Approximate Performance Evaluation Method of Computer Systems with Hybrid Input Source

Itaru Koike, Hikari Yoshii, Hozomi Miyamoto, Mayuko Hirose, and Toshiyuki Kinoshita

***Abstract*—Queuing network techniques are effective for evaluating the performance of computer systems. We considered a queuing network with two input sources, one is a finite input source and the other is an open input source. We call this as hybrid input source. In the finite input source, the finite number of terminals exists and a job is dedicated to its own terminal. After a think-time at the terminal, the job moves to the server, acquires a part of the memory, and executes CPU and Input / Output (I/O) processing. When the job completes at the CPU and I/O processing, it releases the memory and goes back to its own terminal. On the other hand, in the open input source, the job arrives at the server randomly from outside, acquires a part of the memory, and executes CPU and I/O processing, and goes back to the outside after releasing the memory. However, the queuing network model with memory resource has no product form solution and cannot calculate the exact solutions.**

We proposed an approximate queuing network technique to calculate the performance measures of computer systems with hybrid input source in which multiple types of jobs exist. This technique involves dividing the queuing 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 approximately calculate the performance measures of the network. We evaluated the proposed approximation technique by using numerical experiments and a Monte Carlo simulation, and clarified the characteristics of the system response time and the accuracy of the approximation.

***Index Terms*—Performance evaluation, queuing network, central server model, finite input source, open input source.**

I. 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], [2]. With this solution, we can easily calculate the performance measures of computer systems, for example the busy ratio of hardware, the job response time, and so on.

However, when the exclusion controls are active or when a memory resource exists, the queuing network does not have the product form solution. When the queuing network that has no 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 to calculate an exact solution. 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 unknown quantities in the equilibrium equations is equal to the number of states of the queuing network, the number of unknown quantities in the equilibrium equations also drastically increases. Therefore, we cannot perform calculation of the exact solution of the queuing network numerically.

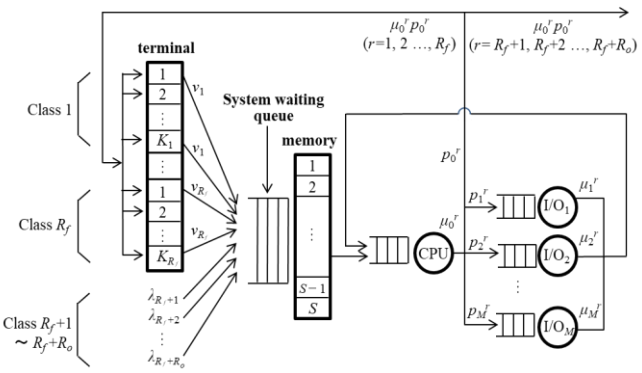


Fig. 1. Central server model in hybrid input source.

Here we consider a queuing network with two input sources, one is a finite input source and the other is an open input source (Fig. 1). We call this as hybrid input source. In a finite input source, the finite number of terminals exist in the network and a job is dedicated to its own terminal. After a think-time at the terminal, the job moves to the server and acquires a part of the memory, and executes 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 original terminal. In an open input source, the job arrives at the server randomly from outside, acquires a part of the memory, and executes the CPU and I/O processing. When the job completes the CPU and I/O processing, it releases the memory and goes back to the outside. The finite input source assumes online real-time processing in the computer system, and the open input source assumes batch processing.

Since a job executes CPU and I/O processing occupying the memory, the memory can be considered as a secondary resource for the CPU and I/O equipment in inner level. Generally, when a queuing network includes a secondary resource, it does not have product form solutions and an approximation technique is required to analyze the network. We have proposed here an approximation technique for

Manuscript received September 3, 2018; revised October 18, 2018.

Itaru Koike, Hikari Yoshii, Nozomi Miyamoto, Mayuko Hirose and Toshiyuki Kinoshita are with School of Computer Science, Tokyo University of Technology, Hachioji Tokyo, 192-0982, Japan (e-mail: amirosu.hokuto@gmail.com, c01145528e @edu.teu.ac.jp, c0114508f8@edu.teu.ac.jp, t.kisaragi3so@gmail.com, knoshi@stf.teu.ac. jp).

calculating the performance measures of computer systems with hybrid input source. We previously reported the results for computer systems with memory resource in open input source, in which jobs arrive from and depart to the outside of the system.

