margin. In this operational situation, because the discrepancy of the buffer is very big, the load is unbalanced, and the requests are abandoned. When the ASP ratio n is less than 2, the load of the two servers is similar. As a result the buffer occupancy is also similar. We use an average value of the buffer size in Eq. (7) which reduces the error margin. We find that when the ASP ratio is n=0.8-1.6, the error ranges from 19.19% to 40.89%, and the average error is 27.49%.

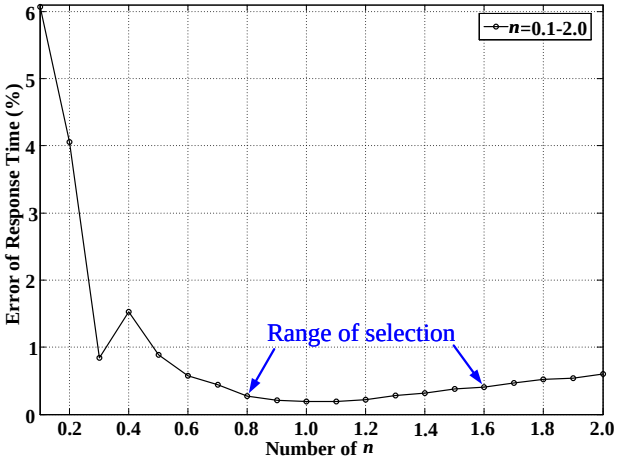

Fig. 12 Average error between the corrected (7) and measurements.

From the data of the measurement we use the least square method and locate the service rate of

the servers based on the loads of ASP ranging from 1 to10^6, as shown in Table 2. In Eq. (7), as the dispatch probability ratio D_1 ranges from 0.1 to 0.9, n is the ratio of the two service rates of the servers. The arrival rate λ used in the measurement has a value between 1 and 100 users, and the service rates are as shown in Table 2. With respect to the two servers, Server 2 uses the dispatch probability ratio to evacuate the performance, under the condition of ASP=100 K, and its equivalent service rate is 98. In modified Eq. (7) we need to estimate the average buffer size according to the variation of the arrival rates.

Table 2 Server service rate as obtained both from measurements and by the least square method

ASP	Service rate	ASP	Service rate
1	236	140 K	96
1000	195	150 K	96
10 K	111	160 K	96
20 K	102	170 K	95
30 K	101	180 K	95
40 K	99	190 K	95
50 K	99	200 K	95
60 K	99	300 K	93
70 K	99	400 K	92
80 K	98	500 K	90
90 K	98	600 K	88
100 K	98	700 K	87
110 K	97	800 K	86
120 K	97	900 K	85
130 K	97	1 M	84

When the value of the ASP ratio of the two servers, n, varies between 0.1 and 2.0, and the arrival rate varies between 1 and 100, then the error margin of modified Eq. (7) varies between 13.28% and 206.77%, and the average error margin is 94.96%. When n varies between 0.8 and 1.6, and the arrival rate varies between 1 and 100, then the error margin of modified Eq. (7) varies between 5% and 50.17%. The average error margin is 27.49%, as shown in Fig. 13. If n varies between 0.1 and 2.0 the error margin tenuously increases and rises, as opposed to n varying between 0.8 and 1.6, when the error margin rises.

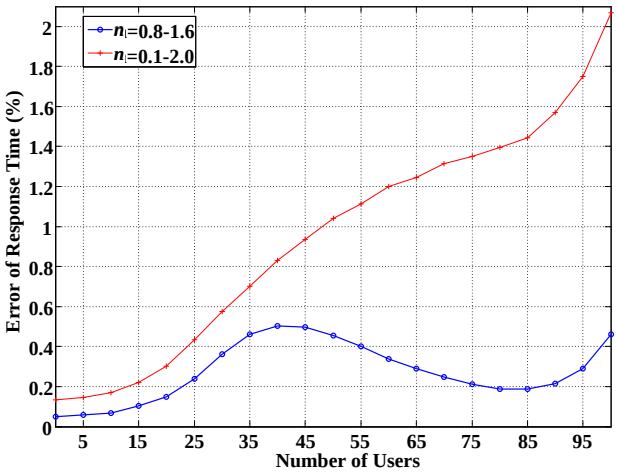

Fig. 13 Comparison of error between results of corrected (7) and of measurements, with n within ranges 0.8-1.6 and 1.0-2.0.

