Queuing Network Approximation Technique for Evaluating Performance of Computer Systems with Input to Terminals

Hikari Yoshii, Nozomi Miyamoto, Mohamad Zaini Nurshafiqah Binti,

Daisuke Miyake, Itaru Koike and Toshiyuki Kinoshita

School of Computer Science, Tokyo University of Technology
Hachioji Tokyo, 192-0982, Japan

Abstract: *Queuing network techniques are effective for evaluating the performance of computer systems. We discuss a queuing network technique for a computer system with input to terminals. The finite number of terminals exists in the network and a job arrives randomly from outside of the terminal. After a think-time at the terminal, the job moves to the server, and it acquires some parts of memory and executes CPU and I/O processing in the server. After the job completes CPU and I/O processing, it releases the memory and goes back to its own terminal. However, because the terminal and the memory resource can be considered as a secondary resource for the CPU and I/O equipment, the queuing network model has no product form solu-tion and cannot be calculated the exact solutions.*

We proposed here an approximation queuing network technique for calculating the performance measures of a computer system with input to terminals on which multiple types of jobs exist. This technique involves dividing the network into two levels; one is "inner level" in which a job executes CPU and I/O processing, and the other is "outer level" that includes terminals and communication lines. By dividing the network into two levels, we can prevent the number of states of the network from increasing and approximate the performance measures of the network. We evaluated the proposed approximation technique by using numerical experiments and clarified the characteristics of the system response time and the mean number of jobs in both level.

Keywords: *performance evaluation, queuing network, central server model, a computer system with input to terminals*

1. Introduction

Queuing network techniques are effective for evaluating the performance of computer systems. In computer systems, two or more jobs are generally executed at the same time, which causes delays due to conflicts in accessing hardware or software resources such as the CPU, I/O equipment, or data files. We can evaluate how this delay affects the computer system performance by using a queuing network technique. Some queuing networks have an explicit exact solution, which is called a product form solution [1]. With this solution, we can easily calculate the performance measures of computer systems, for example the busy ratio of hardware and the job response time. However, when the exclusion controls are active or when a memory resource exists, the queuing network does not have a product form solution. To calculate an exact solution of a queuing network that does not have a product form solution, we have to construct a Markov chain that describes the stochastic characteristics of the queuing network and numerically solve its equilibrium equations. When the number of jobs or the amount of hardware in the network increases, the number of states of the queuing network drastically increases. Since the number of states of the queuing network is the same as the number of unknown quantities in the equilibrium equations, the number of unknown quantities in the equilibrium equations also drastically increases. Therefore, we cannot numerically calculate the exact solution of the queuing network. Moreover, when the queuing network is an open model where jobs arrive from or depart for the outside of the network, the number of states of the network can become infinite (the number of jobs can be infinite), and we cannot actually calculate an exact solution.

Here we discuss the queuing network model for computer systems with input to terminals (Figure. 1). In the model, the job arrives randomly from the outside to the network and acquires a terminal. If all terminals are occupied, the

https://doi.org/10.17758/ERPUB.ER1217248

job joins the system waiting queue and wait until a terminal becomes available. After a think-time at the terminal, the job moves to the server and acquires some parts of the memory and exe-cutes CPU and I/O processing. When the job completes CPU and I/O processing at the server, it releases the memory and goes back to its own terminal. This model resembles that a customer enters and leaves an ATM (Automated Teller Machine) terminal.

Since a job executes CPU and I/O processing occupying the terminal and the memory, they can be considered as a secondary resource for the CPU and I/O equipment. Generally, when a queuing network includes a secondary re-source, it does not have product form solutions

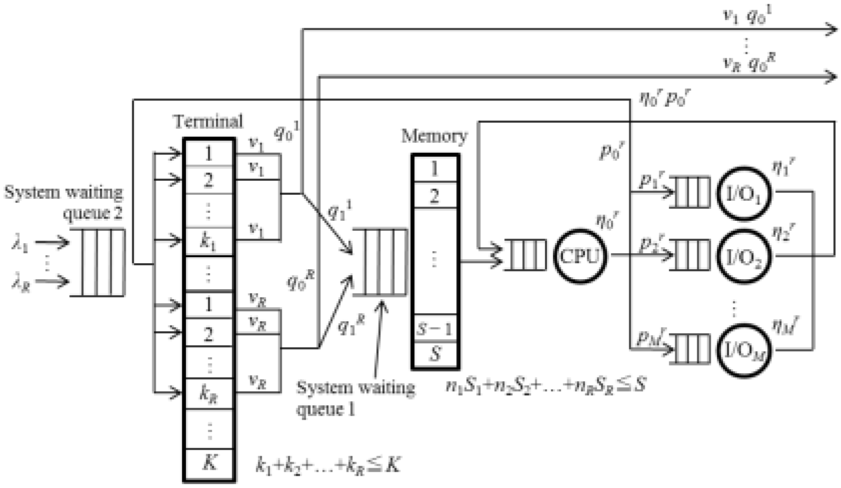


Fig 1. Central server model with input to terminals

To get the strict solution of the Model, we have to construct a Markov chain which de-scribes the entire model and have to numerically solve its equilibrium equa-tions. In order to prevent the number of states of the Markov chain from in-creasing, we divide the model into two levels, one is outer level that includes the terminals and commu-nication lines, and the oth-er is inner level that in-cludes CPU, I/O equip-ment and memory re-sources (Figure 2).

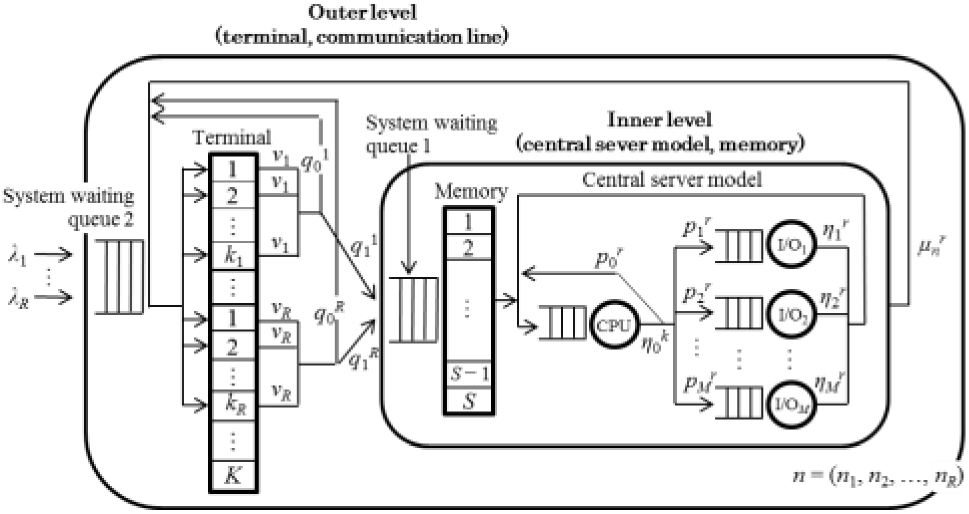


Fig 2. Concept of approximation

In the same ways as [8][9], there are multiple types of job class exist in the network. Each job class behaves differently in the outer level and the inner level. Although both the inner level and the outer level has a product form solution when the model has solution when the model has a single job class, the both level does not have a product form solution when the model has multiple job classes. Therefore, an approximation technique for the both level is needed to analyse its performance measures.