In this paper, we consider the hybrid input source model. We already reported the open input source model and the finite input source model in [3] and [4] respectively, and the input to terminals model in [5]. In order to prevent the number of states of the Markov chain from increasing, we divide the model into two levels, one is outer level that includes the outside of the system, the terminals, and communication lines, and the other one is inner level that includes CPU, I/O equipment and memory resources (Fig. 2). Similar to [3], [4] and [5], multiple types of jobs exist in the inner and the outer level. When there is a single job class, both the inner and outer level has a product form solution. However, if there are multiple job classes with a finite input source and open input source, the inner level has no product form solution. Therefore, an approximation is needed to analyze the inner level.

Dividing the model into two levels is one of two-layer queuing network techniques [6], [7]. Our proposed technique is also a two-layer technique for computer systems with hybrid input source. Meanwhile, the Markov chain involving two dimensional state transition similar to our proposed model was discussed in [8].

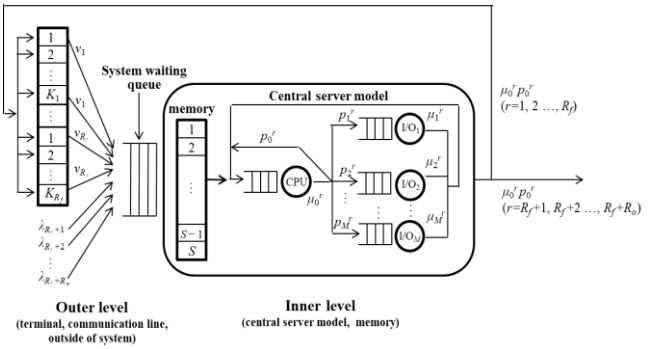


Fig. 2. Concept of approximation.

II. 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_f job classes of finite input source and R_o job classes of open input source exist, and job classes of finite input source are numbered $r = 1, 2, \ldots, R_f$ by affixing r and job classes of open input source are numbered $r = R_f+1, R_f+2, \ldots, R_f+R_o$ by also affixing r. The inner level consists of a single CPU node and multiple I/O nodes. We denote M as the number of I/O nodes. The I/O nodes are numbered $m = 1, 2, \ldots, M$ 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 $\mu_0{}^r$ and the service rate of job class r at an I/O node m is $\mu_m{}^r$. 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 or goes to the outside of the system. The selection probability of I/O node m of job class r is $p_m{}^r$, and the selection probability of CPU node or the completion probability of job class r is $p_0{}^r$. Therefore $\sum_{m=0}^{M} p_m^r = 1 (r = 1,2,\cdots,R_f,R_f+1,\ldots,R_f+R_o)$.

Memory resources are added to this central server model in the inner level. We denote S as the number of memory resources.

In outer level, a job of finite input source stays at the terminal for short while. The staying time is called "think-time" of a job. The think-time is mutually independent random variable subject to common exponential distribution with parameter v_r of job class r (r = 1, 2, …, R_f). This v_r equals to job departure rate from the terminal. After the think-time, the job moves to the inner level. Meanwhile a job of open input source arrives in the inner level from the outside at random at arrival rate λ_r of job class r (= R_f +1, R_f +2, …, R_f +R_o). When a job arrives in the inner level, it requests and acquires a part of the memory resources before entering the central server model. If all the parts of the memory are occupied, the job joins the system waiting queue and waits for a part of 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 model and goes back to its own terminal or goes to the outside. Since the job has to acquire a part of memory before entering the central server model, the number of jobs occupying a memory is always equal to the number of jobs in the central server model. Therefore, at most S jobs can execute CPU and I/O processing at the same time. That is, the maximum job multiplicity in the central server model is S. When the number of jobs of job class r in the central server model is denoted by n_r, $\sum_{r=1}^{R_f+R_o} n_r \le S$.

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 (Fig. 2). In this model, when "CPU → CPU transition" occurs, 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 mean job response time can be considered as a job lifetime.

III. Approximation Model

To obtain the exact solution of the central server model with hybrid input source, we have to describe the entire model with a single Markov chain for each job class. However, this causes the number of states of the Markov chain to drastically increase when the number of jobs and the number of nodes in the network increase. By dividing the network into two levels, and describing each level with two Markov chains, we can prevent the number of states of the model from increasing (Fig. 2). We set the following notations.

