Computing and Informatics, Vol. 29, 2010, 1181–1192

AN EMPIRICAL STUDY ON MULTICRITERIA SCHEDULING

Wijak Srisujjalertwaja, Pattarasinee Bhattarakosol

Department of Mathematics
Faculty of Science, Chulalongkorn University
10330, Thailand
e-mail: wss@cs.science.cmu.ac.th

Manuscript received 28 January 2009; revised 16 September 2009
Communicated by Iveta Zolotová

Abstract. This paper presents an empirical study of non-preemptive Multicriteria-Based, called MCB for short, scheduling policy. MCB scheduling policy uses multiple criteria of each request: arrival time, deadline, and processing time, to balance the requirements on both client and server sites. Weighted aggregation method is applied in this study to conduct the different measurements to a single figure of merit. For the empirical study, an M/G/1 queuing simulation system is implemented with MATLAB to represent a general server's incoming request scheduling system. Comparative simulation results of MCB with best effort scheduling policy on an overload situation show that MCB is an optimal scheduling policy.

Keywords: Scheduling, multicriteria scheduling, M/G/1 queuing simulation

Mathematics Subject Classification 2000: 68T05

1 INTRODUCTION

Nowadays, internet plays an important role in all aspects of everyday life such as business, communication, education, and entertainment. The population of the Internet users grows rapidly which leads to high competition in every layer of services. As a result, an approach that can improve the quality of service in this high competitive situation is the adaptive application that can adapt itself to high variable environments.

Every user's request arriving at the application server has to be queued for an incoming request scheduling as a first step. Here an important problem is the human response time (HRT) where users abandon their requests when no response has been received from the required server over the limits of their interaction delay. Generally, the server uses a fixed scheduling policy, best effort or First-In First-Out (FIFO), to handle the incoming requests. However, the flexible scheduling policy should be considered if the service has been enhanced to the adaptive application.

Multicriteria scheduling is an interesting approach for the flexible scheduling policy. According to the goal of an application server that needs to serve as many users as possible under the limited resources and various user requirements that have many objectives and usually conflict, the multicriteria concept is an appropriate method to provide an optimal solution.

The rest of this paper is organized as follows: Section 2 reviews background of the problem and related works, Section 3 states the research approach, Section 4 describes our experiment and results in detail, Section 5 presents discussion of this study, summarizes our conclusions and offers possible directions for future works.

2 BACKGROUND AND RELATED WORKS

Generally, the application developers develop their application based on functionality of the application by leaving server's resource control duties to the operating systems. When a user's requests become more diverse and vast, either working on the Internet or expanding into mobile and/or wireless computing, critical service problems to application servers occur. Dalal and Jordan [3] considered an impatient user problem that occurs when a user abandons a pending Web request if the response is not available in several seconds. The action causes the server to waste resources on those unwanted requests. Such situations may ultimately prove disastrous and lead to a server deadlock crisis on a heavily loaded server.

Due to this critical situation and limited resources of the server such as memory, disk bandwidth, communication bandwidth, and CPU cycle, resource management is vital. The adaptive application is a technique to extend an application to adapt itself to variable environments. In this study, the considered approach of the adaptive application is incoming request scheduling [1, 3, 4, 10, 11]. This is an approach to adjust handling of the processing order of the incoming request on scarce resources system.

Scheduling can be separated into two running characteristics: preemptive scheme and non-preemptive scheme. In preemptive scheme, another high priority process can interrupt the running process, whereas in non-preemptive scheme, the running process must be completed before another is executed.

The well-known scheduling policies include, for example, First-In-First-Out (FIFO), Earliest-Deadline-First (EDF), Shortest-Processing-Time-First (SPT),

Last-Come-First-Served (LCFS), Round-Robin (RR) with a fixed quantum, Lottery, Fair-Share, and Biggest-In-First-Served (BIFS). Normally, as mentioned earlier, most scheduling policies intend to meet merely a single criterion. In addition, Stankovic et al. [7] presented that the knowledge of the request and its factors such as deadline, processing time, precedence, future release times, and so on, play an important role in the scheduling policy for both uniprocessor and multiprocessor real-time systems.

Many scheduling techniques are proposed as an admission part of many systems. For example, Almeida et al. [1] implemented a priority-based scheduling by assigning priorities to the user requests according to the requested documents. Then, Wang et al. [10] implemented a system that classifies the incoming requests into different priority queues according to their pre-negotiated priorities, and uses a scheduler in queuing part to reschedule the classified requests before sending them to the destination application.