In this paper, we have proposed an approximation technique for calculating the performance measures of a computer system with input to terminals. We previously reported multiple job class including memory resource model arrived randomly from the outside [8] and a model in which a job moves back and forth between a terminal and a network [9]. In this research, we report a model in which a job arrives via a terminal from the outside including the memory resource.

Dividing the model into two levels is one of two-layer queuing network techniques [3]. Our proposed technique is also a two-layer technique for a computer system with input to terminals. In our previous study [4], we reported an approximation technique for evaluating performance of computer systems with file resources. Meanwhile, heterogeneous parallel computer systems with distributed memory was researched in [6], and the Markov chain involving two dimensional state transition similar to our proposed model was discussed in [7].

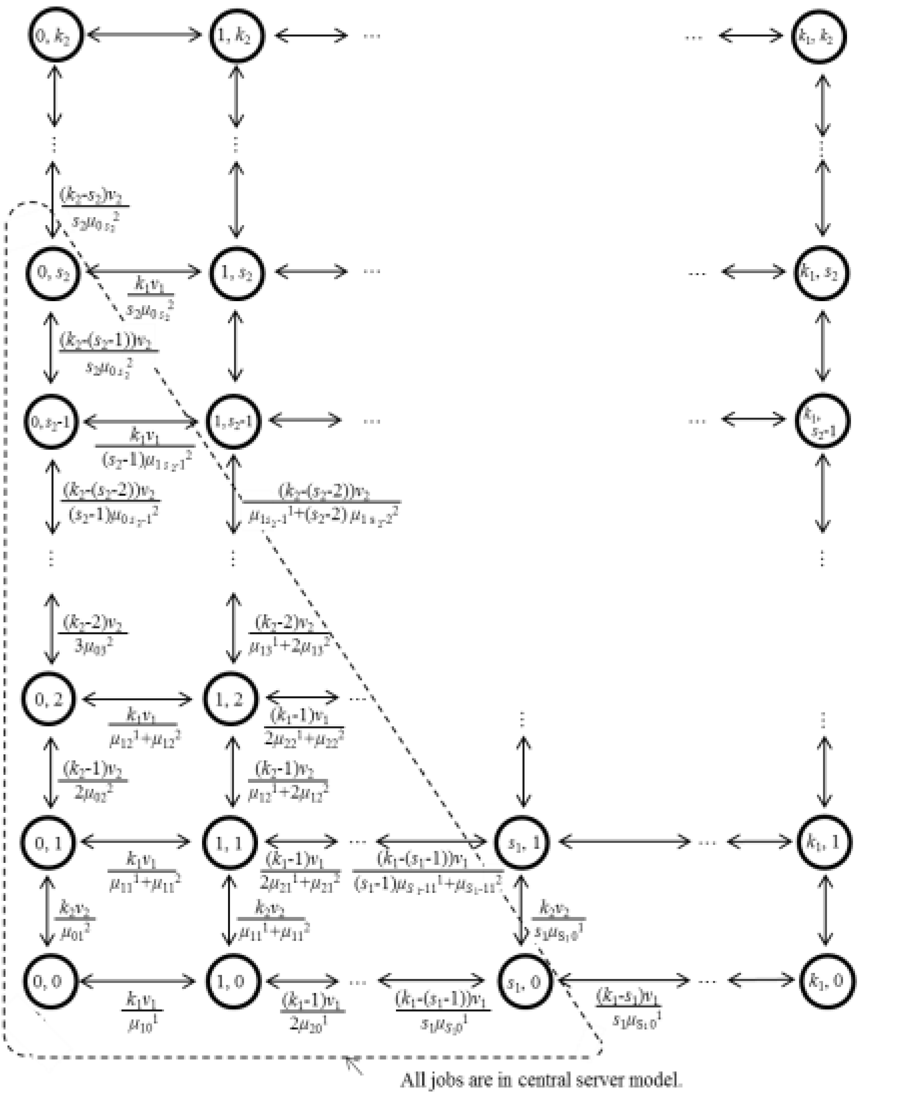


Fig.3: State Transition diagram of computer system with input terminals (two job classes, tow dimensional birth-death process

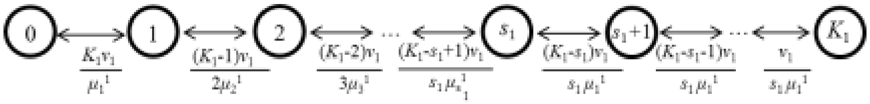


Fig.4: State transition diagram (signal job class)

2. Model Description

The CPU and I/O model in the inner level is equivalent to the ordinary central server model with multiple job types (each of which is called a job class). In this model, R job classes exist, and each of them is numbered r = 1, 2, … , R by affixing r. We denote n_r as the numbers of jobs of job class r in the central server model and n as the total number of jobs in the central server model. We also denote i_r as the number of jobs of job class r in the inner level (the difference of n_r and i_r is the number of jobs in the system waiting queue 1). The inner level consists of single CPU node and multiple I/O nodes. We denote M as the number of I/O nodes. The I/O nodes are numbered m = 1, 2, … , M by affixing m, and the CPU node is numbered m = 0 by also affixing m. The service rate of job class r at the CPU node is $\eta_0{}^r$, and the service rate of job class r at an I/O node m is $\eta_m{}^r$. The service time at each node is a mutually independent random variable subject to common exponential distributions. Jobs are scheduled on a first come first served (FCFS) principle at all nodes. At the end of CPU processing, a job probabilistically selects an I/O node and moves to it, or completes CPU and I/O processing and goes back to its own terminal. The selection probability of I/O node m of job class r is $p_m^r\ (m=1,2,\dots,M;\ r=1,2,\dots,R)$ and the completion probability of job class r is p_0^r. Therefore, $\sum_{m=0}^{M} p_m^r = 1$ $(r=1,2,...,R)$.