R_f : number of job classes of finite input source

R_o: number of job classes of open input source

t_r: mean think-time of jobs in job class r (r = 1, 2, ... , R_f)

v_r: departure rate from the terminal of job class r (r = 1, 2, ... , R_f)

λ_r: arrival rate of job class r ($r = R_f+1, R_f+2, \dots, R_f+R_o$)

τ_{rm}: total mean service time at node-m of jobs in job class r (r =1, 2, ... , R_f+R_o; m =0, 1, ... , M)

n_{rm}: number of jobs in job class r at node-m (r =1, 2, ... , R_f+R_o; m =0, 1, ... , M)

$\boldsymbol{n} = (n_1, n_2, \dots, n_{Rf}, n_{Rf+1}, \dots, n_{Rf+Ro})$: vector of number of jobs

(n_r =0, 1, 2, ..., K_r for r = 1, 2, ... , R_f ; =n_r =0, 1, 2, 3, ... for $r = R_f+1, R_f+2, \dots, R_f+R_o$)

$\boldsymbol{n}^* = (n_{10}, n_{11}, \dots, n_{1M}, n_{20}, n_{21}, \dots, n_{2M}, \dots, n_{Rf+Ro0}, n_{Rf+Ro1}, \dots, n_{Rf+RoM})$: state vector of the central server model

$$\boldsymbol{F}(\boldsymbol{n}) = \{\boldsymbol{n}^* \mid \sum_{m=0}^{M} n_{rm} = n_r,\ n_{rm} \geq 0\ (m=0, 1, \dots, M)\}$$

(r = 1, 2, … , R_f+R_o; $n_1+n_2+\dots+n_{Rf+Ro} \leqq S$): set of all feasible states of the central server model when the number of jobs in job class r is n_r

$P_s(\boldsymbol{n}^*)$: steady-state probability of state $\boldsymbol{n}^*$

$T_{\boldsymbol{n}}{}^r$: mean job response time in the central server model of job class r when the vector of number of jobs is $\boldsymbol{n}$

$\mu_{\boldsymbol{n}}{}^r$: service rate from the central server model of job class r when the vector of number of jobs is $\boldsymbol{n}$

T^r: system response time of job class r

Since the central server model in 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 $P_s(\boldsymbol{n}^*)$ is represented by the following formula.

$$P_s(n^*) = \frac{\prod_{r=1}^{R}\prod_{m=0}^{M} \tau_{rm}{}^{n_{rm}}}{\varphi(n_1, n_2, \cdots, n_R, M)}$$

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 (r =1, 2, ... , R_f+R_o). From these steady-state probabilities, we can calculate the mean job response time $T_{\boldsymbol{n}}{}^r$ of job class r as $T_n{}^r = \frac{n_r \cdot \varphi(n_1, \cdots, n_r, \cdots, n_R, M)}{\varphi(n_1, \cdots, n_r - 1, \cdots, n_R, M)}$, when the number of jobs is n_r [1], [2].

The memory resource 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 customers in 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 $T_{\boldsymbol{n}}{}^r$ of job class r (=1, 2, ... , R_f+R_o) when the vector of number of jobs is $\boldsymbol{n} = (n_1, n_2, \dots, n_r)$ is equal to the mean time while the memory is occupied. Since the service rate of job class r from the central server model $\mu_{\boldsymbol{n}}{}^r$ is denoted as $\mu_{\boldsymbol{n}}{}^r = \frac{1}{T_n^r}$, $\mu_{\boldsymbol{n}}{}^r$ depends on the number of jobs in the central server model n_r. The state transition of the M/M/S queuing model with two job classes (one is for finite input source and the other is for open input source) is shown in Fig. 3, where the service 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 $Q_S(\boldsymbol{n}) = Q_S(n_1, n_2)$, when the maximum number of jobs in the central server model is S and the vector of number of jobs in the central server model is $\boldsymbol{n} = (n_1, n_2)$, are as follows (similar to the case with higher dimensions).

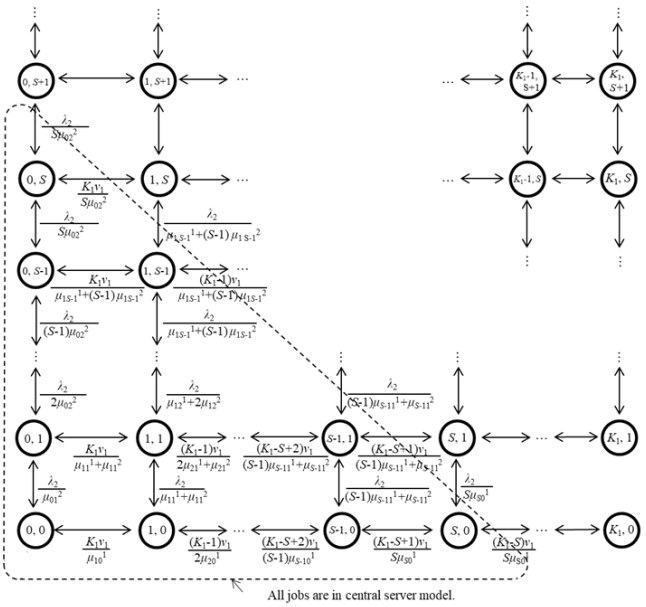


Fig. 3. State transition diagram (two job classes).