We use the load ASP=100 K of Server 2 with the service rate, the system arrival rate and the dispatch probability ratio of Server 2 set for the variable and use a Web-Stress Tool to measure the system response time. Fig. 14 shows n=0.8-1.6 and arrival rate λ=80 as the premise for the measurement. It shows that when the number of users is equal to 80 and the ASP ratio n is equal to 0.8, the system response time is lowest, with a value of 2.328275 secs and the dispatch probability ratio is 4:6. When n=0.9-1.1, the system response time is lowest, with a value of 2.832 secs, 2.92864 secs and 3.636063 secs respectively, and the dispatch probability ratio is 5:5. When n=1.2-1.6, the system response time is lowest with a value of 4.294413 secs, 4.218888 secs, 4.448613secs, 4.470825 secs and 5.069088 secs respectively, and the dispatch probability ratio is 6:4. This is shown in Fig. 14.

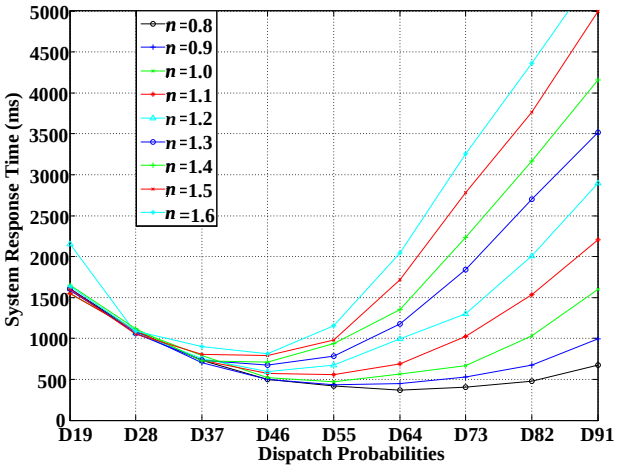

Fig. 14 λ_2=80: Parallel system response time under different dispatch probabilities with n varying from 0.8 to n=1.6 in steps of n=0.1.

From the measurement, the simulation and through the characteristic curves we can understand how the system response time is closely related to the dispatch probability ratio. We can further expect to see that when the ASP ratio of two servers approaches 1 with the dispatch probability ratio equal to 5:5, the minimal system response time becomes available.

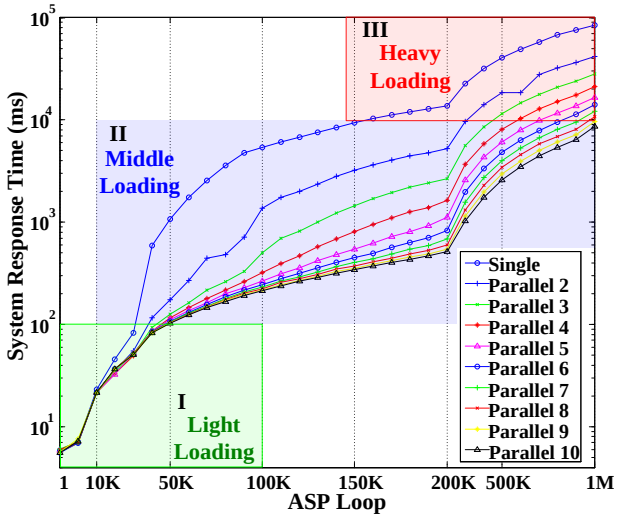

Fig. 15 System response time in various regions of the service rates.

Fig. 15 shows the measurement data in three working regions. A logarithm scale is chosen for the Y axis to express the characteristics of the system response time. In Fig. 15, by utilizing a Light Loading, the parallel connection can't reduce the system response time significantly. However, for a Medium Loading, the parallel connection reduces the system response time by up to four times. If the number of servers in parallel connection increases, the system response time can be reduced by a factor of $1/2m$ - $1/m^2$. With a Heavy Loading the system response time is decreased by $1/m$ of a single server [40].