In the outer level, there are K terminals exist and the job arrives randomly from the outside to the network and acquires a terminal. If all terminals are occupied, the job joins the system waiting queue and wait until a terminal becomes available. When the job completes the CPU and I/O processing in the central server model, it returns to its own terminal. We denote kr as the number of jobs of the job class r (r = 1, 2, ... , R) acquiring the terminal. These k_r are not constant and $\sum_{r=1}^{R} k_r \le K$ holds.

The job stays in the terminal for short while. The staying time is called “think-time”. The think-time is mutually independent random variable subject to common exponential distribution with parameter v_r of job class r (v_r is job departure rate from the terminal).

Memory resources are added to this central server model (Figure 1). We denote S_r as the number of the units of the memory acquired by a job of job class r and S as the total number of the units of the memory. After the think-time, a job of job class r moves to the inner level, and requests and acquires S_r units of the memory before entering the central server model. If sufficient units of the memory do not available, the job joins the system waiting queue 1 and waits for the memory to be released by another job. When the job completes CPU and I/O processing, it releases the memory and leaves the inner level and goes back to its own terminal in the outer level. Since the job has to acquire the memory before entering the central server model, the total number of units of the occupied memory in the central server model has to be less than or equal to S, i.e. $\sum_{r=1}^{R} n_r S_r \le S$.

By replacing “CPU → outer level transition” with “CPU → CPU transition,” the central server model is modified to a closed model in which the number of jobs is constant (Figure 2). 0049n this modified model, when “CPU → CPU transition” occurs, we consider as the job terminates and a new job is born. Therefore, the mean job response time is the mean time between two successive “CPU → CPU transitions.” This means that the job response time can be considered as the job lifetime.

3. Approximation Model

We use the following notations.

t_r : mean think-time of job class r

v_r : departure rate from the terminal of job class r

τ_{rm} : mean total service time of job class r at node-m in the central server model

S_r : number of units of memory acquired by a job of job class r

S : total number of units of memory resorce

n_{rm} : number of jobs of job class r at node-m in the central server model (r=1, 2, ..., R; m=0, 1, ... , M)

n_r : number of jobs of job class r in the central server model

$n = (n_1, n_2, \ldots, n_R)$: vector of number of jobs in the central server model (n_r=0, 1, 2, ..., k_r)

i_r : number of jobs of job class r in the inner level

k_r : number of jobs of job class r in the network (= number of terminals of job class r)

$n^* = (n_{10}, n_{11}, \ldots, n_{1M}, n_{20}, n_{21}, \ldots, n_{2M}, \ldots, n_{R0}, n_{R1}, \ldots, n_{RM})$: state vector of the central server model

$$F(n) = \{n^* \mid \sum_{m=0}^{M} n_m = n_r,\, n_m \ge 0\ (m = 0, 1, \ldots M)\}\ (n_1 S_1 + n_2 S_2 + \ldots n_R S_R \le S)$$

: set of all feasible states of the central server model when the number of jobs of job class r is n_r

$P_s(n^*)$: steady-state probability of state n^*

$T_n{}^r$: mean job response time of the central server model when the vector of number of jobs is n

$\mu_n{}^r$: completion rate from the central server model of job class r

T^r : system response time of job class r (= lifetime of job class r)

3.1. Inner level

Since the central server model in the inner level is equivalent to the ordinary central server model with multiple job classes, it has the product form solution. Then the steady-state probability Ps(n*) is represented by the following formula [1][2].

https://doi.org/10.17758/ERPUB.ER1217248

$P_s(n^*) = \dfrac{\prod_{r=1}^{R}\prod_{m=0}^{M}\tau_{rm}{}^{n_{rm}}}{\varphi(n_1, n_2, \cdots, n_R, M)}$, where $\varphi(n_1, n_2, \cdots, n_R, M) = \sum_{n\in F(n)}\prod_{r=1}^{R}\prod_{m=0}^{M}\tau_{rm}{}^{n_{rm}}$ is the normalizing constant of steady-state probabilities when the number of jobs of job class r in the central server model is n_r (=0, 1, 2, … , k_r ; r =1, 2, ... , R). From these steady-state probabilities, we can calculate the mean job response time $T_n{}^r$ of job class r as $T_n^r = \dfrac{n_r \cdot \varphi(n_1, \cdots, n_r, \cdots, n_R, M)}{\varphi(n_1, \cdots, n_r - 1, \cdots, n_R, M)}$, when the number of jobs in the central server model is n_r.

The memory resource in inner level can be considered as an M/M/S queuing model with S servers. In an ordinary M/M/S queuing model, the service rate at a server is constant, regardless of the number of guests in the service. In the memory resource of our model, however, the service rate changes depending on the number of occupied memories. The mean job response time $T_n{}^r$ of job class r (=1, 2, ... , R) when the vector of number of jobs is $n = (n_1, n_2, \ldots, n_R)$ is equal to the mean time while the memory is occupied. Since the completion rate $\mu_n{}^r$ of job class r from the central server model is denoted as $\mu_n{}^r = \dfrac{1}{T_n^r}$, $\mu_n{}^r$ also depends on $n = (n_1, n_2, \ldots, n_R)$, that is the number of jobs in the central server model. The state transition of the M/M/S queuing model with two job classes is shown in Figure 3, where the completion rates from the central server model change depending on the number of jobs in the central server model. This is a two dimensional birth-death process. The equilibrium equations with the steady-state probability $Q_S(i_1, i_2)$, when the total number of the units of the memory is S and the number of jobs in the inner level is (i_1, i_2), are as follows (similar to the case with higher dimensions). Where s_r is the maximum integer such as $s_r S_r \le S,$ i.e. $s_r = \left[S \diagup S_r \right]$.