a) n_1=0, n_2=0

$$(K_1v_1 + \lambda_2) \cdot Q_S(0, 0) = \mu_{10}{}^1 \cdot Q_S(1, 0) + \mu_{01}{}^2 \cdot Q_S(0, 1)$$

b) n_1=1, 2, ... , $S-1$, n_2=0

$$\{(K_1 - n_1)\, v_1 + \lambda_2 + n_1 \mu^1_{n_1 0}\} \cdot Q_S(n_1, 0) = (K_1 - n_1 + 1)\, v_1 \cdot Q_S(n_1 - 1, 0) + (n_1+1)\, \mu^1_{n_1+10} \cdot Q_S(n_1+1, 0) + \mu^2_{n_1 1} \cdot Q_S(n_1, 1)$$

(c) n_1= S, S+1, ..., K_1-1, n_2=0

$$\{(K_1 - n_1)\, v_1 + \lambda_2 + S\, \mu^1_{S0}\} \cdot Q_S(n_1, 0) = (K_1 - n_1 + 1)\, v_1 \cdot Q_S(n_1 - 1, 0) + S\, \mu^1_{S0} \cdot Q_S(n_1+1, 0) + \mu^2_{n_1 1} \cdot Q_S(n_1, 1)$$

(d) n_1=0, n_2= 1, 2, ... , $S-1$

$$\{K_1v_1 + \lambda_2 + n_2\, \mu^2_{0n_2}\} \cdot Q_S(0, n_2) = \lambda_2 \cdot Q_S(0, n_2 - 1) + \mu^1_{1n_2} \cdot Q_S(1, n_2) + (n_2+1)\, \mu^2_{0n_2+1} \cdot Q_S(0, n_2+1)$$

(e) n_1=0, n_2= S, S+1, S+2, ...

$$\{K_1v_1 + \lambda_2 + S\, \mu^2_{0S}\} \cdot Q_S(0, n_2) = \lambda_2 \cdot Q_S(0, n_2 - 1) + \mu^1_{1n_2} \cdot Q_S(1, n_2) + S\, \mu^2_{0S} \cdot Q_S(0, n_2+1)$$

(f) $n_1+n_2 \leqq S-1$, n_1=1, 2, …, $S-2$, n_2=1, 2, …, $S-2$

$$\{(K_1 - n_1)\, v_1 + \lambda_2 + n_1\, \mu^1_{n_1 n_2} + n_2\, \mu^2_{n_1 n_2}\} \cdot Q_S(n_1, n_2) = (K_1 - n_1 + 1)\, v_1 \cdot Q_S(n_1 - 1, n_2) + \lambda_2 \cdot Q_S(n_1, n_2 - 1) + (n_1+1)\, \mu^1_{n_1+1 n_2} \cdot Q_S(n_1+1, n_2) + (n_2+1)\, \mu^2_{n_1 n_2+1} \cdot Q_S(n_1, n_2+1)$$

(g) $n_1+n_2 = S$, n_1=1, 2, ... , $S-1$, n_2=1, 2, ... , $S-1$

$$\{(K_1-n_1)\,\nu_1+\lambda_2+n_1\,\mu^1_{n_1n_2}+n_2\,\mu^2_{n_1n_2}\}\cdot Q_S(n_1,n_2)=(K_1-n_1+1)\,\nu_1\cdot Q_S(n_1-1,n_2)+\lambda_2\cdot Q_S(n_1,n_2-1)+n_1\,\mu^1_{n_1n_2}\cdot Q_S(n_1+1,n_2)+n_2\,\mu^2_{n_1n_2}\cdot Q_S(n_1,n_2+1)$$

(h) $n_1+n_2 > S$, n_1=1, 2, ... , K_1 , n_2=1, 2, 3, ..

When the lattice point (m_1, m_2) such as $m_1+m_2=S$ (m_1, m_2 = 0, 1, ⋯, S) is on the shortest route from (0, 0) to (n_1, n_2), and $Q_S^{m_1m_2}(n_1, n_2)$ is the steady-state probability along with the route.

$$\{(K_1-n_1)\,\nu_1+\lambda_2+m_1\,\mu^1_{m_1m_2}+m_2\,\mu^2_{m_1m_2}\}\cdot Q_S^{m_1m_2}(n_1,n_2)=(K_1-n_1+1)\,\nu_1\cdot Q_S^{m_1m_2}(n_1-1,n_2)+\lambda_2\cdot Q_S^{m_1m_2}(n_1,n_2-1)+m_1\,\mu^1_{m_1m_2}\cdot Q_S^{m_1m_2}(n_1+1,n_2)+m_2\,\mu^2_{m_1m_2}\cdot Q_S^{m_1m_2}(n_1,n_2+1)$$