It is always expected that the single criterion method can satisfy the user requirements. This unidimensional viewpoint responses to the policy objective in one particular criterion, while usually ignoring other criteria. For example, in SPT policy, the determining criterion which will help increase the number of successful requests is the shortest processing time; however, other criteria such as the deadline and the arrival order of the request are ignored.

T'Kindt and Billaut [8, 14] introduced a definition of a Multicriteria Scheduling Problem (MSP). They recommended that the minimization of several conflicting criteria could change the way to handle scheduling problems. The definitions of weak and strict Pareto optimality are encountered to optimally minimize all the criteria, especially the strict Pareto optimum that is necessary for the multicriteria scheduling in computing system. In addition, Hoogeveen [15] presented a survey on multicriteria scheduling. He presented two bicriteria problems, discussed the results of single machine and multiple machines scheduling, and proposed a new bicriteria model on the problem with interfering job sets that compete to be processed on a single machine.

Multicriteria scheduling concept has been applied to many computing systems and communication layers. There are many techniques to determine a solution for multicriteria problem such as weighted aggregation, fuzzy set theory, goal programming, and utility theory. For example, Cherkasova [2] proposed a tuneable scheduling strategy, called Alpha scheduling. This is a bicriteria with non-preemption policy. It was between FIFO and SPT with the value of a coefficient Alpha. It could improve the overall response time per HTTP request more than three times when heavily loaded.

Furthermore, T'Kindt and Billaut [8] suggested that only a small number of problems consider the minimization of more than two criteria. Thus, we present an empirical study of a three criteria scheduling policy, called MCB. We describe the problem with mathematical model and use simulation technique to substantiate the study.

3 RESEARCH APPROACH

In this study, incoming request scheduling for a general application server is concerned. Normally, for user or client site, fairness service is required. On the other hand, for server site, serving as many users as possible is the main objective. To balance these requirements of both the client and the server sites, we apply weight aggregation technique to multicriteria scheduling policy; MCB is called to conduct these different measurements to a single figure of merit. Instead of using single criterion, multiple criteria of each request (arrival time, deadline, and processing time) are manipulated for MCB. These criteria dominate the system's quality of service called waiting time.

The properties of arrival time, deadline, and processing time are reflected in three traditional single criterion policies. As summarized from [12, 13], first, FIFO policy intends to prioritize account by using the request's arrival time. It works on fair service, which can be inferred to minimize variance of waiting time. Second, EDF policy aims to keep each request on its deadline. It minimizes the maximum lateness of execution time, which can be inferred as the minimum amount of tardy request, i.e. the number of jobs that complete after their due date. Finally, SPT policy considers to execute the request with the shortest processing time first. It attempts to minimize mean flow time, which can be inferred as the maximum amount of serviceable requests.

These findings suggest that both EDF and SPT are used to model the scheduling problem in server site viewpoint, which emphasizes overall satisfaction of most users, although paying less attention to the individual user satisfaction at client site. On the other hand, FIFO states its attempt to guarantee the fairness among users by focusing on maximizing the individual user satisfaction on the client site.

The reference model, the mathematical model, and the scheduling algorithm of our proposed model are shown in the following subsections.

3.1 Reference Model

Generally, incoming request service works in the best effort, FIFO, manner. When a client sends a request to an application server, the request is held in the server's request pool waiting for a processing cycle. Having received the signal, it is pushed to the FIFO queue and waits for the server processing. To improve this incoming service, a scheduler is added to organize the requests into FIFO queue as shown in Figure 1.

The timeline of each request is shown in Figure 2. Arrival time, t_a, is the point of time at which a request arrives at the system's request pool. Waiting time, Δt_w, is the time that the request waits for execution. Launch time, t_l, is the point of time at which the request is executed; it is calculated as follows:

$$t_l = t_a + \Delta t_w.$$

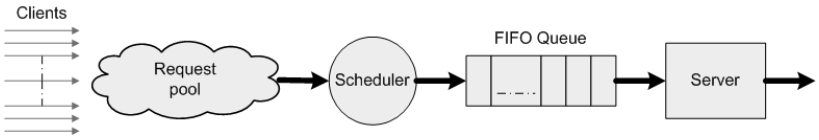

Fig. 1. System reference model

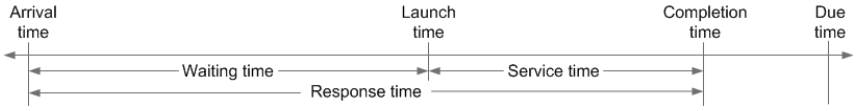

Fig. 2. Timeline of each request

Processing time or service time, Δt_p, is the duration of time used to execute the request. Completion time, t_c, is the point of time at which the request execution has been completed. Response time, t_r, is the total time that the request takes from arrival at the system until its execution is completed, which is calculated as follows:

$$t_r = t_a + \Delta t_w + \Delta t_p.$$

Finally, deadline or due time, t_d, is the point of time at which the request is expected to complete its job, which is calculated as follows:

$$t_d = initiationTime + \Delta t_p + patientTime.$$

The initiation time is the time at which the process is started. The patient time is the time at which the process can be patient for the service. It can be predefined as a commitment in a service license agreement or as a real waiting time of a process for a service.

3.2 Mathematical Model

Based on Multicriteria Decision Making (MCDM) and multicriteria scheduling concept, we describe the compromising problem of MCB with mathematical model. We assess the MCB by help of three well-known factors of the incoming request: arrival time, deadline, and processing time. Definition of the problem is presented as follows.

Definition 1. An aggregated function of objective values of each request: An aggregated function of objective values of each request for scheduling, hereafter referred to as function $\mathcal{F}(x_1, x_2, x_3)$, is defined as follows:

$$\mathcal{F}(x_1, x_2, x_3) = w_1 x_1 + w_2 x_2 + w_3 x_3$$

where

- w_1, w_2, and w_3 are weights of processing time, deadline, and arrival time, respectively,
- x_1, x_2, and x_3 are the variables representing normalized values of amount of processing time, deadline, and arrival time, respectively.

Normalization is a systematic way of ensuring that data suit for general-purpose processing and minimize the bias. For example, suppose x_1 ranges between 1 000 000 and 2 000 000, x_2 ranges between 0.5 and 0.7, x_3 ranges between 10.0 and 20.0, and w_1, w_2, and w_3 are in fair weight; corresponding to Definition 1, x_1 influences over the calculation. Z-score approach is applied to normalize these variables. The standard formula for Z-score is

$$z = \frac{x - \mu}{\sigma}$$

where x is a raw score to be normalized, μ is the mean of the population in each run, and σ is the standard deviation of the population. It is most frequently used to compare a sample to a standard normal deviation which is the way to reduce the bias.

3.3 Scheduling Algorithm

The scheduling algorithm of the reference system is presented as follows:

Set weight w_1, w_2, w_3
Set normalizationFlag
Clear request queue
Do while true
    Get requests from the request pool and append to the request queue
    If normalizationFlag = true then
        Normalize features, x_1, x_2, x_3, of all requests
    End If
    Sort the request queue by weight of each request ascending, where
        $Request_i = w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3}$
    Set request pool time
    Do
        Execute request in the request queue
    Until end of the request pool time
End while

First, weight of each criterion is set when a system is started. Then if the system needs to normalize the request features, the normalizationFlag is set to true.

While running the system, the incoming request scheduling system works in the admission module as shown in the algorithm. It gets the incoming requests from the request pool and appends them to the request queue. This process is done every period of time called request pool time. After appending, the request queue is rescheduled depending on the system policy, which is based on the problem model in Definition 1. The request queue is then sorted by weight of each request ascending where w_1, w_2, and w_3 are weights of processing time, deadline, and arrival time, respectively. Also, x_{i1}, x_{i2}, and x_{i3} are the variables representing normalized values (if required) of processing time, deadline, and arrival time, respectively. Last, the request pool time is set and requests in the request queue are executed until end of the time.

For scheduling complexity analysis, suppose a general sorting algorithm like Quick sort is applied to EDF, SPT, and MCB algorithms, worst-case running time of each scheduling cycle being $\Theta(n^2)$. In addition, if the normalization process, the applied Z-score approach, is done, linear function complexity of the MCB algorithm increases while FIFO has no overhead of normalization and sorting processes.

4 EXPERIMENT AND RESULTS

To verify the scheduling model described in Section 3, the simulation model, assumptions, and environment are set as follows.

4.1 The Simulation Model

Following [2, 3, 11, 16, 17], we implement a simple application server's incoming request scheduling simulator based on MATLAB environment. The incoming request queueing problem is modeled in the M/G/1 queueing system. We consider a single server queue with an overload situation. The queue has the request arrival in a Poisson process with an average arrival rate λ. It has generally distributed service time with service rate μ. The Poisson process is a continuous-time, discrete-state process where the intervals between successive events are independent and identically distributed according to an exponential distribution $F(x) = 1 - e^{-\lambda x}, x > 0$. The server utilization, ρ, equals to λ/μ, while the probability that the server is idle, ν_0, equals to $1 - \rho$. For more definitions and notations of queuing system, see [5, 9].