(1) $i_1=0,\ i_2=0$

$(k_1v_1 + k_2v_2) \cdot Q_S(0, 0) = \mu_{10}{}^1 \cdot Q_S(1, 0) + \mu_{01}{}^2 \cdot Q_S(0, 1)$

(2) $0 < i_1S_1 \le S,\ i_2 = 0$

$\{(k_1 - i_1)\, v_1 + k_2v_2 + i_1\, \mu_{i_1 0}^1\} \cdot Q_S(i_1, 0)$

$= (k_1 - i_1 + 1)\, v_1 \cdot Q_S(i_1 - 1, 0) + (i_1 + 1)\, \mu_{i_1+10}^1 \cdot Q_S(i_1 + 1, 0) + \mu_{i_1 1}^2 \cdot Q_S(i_1, 1)$

(3) $S < i_1S_1,\ i_1 < k_1,\ i_2 = 0$

$\{(k_1 - i_1)\, v_1 + k_2v_2 + s_1\mu_{s_1 0}^1\} \cdot Q_S(i_1, 0) = (k_1 - i_1 + 1)\, v_1 \cdot Q_S(i_1 - 1, 0) + s_1\mu_{s_1 0}^1 \cdot Q_S(i_1 + 1, 0) + \mu_{i_1 1}^2 \cdot Q_S(i_1, 1)$

(4) $i_1 = k_1,\ i_2 = 0$

$(k_2v_2 + s_1\mu_{s_1 0}^1) \cdot Q_S(k_1, 0) = v_1 \cdot Q_S(k_1 - 1, 0) + \mu_{k_1 1}^2 \cdot Q_S(k_1, 1)$

(5) $i_1 = 0,\ 0 < i_2S_2 \le S$

$\{k_1v_1 + (k_2 - i_2)\, v_2 + i_2\, \mu_{0i_2}^2\} \cdot Q_S(0, i_2)$

$= (k_2 - i_2 + 1)\, v_2 \cdot Q_S(0, i_2 - 1) + \mu_{1i_2}^1 \cdot Q_S(1, i_2) + (i_2 + 1)\, \mu_{0i_2+1}^2 \cdot Q_S(0, i_2 + 1)$

(6) $i_1 = 0,\ S < i_2S_2,\ i_2 < k_2$

$\{k_1v_1 + (k_2 - i_2)\, v_2 + s_2\mu_{0S_2}^2\} \cdot Q_S(0, i_2) = (k_2 - i_2 + 1)\, v_2 \cdot Q_S(0, i_2 - 1) + \mu_{1i_2}^1 \cdot Q_S(1, i_2) + s_2\, \mu_{0s_2}^2 \cdot Q_S(0, i_2 + 1)$

(7) $i_1 = 0,\ i_2 = k_2$

$(k_1v_1 + s_2\mu_{0s_2}^2) \cdot Q_S(0, k_2) = v_2 \cdot Q_S(0, k_2 - 1) + \mu_{1k_2}^1 \cdot Q_S(1, k_2)$

(8) $0 < i_1,\ 0 < i_2,\ (i_1 + 1)\, S_1 + (i_2 + 1)\, S_2 \le S$

$\{(k_1 - i_1)\, v_1 + (k_2 - i_2)\, v_2 + i_1\, \mu_{i_1i_2}^1 + i_2\, \mu_{i_1i_2}^2\} \cdot Q_S(i_1, i_2) = (k_1 - i_1 + 1)\, \nu_1 \cdot Q_S(i_1 - 1, i_2)$

$+ (k_2 - i_2 + 1)\, \nu_2 \cdot Q_S(i_1, i_2 - 1) + (i_1 + 1)\, \mu_{i_1+1i_2}^1 \cdot Q_S(i_1 + 1, i_2) + (i_2 + 1)\, \mu_{i_1i_2+1}^2 \cdot Q_S(i_1, i_2 + 1)$

(9) $0 < i_1,\ 0 < i_2,\ i_1S_1 + i_2S_2 \le S$ and $S < (i_1 + 1)\, S_1 + i_2S_2$

$\{(k_1 - i_1)\, v_1 + (k_2 - i_2)\, v_2 + i_1\, \mu_{i_1i_2}^1 + i_2\, \mu_{i_1i_2}^2\} \cdot Q_S(i_1, i_2) = (k_1 - i_1 + 1)\, \nu_1 \cdot Q_S(i_1 - 1, i_2)$

$+ (k_2 - i_2 + 1)\, \nu_2 \cdot Q_S(i_1, i_2 - 1) + i_1\, \mu_{i_1i_2}^1 \cdot Q_S(i_1 + 1, i_2) + (i_2 + 1)\, \mu_{i_1i_2}^2 \cdot Q_S(i_1, i_2 + 1)$

(10) $0 < i_1,\ 0 < i_2,\ i_1S_1 + i_2S_2 \le S$ and $S < i_1S_1 + (i_2 + 1)\, S_2$

$\{(k_1 - i_1)\, v_1 + (k_2 - i_2)\, v_2 + i_1\, \mu_{i_1i_2}^1 + i_2\, \mu_{i_1i_2}^2\} \cdot Q_S(i_1, i_2) = (k_1 - i_1 + 1)\, \nu_1 \cdot Q_S(i_1 - 1, i_2)$

$+ (k_2 - i_2 + 1)\, \nu_2 \cdot Q_S(i_1, i_2 - 1) + (i_1 + 1)\, \mu_{i_1i_2}^1 \cdot Q_S(i_1 + 1, i_2) + i_2\, \mu_{i_1i_2}^2 \cdot Q_S(i_1, i_2 + 1)$

(11) $S < i_1S_1 + i_2S_2,\ 0 < i_1 \le k_1,\ 0 < i_2 \le k_2$

$\Re(i_1, i_2)$ denotes the set of the shortest routes from (0, 0) to (i_1, i_2), and there is the lattice point (j_1, j_2) on a route $u \in \Re(i_1, i_2)$ such as $j_1S_1 + j_2S_2 \le S$ and $S < (j_1+1)S_1 + j_2S_2$ or $S < j_1S_1 + (j_2+1)S_2$. When we denote the steady-state probability along the route u as $Q_S^{j_1j_2}(i_1, i_2)$, the following equilibrium equation holds.

$$\{(k_1-i_1)\nu_1 + (k_2-i_2)\nu_2 + j_1\mu^1_{i_1i_2} + i_2\mu^2_{i_1i_2}\}\,Q_S^{j_1j_2}(i_1, i_2)$$
$$= (k_1-i_1+1)\nu_1\,Q_S^{j_1j_2}(i_1-1, i_2) + (k_2-i_2+1)\nu_2\,Q_S^{j_1j_2}(i_1, i_2-1) + j_1\mu^1_{j_1j_2}\,Q_S^{j_1j_2}(i_1+1, i_2) + j_2\mu^2_{j_1j_2}\,Q_S^{j_1j_2}(i_1, i_2+1)$$

$Q_S(i_1, i_2)$ can be represented as $Q_S(i_1,i_2) = \sum_{\substack{u\in\Re(i_1,i_2)\\ (j_1,\, j_2)\, is\, on\, u}} Q_S^{j_1j_2}(i_1,i_2).$

For the state (i_1, i_2) of the Markov chain, when $i_1S_1 + i_2S \le S$, all jobs are in the central server model and executing CPU and I/O processing, and when $S < i_1S_1 + i_2S_2$, some jobs are in the system waiting queue 1 and waiting for a part of the memory to be released. The transition diagram of the two dimensional birth-death process is shown in Figure 3. However, the equilibrium equations do not have the product form solution. Therefore, some approximation is required to solve it.