(h-1) n_1=1, 2, ... , S , $n_2=S-n_1+1, S-n_1+2, \ldots, S$

$$\Rightarrow Q_S(n_1,n_2)=\sum_{m_1=S-n_2}^{n_1} Q_S^{m_1m_2}(n_1,n_2)$$

(h-2) n_1=S+1, S+2, ... , K_1, n_2=1, 2, ... , S

$$\Rightarrow Q_S(n_1,n_2)=\sum_{m_1=S-n_2}^{S} Q_S^{m_1m_2}(n_1,n_2)$$

(h-3) n_1=1, 2, ... , S, n_2=S+1, S+2, S+3, ...

$$\Rightarrow Q_S(n_1,n_2)=\sum_{m_1=0}^{n_1} Q_S^{m_1m_2}(n_1,n_2)$$

(h-4) n_1= S+1, S+2, ... , K_1, n_2=S+1, S+2, ...

$$\Rightarrow Q_S(n_1,n_2)=\sum_{m_1=0}^{S} Q_S^{m_1m_2}(n_1,n_2)$$

For the state (n_1, n_2) of the Markov chain, when $n_1+n_2 \leqq S$, all jobs are in the central server model and executing CPU and I/O processing, and when $n_1+n_2 > S$, n_1+n_2-S jobs are in the system waiting queue and waiting for a part of the memory resources to be released. The transition diagram of the two dimensional birth-death process is shown in Fig. 3. However, since the equilibrium equation does not have the product form solution, 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 Fig. 4, and the equilibrium equation is as follows:

(a) Finite input source

(i) n_1 =0

$$K_1\,\nu_1\cdot Q_S(0)=\mu_1{}^1\cdot Q_S(1)$$

(ii) n_1 =1, 2, …, $S-1$

$$\{(K_1-n_1)\nu_1+n_1\,\mu^1_{n_1}\}\cdot Q_S(n_1)=(K_1-n_1+1)\,\nu_1\cdot Q_S(n_1-1)+(n_1+1)\,\mu^1_{n_1+1}\cdot Q_S(n_1+1)$$

(iii) n_1 =S, S+1, … , K_1-1

$$\{(K_1-n_1)\nu_1+S\,\mu^1_S\}\cdot Q_S(n_1)=(K_1-n_1+1)\,\nu_1\cdot Q_S(n_1-1)+S\,\mu^1_S\cdot Q_S(n_1+1)$$

(iv) n_1 =K_1

$$S\,\mu^1_S\cdot Q_S(K_1)=\nu_1\cdot Q_S(K_1-1)$$

(b) Open input source

(i) n_2 =0

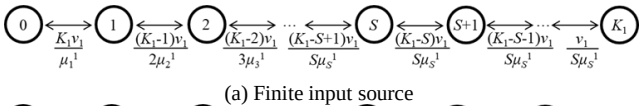
$$\lambda_2\cdot Q_S(0)=\mu_1{}^2\cdot Q_S(1)$$

(ii) n_2 =1, 2, …, $S-1$

$$(\lambda_2+n_2\,\mu^2_{n_2})\cdot Q_S(n_2)=\lambda_2\cdot Q_S(n_2-1)+(n_2+1)\,\mu^2_{n_2+1}\cdot Q_S(n_2+1)$$

(iii) n_2 = S, S+1, S+2, …

$$(\lambda_2+S\,\mu^2_S)\cdot Q_S(n_2)=\lambda_2\cdot Q_S(n_2-1)+S\,\mu^2_S\cdot Q_S(n_2+1)$$



(a) Finite input source

(b) Open input source

Fig. 4. State transition diagram (single job class).

Solutions for the equilibrium equation are in the following product form.

(a) Finite input source

$$Q_S(n_1)=\begin{cases} Q_S(0)\cdot\prod_{i=1}^{n_1}\dfrac{(K_1-i+1)\nu_1}{i\cdot\mu_i^1} & (n_1=1,2,\ldots,S-1)\\ Q_S(0)\cdot\prod_{i=1}^{S-1}\dfrac{(K_1-i+1)\nu_1}{i\cdot\mu_i^1}\cdot\prod_{i=S}^{n_1}\dfrac{(K_1-i+1)\nu_1}{S\cdot\mu_S^1} & (n_1=S,S+1,\cdot\cdot,K_1)\end{cases}$$