We present an aggregated function of objective values of each request, from Definition 1, by applying weighted aggregation method to decide a scalar constraint value for MCB policy. This value is used to organize the requests in the FIFO queue where the request with minimum value gets the highest priority to execute. The criteria in this study, arrival time, deadline, and processing time, have values of the same magnitude, therefore normalization process is not required. A general function of the problems can be denoted as

$$f_i = \sum_{j=1}^{K} w_j x_{ij} \tag{1}$$

where i is a request index, f_i is a scalar constraint value of each request, j is a criterion index, K is number of considered criteria, w_j is weight of the j^{th} criterion, $w_j \geq 0, \forall j$ and $\sum_{j=1}^{K} w_j = 1$, and x_{ij} is a value of the j^{th} criterion of the i^{th} request. Weight summation can be assigned to any value. It is determined as the priority of the criteria.

In this paper, processing time, deadline, and arrival time are considered as critical criteria of the condition. Each request is characterized by the following parameters:

$$\mathit{Request}_i(x_{i1}, x_{i2}, x_{i3}) \tag{2}$$

where i is a request index, as mapping from (1), x_{i1} is processing time of the i^{th} request, x_{i2} is deadline of the i^{th} request, and x_{i3} is arrival time of the i^{th} request. For MCB, the request with the minimum constraint value has the highest priority to be executed. The scheduling function of MCB policy can be represented as

$$g_i = w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} \tag{3}$$

where g_i is a scalar value of the i^{th} request, x_{i1} is processing time of the i^{th} request, x_{i2} is deadline of the i^{th} request, x_{i3} is arrival time of the i^{th} request, and $w_1, w_2, w_3 \geq 0$ and $w_1 + w_2 + w_3 = 1$. This MCB's scheduling function can be mapped to the other scheduling policies as follows: for FIFO, arrival time is weighted with w_3 and equals to 1 while w_1 and w_2 equal to zero; for EDF, deadline is weighted with w_2 and equals to 1 while w_1 and w_3 equal to zero, and lastly, for SPT, processing time is weighted with w_1 and equals to 1 while w_2 and w_3 equal to zero.

In this study, FIFO and MCB scheduling policies are applied to the simulation model. To measure each scheduling policy, the experiment is set to a steady state where the entire system has stabilized. Corresponding to MCB scheduling policy's objective, the number of succesful requests, average waiting time, maximum waiting time, as well as standard deviation (Sd.) waiting time are performance measurements of the experiment.

4.1.1 Assumptions

We assume that the system is a single queue server based on single machine scheduling problem. Every scheduling policy runs on non-preemptive characteristics where other requests could not interrupt the processing of the current request. Incoming request's buffer in the queueing system has a finite size. Samples of a general method to find the processing time is presented by Ye et al. [11] and NS-2 [6]. A request size is divided by service rate to decide about the processing time of the request. To enable direct performance measurement and to control the bias of each scheduling policy, the system's running time is assessed from the system's processing times,

while the other times, such as request getting time and scheduling time, are trivial. In addition, we assume that there are no timeouts of the request in any case. Request arrival time is a Poisson distribution and inter-arrival time is an exponential distribution. Each incoming request comes with arrival time, deadline, and request size information, which can be used to calculate the request processing time. The request size and deadline are assumed to be a normal distribution. Consequently, the system model becomes the M/G/1 queueing model. Finally, the arrival time, deadline, and processing time have values in the same magnitude, therefore normalization process is not required.

An admission control policy is proposed to handle the quality of service of MCB policy compared to a best effort system, FIFO, that has non-QoS handling process. The system measures the waiting time that starts when a request enters to the incoming queue and ends when the request is processed or rejected. The system with the admission control process handles its incoming requests by rejecting all requests in the request pool if the incoming queue is full and rejecting the requests that have deadline time longer than the system's current time.