When the model has a single job class, it can be described with a one dimensional birth-death process. Its transition diagram is shown in Figure 4, and the equilibrium equations are as follows:

(1) $k_1\,\nu_1 \cdot Q_S(0) = \mu_1{}^1 \cdot Q_S(1)$

(2) $\{(k_1 - i_1)\,\nu_1 + i_1\,\mu^1_{i_1}\} \cdot Q_S(i_1) = (k_1 - i_1 + 1)\,\nu_1 \cdot Q_S(i_1 - 1) + (i_1+1)\,\mu^1_{i_1+1} \cdot Q_S(i_1+1)$ $(0 < i_1S_1 \leqq S)$

(3) $\{(k_1 - i_1)\,\nu_1 + s_1\,\mu^1_{s_1}\} \cdot Q_S(i_1) = (k_1 - i_1 + 1)\,\nu_1 \cdot Q_S(i_1 - 1) + s_1\,\mu^1_{s_1} \cdot Q_S(i_1+1)$ $(S < i_1S_1 < k_1)$

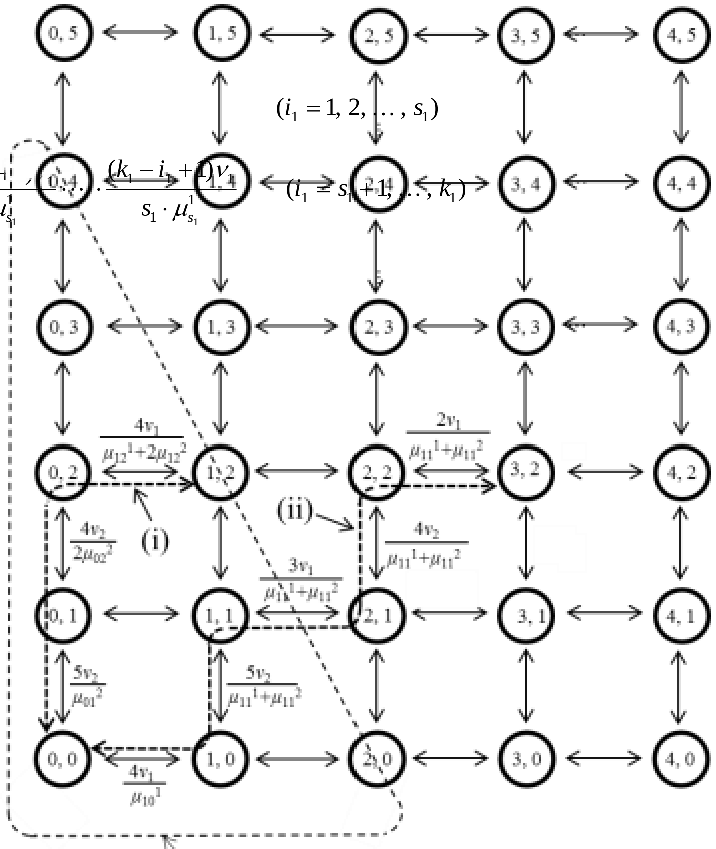
(4) $s_1\mu^1_{s_1} \cdot Q_S(k_1) = \nu_1 \cdot Q_S(k_1 - 1)$

The solutions for the equilibrium equations are described in the following product form.

$$Q_S(i_1) = \begin{cases} Q_S(0)\cdot\dfrac{k_1\nu_1}{1\cdot\mu^1_1}\cdot\dfrac{(k_1-1)\nu_1}{2\cdot\mu^1_2}\cdots\cdots\dfrac{(k_1-i_1+1)\nu_1}{i_1\cdot\mu^1_{i_1}} & (i_1 = 1, 2, \ldots, s_1) \\ Q_S(0)\cdot\displaystyle\prod_{i=1}^{s_1}\dfrac{(k_1-i+1)\nu_1}{i\cdot\mu^1_i}\cdot\dfrac{(k_1-s_1)\nu_1}{s_1\cdot\mu^1_{s_1}}\cdot\dfrac{(k_1-s_1+1)\nu_1}{s_1\cdot\mu^1_{s_1}}\cdots\cdot\dfrac{(k_1-i_1+1)\nu_1}{s_1\cdot\mu^1_{s_1}} & (i_1 = s_1+1, \ldots, k_1) \end{cases}$$

In this formula, for the state transition at i =1, 2, ..., s_1 − 1, multiply by factor $\dfrac{(k_1-i+1)\nu_1}{i\cdot\mu^1_i}$, while for the state transition at $i = s_1, s_1+1, \ldots, k_1$ multiply by factor $\dfrac{(k_1-i+1)\nu_1}{s_1\cdot\mu^1_{s_1}}$.

For two dimension case, we consider a shortest route from lattice point (0, 0) to (i_1, i_2) shown in Figure 5, and for the horizontal state transition at the lattice point (i_1, i_2) such as $i_1S_1 + i_2S_2 \le S$ on the route, multiply by factor $\dfrac{(k_1-i_1+1)\nu_1}{i_1\cdot\mu^1_{i_1i_2} + i_2\cdot\mu^2_{i_1i_2}}$, and multiply by factor $\dfrac{(k_2-i_2+1)\nu_2}{i_1\cdot\mu^1_{i_1i_2} + i_2\cdot\mu^2_{i_1i_2}}$ for the horizontal state transition, When the lattice point (i_1, i_2) such as S