(b) Open input source

$$Q_S(n_2)=\begin{cases} Q_S(0)\cdot\prod_{i=1}^{n_2}\dfrac{\lambda_2}{i\cdot\mu_i^2} & (n_2=1,2,\ldots,S-1)\\ Q_S(0)\cdot\prod_{i=1}^{S-1}\dfrac{\lambda_2}{i\cdot\mu_i^2}\cdot\prod_{i=S}^{n_2}\dfrac{\lambda_2}{S\cdot\mu_S^2} & (n_2=S,S+1,S+2,\ldots)\end{cases}$$

In finite input source, for the state transition at i =1, 2, ..., $S-1$, multiply by factor $\dfrac{(K_1-i+1)\nu_1}{i\cdot\mu_i^1}$, while for the state transition at i = S, S+1, ..., K_1, multiply by factor $\dfrac{(K_1-i+1)\nu_1}{S\cdot\mu_S^1}$, and in open input source, for the state transition at i =1, 2, ..., $S-1$, multiply by factor $\dfrac{\lambda_2}{i\cdot\mu_i^2}$, while for the state transition at i = S, S+1, S+2, ... multiply by factor $\dfrac{\lambda_2}{S\cdot\mu_S^2}$. For two dimension case, we consider a route from lattice point (0, 0) to (n_1, n_2) shown in Fig. 5, and for the horizontal state transition at the lattice point (i_1, i_2) such as $i_1+i_2 \leqq S$ 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 $\dfrac{\lambda_2}{i_1\cdot\mu^1_{i_1i_2}+i_2\cdot\mu^2_{i_1i_2}}$ for the vertical state transition. When the lattice point (i_1, i_2) such as $i_1+i_2 > S$, for the state transition outside of the lattice point (m_1, m_2) such as $m_1+m_2=S$ on the route (between (m_1, m_2) and (i_1, i_2)), multiply by factor $\dfrac{(K_1-m_1+1)\nu_1}{m_1\cdot\mu^1_{m_1m_2}+m_2\cdot\mu^2_{m_1m_2}}$ or $\dfrac{\lambda_2}{m_1\cdot\mu^1_{m_1m_2}+m_2\cdot\mu^2_{m_1m_2}}$. Thus, the coefficient of $Q_S(n_1, n_2)$ related to $Q_S(0, 0)$ is

represented as the summation of the multiplication based on all the routes from (0, 0) to (n_1, n_2). For example, for the route from (0, 0) to

(1, 2) when S=3, and K_1=5, which is the case of $n_1+n_2 \leqq S$, the multiplication along the route of broken line (i) in Fig. 5 is $Q_S(0,0)\cdot\frac{\lambda_2}{\mu_{01}^2}\cdot\frac{\lambda_2}{2\mu_{02}^2}\cdot\frac{5\nu_1}{\mu_{12}^1+2\mu_{12}^2}$. For the route from (0, 0) to (4,2), which is the case of $n_1+n_2 > S$, the multiplication along the route (ii) is $Q_S(0,0)\cdot\frac{5\nu_1}{\mu_{01}^2}\cdot\frac{\lambda_2}{\mu_{11}^1+\mu_{11}^2}\cdot\frac{4\nu_1}{2\mu_{21}^1+\mu_{21}^2}\cdot\frac{\lambda_2}{2\mu_{21}^1+\mu_{21}^2}\times\frac{2\nu_1}{2\mu_{21}^1+\mu_{21}^2}$. Since there are multiple routes from (0, 0) to

(n_1, n_2), the coefficient of $Q_S(n_1, n_2)$ related to $Q_S(0, 0)$ is approximately represented as the total of the multiplication based on all routes. Similarly, to the case above, we can approximately calculate the state probability of a queuing net-work with multiple job classes when R_f >1 or R_o >1.

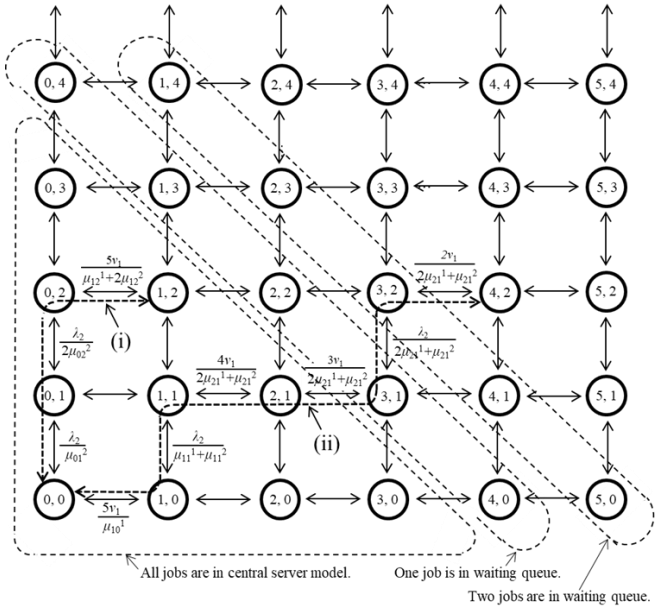


Fig. 5. Calculation state probability for two job classes.