This section found the measurement: when n is 0.8-1.6, system response time can be improved and the ratio of optimized dispatch probabilities can be gathered.

6 Conclusions

If the service rates of two parallel servers are the same, the dispatch probability shall be the same as in the model to obtain the best balance and to reduce the system response time by a maximum according to our experimental measurements. If the service rates of two servers are different, the dispatch probability should be chosen based on the service rate of the server.

The parallel connection servers with a medium to high CPU load can improve the system response time because of the nonlinear characteristics of the system response time. After investigating the characteristics of the servers in parallel connection, we will in the future work towards partitioning complex servers in series-parallel connection, which are capable of predicting the system response times.

When two servers operate in parallel connection, and their service rates differ by less than 200%, we can fine-tune the dispatch probability to find out the best system response time. It is not possible for the difference to be greater than 200%.

In the region of heavy loading, servers in parallel connection with the same service rate can improve their performance when their ratio of dispatch probability is equal. At a medium loading, two servers, in parallel connection with the same service rate can improve their overall performance. In the region of light loading we do not need to increase the number of servers since the improvement on the performance would be trivial.

References:

[1] F. S. Tu, D. P. Song, and Sheldon X. C. Lou, "Steady state performances analysis in stochastic serial production lines," *Proceedings of the 32nd IEEE Conference on Decision and Control*, vol.3, 1993, pp. 2749-2751.

[2] Chung-Ping Chen, Ying-Wen Bai and Yin-Sheng Lee, "Performance Measurement and Queueing Analysis with Low Blocking Probability of Serial Connection Networks," *Proceedings of the I²MTC 2008 – IEEE International Instrumentation and Measurement Technology Conference*, Victoria, Vancouver Island, Canada. 2008, pp.2216-2221.

[3] Chung-Ping Chen, Ying-Wen Bai and Cheng-Hung Tsai, "Performance Measurement and Queueing Analysis at Medium-High Blocking Probability of Parallel Connection Servers with Identical Service Rates," *Proceedings of the 8th WSEAS International Conference on Communications on Data Networks, Communications, Computers (DNCOCO '09)*, Baltimore, USA,pp. 173-178, November 7-9, 2009.

[4] Gunter Bolch, Stefan Greiner, Hermann de Meer, and Kishor S. Trivedi, *Queueing Networks and Markov Chains*, Wiley-Interscience, 2006.

[5] C. Ykman-Couvreur, J. Lambrecht, D. Verkest, F. Catthoor, A. Nikologiannis, and G. Konstantoulakis, "System-level performance optimization of the data queueing memory management in high-speed network processors," *Proceedings of Design Automation Conference, 2002, 39th*, 2002, pp. 518 – 523.

[6] Yingwu Zhu, and Yiming Hu, "Efficient, proximity-aware load balancing for DHT-based P2P systems," *IEEE Transactions on Parallel and Distributed Systems*, vol.16, no.4, 2005, pp. 349-361.

[7] Chyouhwa Chen, and Kun-Cheng Tsai, "The Server Reassignment Problem for Load Balancing in Structured P2P Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol.19, no.2, 2008, pp.234-246.

[8] Qi Zhang, ALma Riska, Wei Sun, Evgenia Smirni, and Gianfranco Ciardo, "Workload-aware load balancing for clustered Web servers," *IEEE Transactions on Parallel and Distributed Systems*, vol.16, no.3, 2005, pp. 219-233.

[9] Jiani Guo, and Laxmi Narayan Bhuyan, "Load Balancing in a Cluster-Based Web Server for Multimedia Applications," *IEEE Transactions on Parallel and Distributed Systems*, vol.17, no.11, 2006, pp.1321-1334.

[10] Ho-Lin Chen, Jason R. Marden, and Adam Wierman, "On the Impact of Heterogeneity and Back-End Scheduling in Load Balancing Designs," *Proceedings of the IEEE INFOCOM 2009 - IEEE 28th Conference on Computer Communications*, Rio de Janeiro, Brazil, 2009 , pp.2267-2275.