4.1.2 Environment

We assign a fixed processing rate to the server, 300 Kbytes per second, with a fixed processing cycle. The request size has average size 75 Kbytes per request with Sd. 10 Kbytes. Thus the server's service rate (μ), i.e. the server's processing rate divided by the average request size, equals to 4 requests per second. The request's deadline has average time 2.0 seconds with Sd. 0.2. Estimated success rate is the capability of the system that can process the requests. It equals to $100 \times 1/\rho$ or $100 \times \mu/\lambda$. We decide to test the system in an overload situation where the estimated success rate is 40 %. Therefore, arrival rate or λ is 10 requests per second. The running case has thirty minutes period. Thus, total number of request is 18 000 requests. We test the system by deadline checking to assess the deadline effect on each policy.

4.2 The Simulation Results

It can be observed that there are two systems to compare: a server with admission control process and a best effort server. We choose two MCB results, MCB-2 and MCB-3, as instances of MCB approach for the server with admission control process comparing to FIFO as a scheduling policy for the best effort server. MCB-2 is an instance of a bicriteria policy with $w_1 = w_2 = 0.5$ and $w_3 = 0$. MCB-3 is an instance of a multicriteria policy with $w_1 = w_2 = w_3 = 0.33$.

Testing was made on five queue sizes, 5, 10, 20, 30, 40. The results are shown in Table 1. Observe the results where queue size is 20, the rate of success of MCB-3 is 8.58 % and MCB-2 is higher by 8.54 % than FIFO. Furthermore, average waiting time of MCB-3 is 48.97 % and that of MCB-2 is faster by 49.45 % than FIFO. Comparing to the same scheduling policy viewpoint, the results show that increasing queue

size does not improve scheduling performance, while having important effect on average and maximum waiting times. As can be seen from the table, the number of successful request results of queue sizes 30 and 40 does not differ from the result of queue size 20. On the other hand, the maximum waiting times of queue sizes 30 and 40 increased significantly compared to the result of queue size 20.

Queue size	Policy	Successful	Avg. wait	Max. wait	Sd. wait
5	FIFO	7 362	0.1918	1.1612	0.2159
5	MCB-3	7 399	0.3779	2.1772	0.3035
5	MCB-2	7 420	0.3750	2.0779	0.3000
10	FIFO	7 362	0.4674	1.8208	0.3630
10	MCB-3	7 771	0.4668	2.4954	0.3853
10	MCB-2	7 768	0.4672	2.5235	0.3884
20	FIFO	7 362	1.0376	2.1371	0.3985
20	MCB-3	8 053	0.5295	2.4857	0.4763
20	MCB-2	8 051	0.5245	2.4427	0.4726
30	FIFO	7 362	1.1245	2.1371	0.3805
30	MCB-3	8 068	0.5340	3.3828	0.4817
30	MCB-2	8 070	0.5324	3.3255	0.4847
40	FIFO	7 362	1.1249	2.1371	0.3804
40	MCB-3	8 068	0.5340	3.3828	0.4817
40	MCB-2	8 070	0.5324	3.3255	0.4847

Table 1. Comparison of the admission control process result with the best effort result

5 DISCUSSION AND CONCLUSION

In this paper, a non-preemptive multicriteria scheduling policy, called MCB, is presented as an empirical study. Incoming request scheduling for an application server is the considered application. The three well-known characteristics of the request: arrival time, deadline, and processing time, are considered as the scheduling criteria. The objective of MCB is to minimize the average, maximum, and Sd. waiting times. The scheduling model is expressed in the scheduling function with the weighted aggregation method. An M/G/1 simulation system is implemented based on MATLAB to represent the performance of the policy.

Comparing to the traditional scheduling policy, FIFO, the simulation results indicate that MCB is an optimal policy in some situations. It can be observed on the results that where queue size is more than 10, the rate of success of MCB-2, which is a bicriteria policy, and MCB-3, which is a tricriteria policy, are higher by about 9 % than FIFO. Furthermore, average waiting times of MCB-2 and MCB-3 are faster by about 50 % than FIFO. In addition, queue size has a major effect on the experiment results. Thus, it is important to find the proper or the optimal queue size for individual systems to ensure the highest performance.

For future work, we recommend to focus on increasing the capabilities of the simulation system considering the other objectives of the system. The capabilities of the simulation system, for instance, preemptive running, differentiated services, and other request characteristics, will be increased to fulfill the system. The other objectives of the system, such as minimizing mean and maximum lateness, the number of tardy job, and the number of rejected request, will be considered for comparison. Furthermore, the proposed policy will be applied as part of the dynamic admission control system of the adaptive QoS Web server to improve the user satisfaction.

REFERENCES

[1] Almeida, J.—Dabu, M.—Manikutty, A.—Cao, P.: Providing Differentiated Levels of Service inWeb Content Hosting. Proceedings of the FirstWorkshop Internet Server Performance, Wisconsin, June 1998.