IV. Numerical Experoments

We evaluated the proposed approximation technique through numerical experiments and compared it with the simulation results. We used the following parameters.

1) Number of terminals: K_1 = 3 ~ 20 or K_1 = 5 ~ 20
2) Number of memory resources: S = 3 or 5
3) Think-time: t_1 = 10
4) Arrival rate: λ_2=0.02 ~ 0.4
5) Number of I/O nodes: M = 2
6) Total service time at each node

$$\tau_{10}=1.0,\ \tau_{11}=\tau_{12}=0.5,$$

$$\tau_{20}=1.0,\ \tau_{21}=\tau_{22}=1.0,$$

where τ_{rm} is the total service time of job class r at node m.

Fig. 6 to Fig. 9 show the mean system response times of job classes 1 and 2 calculated by the proposed method and the simulation, when S is fixed at 3 or 5. Fig. 6 and Fig. 7 show the cases of λ_2=0.2, K_1=S, S+1, …, 20, and Fig. 8 and Fig. 9 show the cases of K_1=6, λ_2=0.02, 0.04, …, 0.4. The mean system response time is the mean time from job arrival to departure from the inner level, that is the mean time from departure from the terminal to coming back to the terminal in the finite input source and that is the mean time from arrival to departure to the outside of the system in the open input source. Similar to the case of a single job class, the mean system response time for both job class monotonically increases and the mean system response time for the finite input source draws a convex curve. When the number of terminals K_1 of job class 1 increases and the arrival ate λ_2 of job class 2 is fixed (the only traffic of job class 1 increases), both job class 1 and job class 2 mean response times increase. This is because of the entire central server model is more crowded by increasing traffic of the job class 1. Similarly, when the number of terminals K_1 of job class 1 is fixed and the arrival rate λ_2 of job class 2 increases, both response times monotonically increase. We can see that the mean response time of job class 1 increases more rapidly than job class 2 in heavier traffic range. This reason can be presumed that the behavior of the mean system response time of job class 1 in the heavier traffic range is approximately linear to the number of jobs in the central server model.

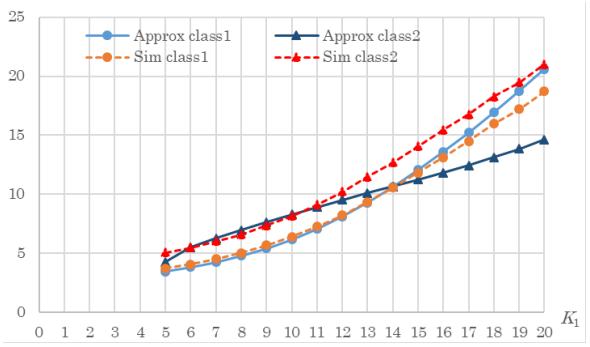


Fig. 6. Mean system response time (S=3, t_1=10, λ_2=0.2).

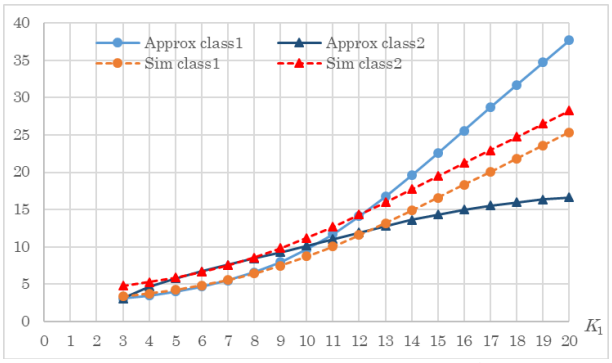


Fig. 7. Mean system response time (S=5, t_1=10, λ_2=0.2).

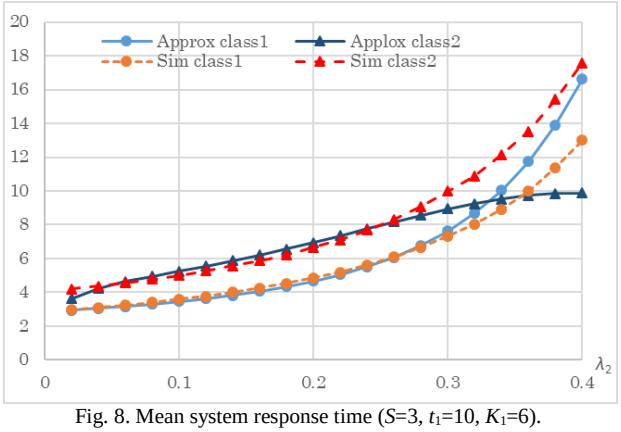


Fig. 8. Mean system response time (S=3, t_1=10, K_1=6).