[11] Jih-Sheng Chang and Ruay-Shiung Chang, "A Probabilistic cost Estimation Model for RFID Real-Time Applications in Data Grids," *International Journal of Innovative Computing, Information and Control*, vol.6, no.12, 2010, pp.5421-5438.

[12] Shunsuke Oshima and Takuo Nakashima, "Performance Evaluation for Linux Under SYN Flooding Attacks," *International Journal of Innovative Computing, Information and Control*, vol.5, no.3, 2009, pp.555-565.

[13] Takuo Nakashima, "Properties of the Correlation Between Queue Length and Congestion Window Size Under Self-Similar Traffics," *International Journal of Innovative Computing, Information and Control*, vol.5, no.11(B) , 2009, pp.4373-4381.

[14] Takuo Nakashima, "Queueing Property for Different Type of Self-Similar Traffics," *International Journal of Innovative Computing, Information and Control*, vol.6, no.3(B) , 2010, pp.1279-1288.

[15] Xinlu Ma, Lindu Zhao and Lothar Schulze, "Performance of Automated Storage/Retrieval Systems Under Stochastic Demand Using Queueing Theory," *International Journal of Innovative Computing, Information and Control*, vol.5, no.12(A) , 2009, pp.4469-4477.

[16] Manjur SK, Mosleh Abu-Alhaj, Omar Abouabdalla Tat-Chee Wan, Rahmat Budiarto and Ahmed M. Manasrah, "Conference Gateway for Heterogeneous Clients: Real Time Switching Clients and Interasterisk Exchange Clients," *International Journal of Innovative Computing, Information and Control*, vol.7, no.1, 2011, pp.395-406.

[17] Hongjie Li, "Synchronization Stability for Discrete-Time Stochastic Complex Networks with Probabilistic Interval Time-Varying Delays," *International Journal of Innovative Computing, Information and Control*, vol.7, no.2, 2011, pp.697-708.

[18] Vincent S. Tseng, Ming-Hua Hsieh and Kawuu W. Lin, "A Novel Cache Replacement Algorithm for Cooperative Caching in Wireless Multimedia Sensor Networks," *International Journal of Innovative Computing, Information and Control*, vol.7, no.2, 2011, pp.763-776.

[19] Hsiu-Yuan Chu, Kuo-Hui Tsai and Wen-Jer Chang, "Fuzzy Control of Active Queue Management Routers for Transmission Control Protocol Networks via Time-delay Affine Takagi-Sugeno Fuzzy Models," *International Journal of Innovative Computing, Information and Control*, vol.4, no.2, 2008, pp.291-312.

[20] Jiann-Fu Lin, "Scheduling Parallel Tasks with Intra-Communication Overhead in a Grid Computing Environment," *International Journal of Innovative Computing, Information and Control*, vol.7, no.2, 2011, pp.881-896.

[21] Reu-Ching Chen and Cheng-Min Lin, "Adaptive End-to-End Delay Equalizations for TCP Virtual Path Transmissions in Internet Environments," *International Journal of Innovative Computing, Information and Control*, vol.6, no.3(A) , 2010, pp.1069-1078.

[22] Chin-Fu Ku, Sao-Jie Chen and Jan-Ming Ho, "Improving End-to-End Communication Performance by Controlling Behaior of Intermediate Network Node," *International Journal of Innovative Computing, Information and Control*, vol.6, no.4, 2010, pp.1893-1904.

[23] Albert L. Hopkins, T. Basil Smith, III, and Jaynarayan H. Lala, "FTMP—A highly reliable

fault-tolerant multiprocess for aircraft," *Proceedings of the IEEE* , vol.66, no.10, 1978, pp. 1221-1239.

[24] H. Zha, X. S. Han, Y. L. Wang, and L. Han, "An effective algorithm of the power system probabilistic optimal dispatching," *Proceedings of the Third International Conference on Electric Utility Deregulation and Restructuring and Power Technologies, 2008. DRPT 2008,* International Conference Hotel of Nanjing, 2008, pp.1092-1096.