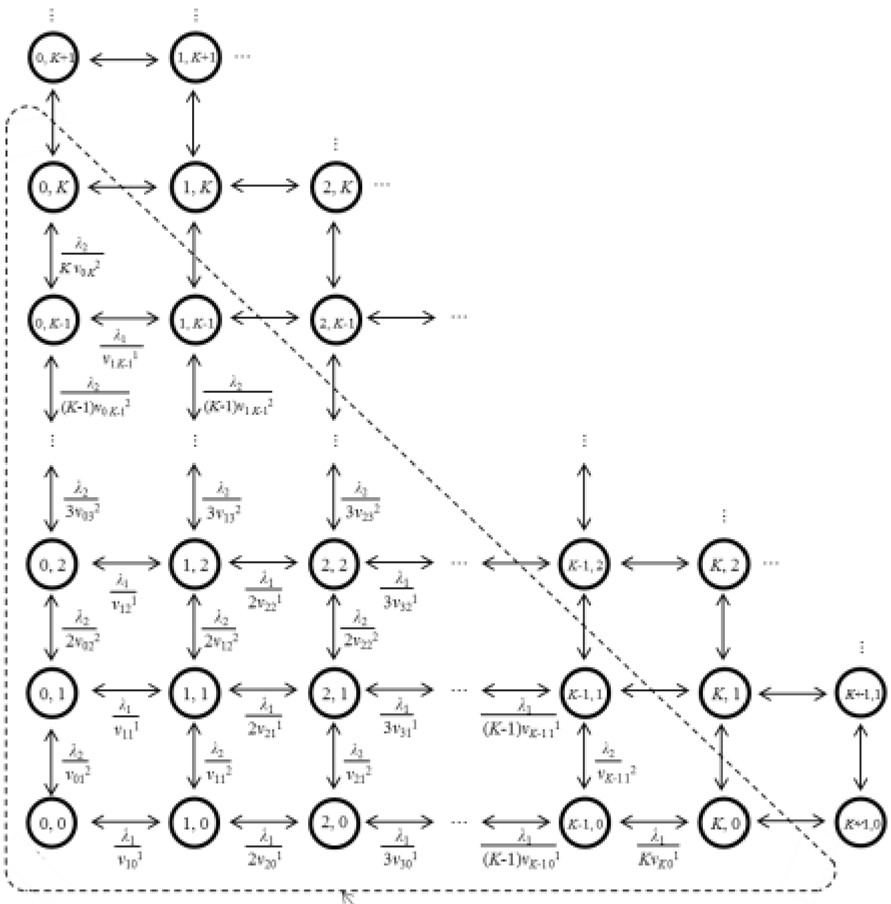

0, 5
1, 5
2, 5
3, 5
4, 5
0, 4
1, 4
2, 4
3, 4
4, 4
0, 3
1, 3
2, 3
3, 3
4, 3
0, 2
1, 2
2, 2
3, 2
4, 2
0, 1
1, 1
2, 1
3, 1
4, 1
0, 0
1, 0
2, 0
3, 0
4, 0
$\frac{4\nu_1}{\mu_{12}{}^1+2\mu_{12}{}^2}$
$\frac{2\nu_1}{\mu_{11}{}^1+\mu_{11}{}^2}$
(i)
(ii)
$\frac{4\nu_2}{2\mu_{02}{}^2}$
$\frac{4\nu_2}{\mu_{11}{}^1+\mu_{11}{}^2}$
$\frac{3\nu_1}{\mu_{1k}{}^1+\mu_{11}{}^2}$
$\frac{5\nu_2}{\mu_{01}{}^2}$
$\frac{5\nu_2}{\mu_{11}{}^1+\mu_{11}{}^2}$
$\frac{4\nu_1}{\mu_{10}{}^1}$

All jobs are in central server model.

Fig.5 State Transition diagram (two job classes) (k_1=4, k_2=5)

https://doi.org/10.17758/ERPUB.ER1217248

$< i_1S + i_2S_2$, for the state transition outside of the lattice point (j_1, j_2) such as $j_1S_1 + j_2S_2 < S$ and $S < (j_1+1)\,S_1 + j_2S_2$ or $j_1S_1 + (j_2+1)\,S_2 \le S$ on the route (between(j_1,j_2) and (i_1, i_2)), multiply by factor $\frac{(k_1 - j_1 + 1)\nu_1}{j_1 \cdot \mu^1_{j_1j_2} + j_2 \cdot \mu^2_{j_1j_2}}$ for the horizontal state transition or $\frac{(k_2 - j_2 + 1)\nu_2}{j_1 \cdot \mu^1_{j_1j_2} + j_2 \cdot \mu^2_{j_1j_2}}$ for the vertical state transition Thus, the coefficient of $Q_S(i_1, i_2)$ related to $Q_S(0, 0)$ is represented as the summation of the product along all the routes from (0, 0) to (i_1, i_2). For example, for the route from (0, 0) to (1, 2) when S=4, S_1=2, S_2=1, and k_1=5, k_2=4, which is the case of $i_1S_1 + i_2S_2 \le S$ ($2i_1 + i_2 \le 4$), the product along the route of broken line (i) in Figure 5 is $Q_S(0,0) \cdot \frac{5\nu_2}{\mu^2_{01}} \cdot \frac{4\nu_2}{2\mu^2_{02}} \times \frac{4\nu_1}{\mu^1_{12} + 2\mu^2_{12}}$. For the route from (0, 0) to (3, 2), which is the case of $S < i_1S_1 + i_2S_2$, $(4 < 2i_1 + i_2)$, the multiplication along the route (ii) is $Q_S(0,0) \cdot \frac{4\nu_1}{\mu^2_{01}} \frac{5\nu_2}{\mu^1_{11} + \mu^2_{11}} \times \frac{3\nu_1}{\mu^1_{11} + \mu^2_{11}} \cdot \frac{4\nu_2}{\mu^1_{11} + \mu^2_{11}} \cdot \frac{2\nu_1}{\mu^1_{11} + \mu^2_{11}}$.



All jobs obtain a memory.

Fig.6: State Transition diagram of outer level

Since there are multiple routes from (0, 0) to (i_1, i_2), the coefficient of $Q_S(i_1, i_2)$ related to $Q_S(0, 0)$ is approximately represented as the total of the products along all routes from (0, 0) to (i_1, i_2). Similar to the case above, we can approximate the state probability of a queuing network with multiple job classes when R >2.

3.2. Outer level

Figure 6 shows a state transition diagram of the outer level. The outer level is also expressed in a two-dimensional

birth-death process. In the dashed-line triangle of Figure 6, all jobs are in a state of acquiring a terminal. Unlike the inner level, it is an open queuing network where the number of jobs in each class can be infinite. The equilibrium equations with the steady-state probability $U_K(\boldsymbol{k}^*)= U_K(k_1, k_2)$, when the number of terminals is K and the number of occupied terminals is $\boldsymbol{k}^*=(k_1, k_2)$, are as follows (similar to the case with higher dimensions).