[2] Cherkasova, L.: Scheduling Strategy to Improve Response Time for Web Applications. Proceedings of the International Conference and Exhibition on High-Performance Computing and Networking, April 1998, pp. 305–314.

[3] Dalal, A. C.—Jordan, S.: Improving User-Perceived Performance at a World Wide Web Server. Proceedings of the Global Telecommunications Conference GLOBECOM '01, November 2001, Volume 4, pp. 2465–2469

[4] Elnikety, S.—Nahum, E.—Tracey, J.—Zwaenepoel, W.: A Method for Transparent Admission Control and Request Scheduling in E-Commerce Web Sites. Proceedings of the 13th International Conference on World Wide Web, New York, May 2004, pp. 276–286.

[5] Gelenbe, E.—Pujolle, G.: Introduction to Queueing Networks. John Wiley & Sons, 1999.

[6] The Network Simulator (NS Version 2). Available on: `http://www.isi.edu/nsnam/ns/`, 2008.

[7] Stankovic, J. A.—Spuri, M.—di Natale, M.—Buttazzo, G. C.: Implications of Classical Scheduling Results for Real-Time Systems. Computer, June 1995, Volume 28, No. 6, pp. 16–25.

[8] T'Kindt, V.—Billaut, J.-C.: Some Guidelines to Solve Multicriteria Scheduling Problems. Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC '99), Tokyo, October 1999, Volume 6, pp. 463–468.

[9] Trivedi, K. S.: Probability and Statistics with Reliability, Queuing and Computer Science Applications. John Wiley & Sons, 2002.

[10] Wang, C.—Tham, C. K.—Jiang, Y.: A Framework of Integrating Network QoS and End System QoS. Proceedings of the IEEE International Conference on Communications (ICC '02), April-May 2002, Volume 2, pp. 1225–1229.

[11] Ye, N.—Gel, E. S.—Li, X.—Farley, T.—Lai, Y.-C.: Web Server QoS Models: Applying Scheduling Rules from Production Planning. Computers & Operations Research, May 2005, Volume 32, No. 5, pp. 1147–1164.

[12] Baker, K. R.: Introduction to Sequencing and Scheduling. Computers & Operations Research, 1974.

[13] French, S.: Sequencing and Scheduling: An Introduction to the Mathematics of the Job-shop. Computers & Operations Research, 1982.

[14] T'Kindt, V.—Bullaut, J.-C.: Multicriteria Scheduling Problems. Multiple Criteria Optimization: State of the Art Annotated Bibliographic Survey, Springer New York 2003, pp. 445–491.

[15] Hoogeveen, H.: Multicriteria Scheduling. European Journal of Operational Research, Vol. 167, 2005, pp. 592–623.

[16] Robertsson, A.—Wittenmark, B.—Kihl, M.: Analysis and Design of Admission Control in Web-Server Systems. Proceedings of the American Control Conference, June 2003, Vol. 1, pp. 254–259.

[17] Chen, K.—Decreusefond, L.: An Approximate Analysis of Waiting Time in Multi-Class M/G/1/./EDF Queues. ACM SIGMETRICS Performance Evaluation Review, Vol. 24, May 1996, No. 1, pp. 190–199.

Wijak Srisujjalertwaja received his B. Sc. (CS) degree from Bangkok University, Thailand, in 1991 and his M. Sc. (CS) degree from Chiang Mai University, Thailand, in 1997. In 2009 he received his Ph. D. (CS) from Chulalongkorn University, Thailand. Currently he is a lecturer at Department of Computer Science, Faculty of Science, Chiang Mai University,Thailand, where his research interests include Adaptive Application, Recommender System, Trust Management, and Software Engineering.

Pattarasinee Bhattarakosol became an ACM member since 2000 and an IEEE member since 2002. She obtained her Doctoral degree in Computer Science from Wollongong University, Wollongong, Australia, in the year 1996. Her major area is in software engineering, solving problems of distributed information causing by the network corruption. She received her M. Sc. degree in applied statistics from National Institute of Development Administration, Bangkok, Thailand in 1986 and B. Sc. degree in mathematics from Chulalongkorn University, Bangkok, Thailand in 1983, respectively. Currently, she is an Assistant Professor at the Department of Mathematic, Faculty of Science, Chulalongkorn University, Bangkok, Thailand. She has publications in technical journals and conference proceedings, as well as a book chapter.