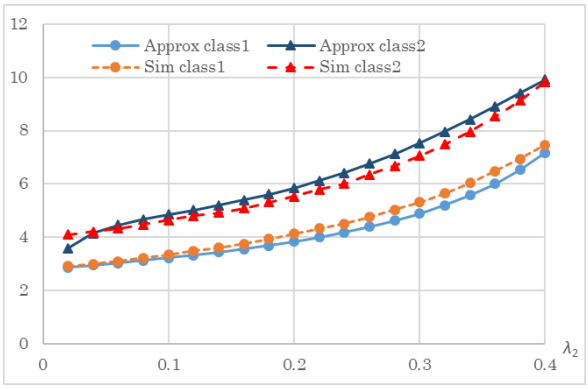


Fig. 9. Mean system response time (S=5, t_1=10, K_1=6).

Compared to the simulation, the results of job class 1 are well consistent with the simulation results in Fig. 6 and Fig. 7, and the results of job class 2 are well consistent with the simulation results in Fig. 8 and Fig. 9, this means that the results of the classes that controlled their traffic are better consistent. When the results are indirectly changed due to the influence of another class, it is considered that the accuracy of the approximation can be deteriorated. In Fig. 9, both results of job class 1 and 2 are well consistent with the simulation results. Fig. 9 shows the case that the memory load is the smallest.

V. Conclusion

We proposed an approximation technique for evaluating the performance of computer systems in hybrid input source using a queuing network 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 outer level (terminals, communication lines and the outside of the system). The numerical experiments clarified the characteristics of the system response time.

In the future, we are planning to analyze the queuing network model that a job arrives from the outside to terminals instead of the hybrid input source.

References

[1] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacious, "Open, closed, and mixed networks of queues with different classes of customers," *J. ACM*, vol. 22, no. 2, pp. 248-260, April 1975.
[2] H. Kobayashi, *Modeling and Analysis*, Addison-Wesley Publishing Company, Inc. 1978.
[3] M. Takaya, M. Ogiwara, N. Matrazali, C. Itaba, I. Koike, and T. Kinoshita, "Queuing network approximation technique for evaluating performance of computer systems with memory resource used by multiple job types," in *Proc. CSC2014*, July 2014, pp. 41-46.
[4] M. Hirose, M. Shiratori, N. Matrazali, R. Tsuboi, I. Koike, and T. Kinoshita, "Queuing network approximation technique for evaluating performance of computer systems with finite input source," in *Proc. CSC2015*, July 2015, pp. 9-15.
[5] H. Yoshii, N. Miyamoto, MZ. Nurshafiqah, D. Miyake, I. Koike, and T. Kinoshita, "Queuing network approximation technique for evaluating performance of computer systems with input to terminals," in *Proc. CCCME2017*, Dec. 2017
[6] T. Kurasugi and I. Kino, "Approximation method for two-layer queueing models," *Performance Evaluation*, pp. 55-70, 1999.
[7] 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.
[8] 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," in *Proc. SIGMETRICS'13*, June 2013, pp. 153-166.

Itaru Koike graduated from Tokyo University of Technology in 2010. He received his master of engineering from Tokyo University of Technology in 2013. He is an assistant in Tokyo University of Technology. Mr. Koike is majoring in operating system and computer simulation technique.

Hikari Yoshii is the 4th year student of Kinoshita laboratory in Tokyo University of Technology. She is researching for computer and network system performance. Ms. Yoshii is majoring in queuing theory for evaluating computer system performance.

Nozomi Miyamoto is the 4th year student of Kinoshita laboratory in Tokyo University of Technology. She is researching for the personal authentication by tach operations on a smartphone. Ms. Miyamoto is majoring in cryptographic theory and biometric authentication.

Ms. Mayuko Hirose is the 2nd year student of Graduate school of Tokyo University of Technology. She is researching for computer and network system performance by queuing network or system simulation technique. Ms. Hirose is majoring in computer system performance evaluation.

Toshiyuki Kinoshita received his master of science from Tokyo University in 1977 and worked for Systems Development Laboratory, Hitachi Ltd. for 28 years. He received his Ph.D. of science from Tokyo Institute of Technology in 2000 and has become a professor in Tokyo University of Technology from 2005. Dr. Kinoshita is majoring in internet security, system programming and queuing theory.