(1) k_1=0, k_2=0

$(\lambda_1+\lambda_2) \cdot U_K(0, 0) = \nu_{10}{}^1 \cdot U_K(1, 0) + \nu_{01}{}^2 \cdot U_K(0, 1)$

(2) k_1=1, 2, ... , $K-1$, k_2=0

$$(\lambda_1+\lambda_2+k_1\nu^1_{k_1 0})\cdot U_K(k_1, 0) = \lambda_1\cdot U_K(k_1-1, 0) + (k_1+1)\ \nu^1_{k_1+10}\cdot U_K(k_1+1, 0) + \nu^2_{k_1 1}\cdot U_K(k_1, 1)$$

(3) $k_1= K, K+1, ..., k_2=0$

$$(\lambda_1+\lambda_2+ K\nu^1_{K0})\cdot U_K(K, 0) = \lambda_1\cdot U_K(K-1, 0) + K\nu^1_{K0}\cdot U_K(K+1, 0) + \nu^2_{K1}\cdot U_K(K, 1)$$

(4) $k_1=0, k_2= 1, 2, ... , K-1$

$$(\lambda_1+\lambda_2+k_2\nu^2_{0k_2})\cdot Q_S(0, k_2) = \lambda_2\cdot U_K(0, k_2-1) + \nu^1_{1k_2}\cdot Q_S(1, k_2) + (k_2+1)\nu^2_{0k_2+1}\cdot U_K(0, k_2+1)$$

(5) $k_1=0, k_2= K, K+1, ...$

$$(\lambda_1+\lambda_2+K\nu^2_{0K})\cdot U_K\ (0, K) =\lambda_2\cdot U_K(0, K-1) + \nu^1_{1K}\cdot U_K(1, K) +K\nu^2_{0K}\cdot U_K(0, K+1)$$

(6) $k_1+k_2 \leqq K-1, k_1=1, 2, .. , K-2, k_2=1, 2,.., K-2$

$$(\lambda_1+\lambda_2+k_1\nu^1_{k_1k_2} +k_2\nu^2_{k_1k_2})\cdot U_K(k_1, k_2) =\lambda_1\cdot U_K(k_1-1, k_2) + \lambda_2\cdot U_K(k_1, k_2-1) + (k_1+1)\ \nu^1_{k_1+1k_2}\cdot U_K(k_1+1, k_2) +$$
$$(k_2+1)\ \nu^2_{k_1k_2+1}\cdot U_K(k_1, k_2+1)$$

(7) $k_1+k_2= K, k_1=1, 2, ... , K-1, k_2=1, 2, ... , K-1$

$$(\lambda_1+\lambda_2+k_1\nu^1_{k_1k_2} +k_2\nu^2_{k_1k_2})\cdot U_K(k_1, k_2) =\lambda_1\cdot U_K(k_1-1, k_2) + \lambda_2\cdot U_K(k_1, k_2-1) + k_1\nu^1_{k_1k_2}\cdot U_K(k_1+1, k_2) +$$
$$k_2\nu^2_{k_1k_2}\cdot U_K(k_1,\quad k_2+1)$$

(8) $k_1+k_2> K, k_1=1, 2, ... , k_2=1, 2, ...$

When the lattice point (l_1, l_2) such as $l_1+l_2=K$ *Solution* the shortest route l from (0, 0) to (k_1, k_2) and $U^l_K(k_1, k_2)$ is the state probability along the route l of the state (k_1, k_2),

$$(\lambda_1+\lambda_2+l_1\nu^1_{l_1l_2} + l_2\nu^2_{l_1l_2})\cdot U^l_K(k_1, k_2) = \lambda_1\cdot U^l_K(k_1-1, 0) + \lambda_2\cdot U^l_K(0, k_1-1) + l_1\nu^1_{l_1l_2}\cdot U^l_K(k_1+1, k_2) +$$
$$l_2\nu^2_{l_1l_2}\cdot U^l_K(k_1, k_1+1)).$$

(a) $k_1+k_2> K, k_1=1, 2, ... , K , k_2=1, 2, ... , K \Rightarrow U_K(k_1, k_2) = \sum_{l=K-k_2}^{k_1} U^l_K(k_1, k_2)$

(b) $k_1+k_2> K, k_1=K+1, K+2, ... , k_2=1, 2, ... , K \Rightarrow U_K(k_1, k_2) = \sum_{l=K-k_2}^{K} U^l_K(k_1, k_2)$

(c) $k_1+k_2> K, k_1=1, 2, ... , K , k_2=K+1, K+2, ... \Rightarrow U_K(k_1, k_2) = \sum_{l=0}^{k_1} U^l_K(k_1, k_2)$

(d) $k_1+k_2> K, k_1= K+1, K+2, ... , k_2=K+1, S+2, ... \Rightarrow U_K(k_1, k_2) = \sum_{l=0}^{K} U^l_K(n_1, n_2)$

4. Numerical Experiments

We evaluated the proposed approximation technique through numerical experiments. We used the following parameters.

1. Number of terminals: $K = 10$
2. Number of memory resources: $S = 5$
3. Number of I/O nodes: $M = 2$
4. Total service time at each node
   $\tau_{10}=1.0, \tau_{11}=\tau_{12}=1.0, \tau_{20}=1.0, \tau_{21}=\tau_{22}=0.5$,
   where τ_{rm} is the total service time of job class r at node m.
5. Think-time and arrival rate:
   - Figure 7, 8
     $(t_1, t_2) = (2.0, 1.0)$
     $(\lambda_1, \lambda_2) = (0.05, 0.1), (0.06, 0.1),…, (0.2, 0.1)$
   - Figure 9, 10
     $(t_1, t_2) = (1.0, 1.0), (1.1, 1.0),…, (2.5, 1.0)$
     $(\lambda_1, \lambda_2) = (0.2, 0.1)$

   where t_r is the mean think-time of job class r and λ_r is the arrival rate of job class r (r= 1, 2)

https://doi.org/10.17758/ERPUB.ER1217248

Figures 7 ~ 10 show the mean system response times and mean numbers of job as in the inner level of job classes 1 and 2 respectively, when the arrival rate λ_2 is fixed and λ_1 change from 0.05 to 0.2 by 0.01, and the think-time t_2 is fixed at 1.0 and t_1 change from 1.0 to 2.5 by 0.01. The mean system response time is the mean time from job arrival to departure from the network (that is the mean time of moving between terminal and the central server model plus the think-time at the terminal). Similar to the case of a single job class, the mean system response time for both job class increases monotonically in a convex curve. When the think-time of job class 1 increases, the mean number of jobs in the inner level increases linearly along increasing of the think-time of job class 1.