[25] Oliviero Barana, Adriano Luchetta, Gabriele Manduchi, and Cesare Taliercio, "A general-purpose Java tool for action dispatching and supervision in nuclear fusion experiments," *IEEE Transactions on Nuclear Science*, vol. 49, no 2, 2002, pp. 469-473.

[26] Siqing Sheng, Youjiang Sun, Tan Liu, Wenqin Zhang, and Yihan Yang, "Random adaptive optimizer restores distribution service," *IEEE Computer Applications in Power*, vol. 13, no. 2, 2000, pp. 48-51.

[27] K. Shanti Swarup and B. Satish, "Integrated ANN approach to forecast load," *IEEE Computer Applications in Power*, vol. 15, no. 2, 2002, pp. 46-51.

[28] Yan Liu, and Xueping Gu, "Skeleton-Network Reconfiguration Based on Topological Characteristics of Scale-Free Networks and Discrete Particle Swarm Optimization," *IEEE Transactions on Power Systems*, vol. 22 no. 3, 2007, pp. 1267-1274.

[29] Chung-Ming Huang, Chung-Wei Lin, and Wan-Ping Tsai, "A multi-core based parallel streaming mechanism for concurrent video-on-demand applications." *IEEE Communications Letters*, vol. 13, no. 4, 2009, pp. 286-288.

[30] Raimner Ohlendorf, Michael Meitinger, Thomas Wild, and Andreas Herkersdorf, "A Processing Path Dispatcher in Network Processor MPSoCs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 16, no. 10, 2008, pp. 1335-1345.

[31] Bill Schilit, Jason Hong, and Marco Gruteser, "Wireless location privacy protection," *Computer*, vol. 36, no 12, 2003, pp. 135-137.

[32] M. Palanivelan and Sheila Anand, "Reduced Complexity Max Norm Based PAPR Optimization in OFDM Systems," *Proceedings of WSEAS Transations on Communications*, Issue 5, Volume 11, May 2012, pp.171-181.

[33] Hua Hou and Gen-Xuan Li, "Cross-layer Packet Dependent OFDM Scheduling Based on Proportional Fairness," *Proceedings of WSEAS Transations on Communications*, Issue 1, Volume 11, January 2012, pp. 1-15.

[34] Hichem Chaker, Sidi Mohamed Meriah and Fethi Bendimerad, "One and Two Dimensions Unequally Array Pattern Synthesis with the use of a Modified Particle Swarm Optimization Algorithm," *Proceedings of WSEAS Transations on Communications*, Issue 6, Volume 11, June 2012, pp. 207-217.

[35] Milan Tuba, Nebojsa Bacanin and Nadezda Stanarevic, "Adjusted Artificial Bee Colony (ABC) Algorithm for Engineering Problems," *Proceedings of WSEAS Transations on Computers*, Issue 4, Volume 11, April 2012, pp. 111-120.

[36] Manjusha Pandey and Shekhar Verma, "Enhanced Energy Efficient Privacy Provisioning in WSN, " *Proceedings of WSEAS Transations on Communications*, Issue 7, Volume 11, July 2012, pp.262-273.

[37] Vincent Roberge and Mohammed Tarbouchi, "Parallel Particle Swarm Optimization on Graphical Processing Unit for Pose Estimation," *Proceedings of WSEAS Transations on Computers*, Issue 6, Volume 11, June 2012, pp. 170-179.

[38] Hema Tahilramani Kaur, D. Manjunath and Sanjay K. Bose, "The Queueing Network Analysis Tool (QNAT)," *Proceedings of International Symposium on Modeling, Analysis and simulation of Computer and Telecommunication Systems*, 2000, pp. 341-347.

[39] http://www.paessler.com/webstress/

[40] Chung-Ping Chen and Ying-Wen Bai, "Performance Measurement and Queueing Analysis at Medium-High Blocking Probability of Parallel Connection Servers with Identical Service Rates," *WSEAS Transactions on Communications*, Issue 12, Vol. 8, 2009, pp. 1253-1262.