5. Conclusion

We proposed an approximation technique for evaluating the performance of a computer system with input to terminals using a queuing network technique and analyzed its performance measures through numerical experiments. The concept of the approximation is based on separately analyzing the inner level (CPU, I/O equipment, and memory) and the outer level (terminals and communication lines). The numerical experiments clarified the characteristics of the system response time.

In the future we plan to examine the accuracy of the proposed approximation technique by comparing it with exact solutions or simulation results.

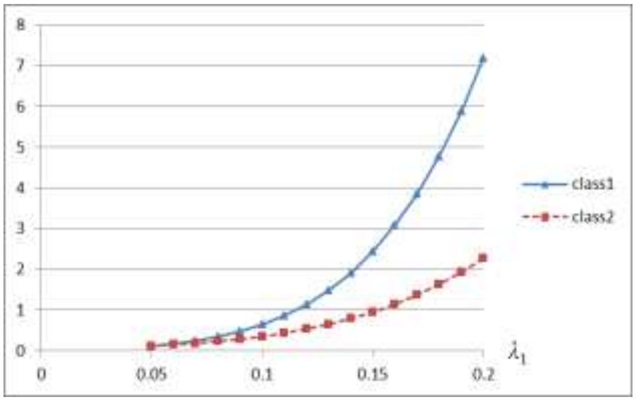


Fig. 7: Mean number of jobs in the central server modal

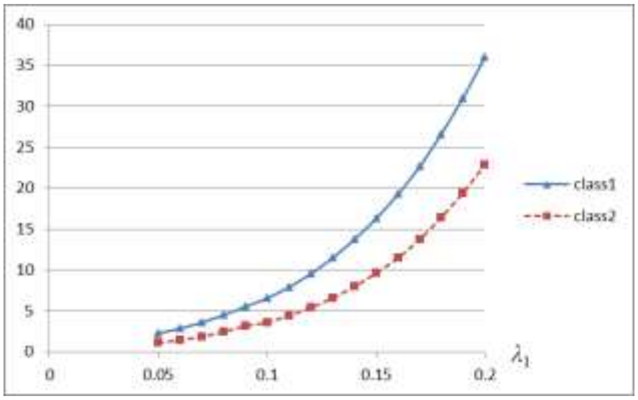


Fig.8: Mean system response time

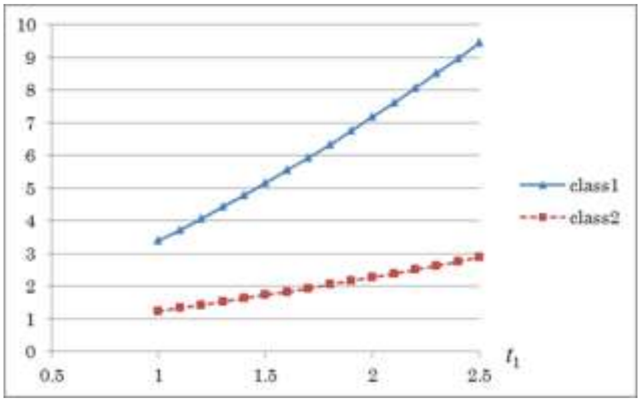


Fig. 9: Mean number of jobs in the central server modal

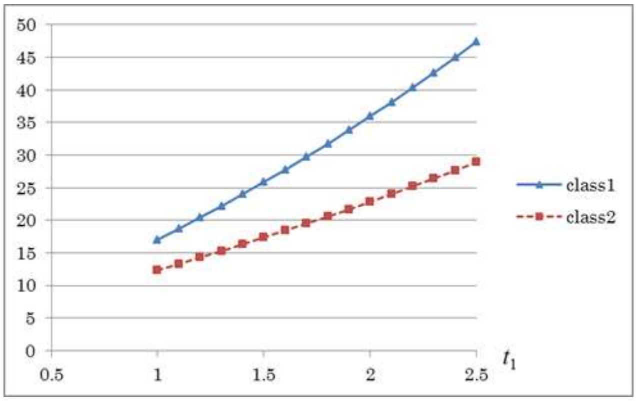


Fig.10: Mean system response time

6. References

[1] F. Baskett, K. M. Chandy, R. R. Muntz and F. G. Palacious, “Open, Closed, and Mixed Net-works of Queues with Different Classes of Customers," J. ACM, Vol.22, No.2, pp.248-260, April 1975.
https://doi.org/10.1145/321879.321887

[2] H. Kobayashi, “Modeling and Analysis," Addison-Wesley Publishing Company, Inc. 1978.

[3] J. A. Rolia and K. C. Sevcik, “The Method of Layers," IEEE Trans. on Software Engineering, Vol.21, No.8, pp.689-700, Aug. 1995.
https://doi.org/10.1109/32.403785

[4] T. Kinoshita and Y. Takahashi, “A Queuing Network Modeling and Performance Evaluation Method for Computer Systems with Resource Requirement," IEICE D-I, Vol. J 82-D-I, No.6, pp.701-710, Jun. 1999

[5] T. Kinoshita and X. Gao, “Queuing Network Approximation Technique for Evaluating Per-formance of Computer Systems with Memory Resources,” PDPTA2010, pp.640-646, July 2010

[6] O. E. Oguike, M. N. Agu and S.C. Echezona, ”Modeling Variation of Waiting Time of Dis-tributed Memory Heterogeneous Parallel Computer System Using Recursive Models,” Interna-tional Journal of Soft Computing and Engineering, vol. 2, Issue 6, Jan 2013

[7] A. Gandhi, S. Doroudi, M. Harchol-Balter and A. Scheller-Wolf, “Exact Analysis of the M/M/k/setup Class of Markov Chains via Recursive Renewal Reward,” SIGMETRICS’13, pp.153-166, June 2013

[8] M. Takaya, M. Ogiwara, N. Matrazali, C. Itaba, I. Koike, T. Kinoshita, “Queuing Network Approximation Technique for Evaluating Performance of Computer Systems with Memory Re-source used by Multiple job types,” CSC2014, pp.41-46, July 2014

[9] K. Katsumata, M. Noorafiza, S. Ito, I. Koike, T. Kinoshita, “Queuing Network Approximation Technique for Evaluating Performance of Computer Systems Acquiring Difference Memory with Finite Input Source,” CATA2016, pp.43-49, April 2016