An Analysis of Dynamic Relocation

Kimio Izawa*

Abstract

A probabilistic model for dynamic relocation is presented. And we analytically investigate some stationary characteristics of the model.

1. Introduction

In control of a multiprogramming/multiprocessing computer system, one of the problems is how to manage the allocation of memory space of the system.[1)] In this paper we consider a method of dynamic relocation, namely, compacting, present a probabilistic model for it, and investigate such stationary characteristics of the model as the probability distribution of the degree of multiprogramming, the expectation for it, the probability ditributions of the utilizations of the processors as well as the memory space, the expectations for them, and so on.

2. Model

The model consists of a first-in-first-served queue of which the length is kept fixed or infinite all the time and a service system which comprises some processors and a memory space of finite capacity, say N blocks (Figure 1).

The entities to be processed, called tasks, have the following characteristics:

(1) The time x required to process a task, in other words the length of stay of a task on the memory space, is exponentially distributed, if the number of tasks on the memory space, called the degree of multiprogramming, keeps constant. When the degree of multiprogramming is m at time t, the probability density function of x is $\mu(t)e^{-\mu(t)x}$, where $\mu(t)$ takes the value μ_m which is determined only in dependence on m. Usually we can expect that the following condition is satisfied:

$$\left.\begin{array}{l}\mu_1 \geq \mu_2 \geq \cdots \geq \mu_m \geq \cdots, \\ \mu_1 \leq 2\mu_2 \leq \cdots \leq m\mu_m \leq \cdots.\end{array}\right\} \qquad (1)$$

(2) The number j of blocks which a task requires to be allocated has a probability

---

This paper first appeared in Japanese in Joho－Shori (Journal of the Information Processing Society of Japan), Vol. 16, No. 5 (1975), pp. 410～418.

* Computation Center, Osaka University

distribution of general form $G(j)$. We denote g_j as the probability that a task requests j blocks, and so we have $G(j)=\Sigma_{i=1}^{j} g_i$.

(3) We assume that all the tasks are statistically identical and independent.

The control of the system is as follows:

(1) Tasks are transferred from the queue onto the memory space in the following manner. First, a task waiting at the top of the queue is compared with the free space. If the amount of space requested by the task is larger than the size of the free space, the move is deferred; otherwise, as many tasks as the free space can accomodate are moved one by one from the top of the queue within the limit of the maximum number M ($M \leq N$) of tasks on the memory space.

(2) On the memory space, tasks are placed in contact with previously placed one so that the free space is not split into fragments. Each task occupies a single continuous domain of as many blocks as it requested, and it does not change the size in mid course.

(3) When a task (say, task $\#n+h$ in Figure 1) which does not reside at the extreme right terminates, the memory space is repacked so that the gap does not remain between task $\#n+h-1$ (or the left end of the space in the case where $h=0$) and task $\#n+h+1$. After repacking (compacting), loading of new tasks begins again.

(4) The overhead, namely, the time required to control the system in accordance with (1), (2), and (3) above is neglected.

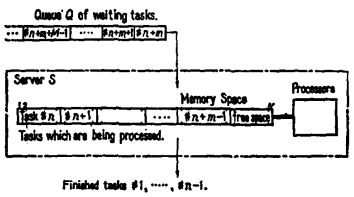

Fig. 1 Diagram for the Model I of dynamic relocation

3. Analysis

In this section we investigate some stationary properties of the model.

[Definition] Suppose that tasks $\#n$, $\#n+1,\ldots,$ $\#n+m-1$ ($1 \leq m \leq M$) reside on the memory space at time t, and sizes (numbers of blocks) of areas allocated to them are $i_1, i_2, \ldots, i_m$, respectively; at the same time, sizes of areas which top M tasks $\#n+m,\ldots,\#n+m+M-1$ in the queue will request are $j_1,\ldots,j_M$, respectively. We define a state $Q_{i_1 \cdots i_m 0 \cdots 0}^{i_1 \cdots i_m j_1 \cdots j_M}$ (or $Q_{i_1 \cdots i_m 0 \cdots 0}^{i_1 \cdots i_m j_1 \cdots j_M}(t)$) of the system as a $2M$-tuple $(i_1,\ldots,i_m,0,\ldots,0;j_1,\ldots,j_M)$.

First, we analyse the case where the maximum degree M of multiprogramming is 2. Subsequently, we expand the results to the general case where $2 \leq M \leq N$.

[In the case where $M=2$] We denote p_{ij}^{kl} (or $p_{ij}^{kl}(t)$) as the probability that the system is in state Q_{ij}^{kl} at time t. The state transitions of the system between time t and $t+\Delta t$ are as follows:

(1) Suppose that only a task of size i resides on the memory space, and the first two tasks on the top of the queue are of sizes k and l, respectively, at time $t+\Delta t$, in other words, the system is in state $Q_{i0}^{kl}(t+\Delta t)$ ($1\leq i\leq N$, $N-i+1\leq k\leq N$, $1\leq l\leq N$). The system can reach this state through one of the four ways of transition:

(a) At time t the system is already in state Q_{i0}^{kl}, and any transition does not occur between t and $t+\Delta t$. The probability for this case is $p_{i0}^{kl}(t)(1-\mu_1\Delta t)$, because the probability that a task terminates between t and $t+\Delta t$ is $\mu_m\Delta t$, if the degree of multiprogramming is m.

(b) At time t tasks of sizes i and h ($1\leq h\leq N-i$) are on the memory space, and the task of size h terminates between t and $t+\Delta t$. In spite of compacting, the task of size k at the top of the queue can not be moved onto the memory space because of a shortage of free space. In this case, as the order of tasks of sizes h and i is either hi or ih, the probability is $\Sigma_{h=1}^{N-i}p_{hi}^{kl}(t)\mu_2\Delta t(1-\mu_2\Delta t)+\Sigma_{h=1}^{N-i}p_{ih}^{kl}(t)\mu_2\Delta t(1-\mu_2\Delta t)$.

(c) At time t only a task of size h ($N-i+1\leq h\leq N$) resides on the memory space, and tasks of sizes i and k are on the top of the queue. Between t and $t+\Delta t$ the task of size h terminates, only the task of size i is moved onto the memory space, and the task of size k advances to the top so that a task of size l appears at the second place. The probability for this case is $\Sigma_{h=N-i+1}^{N}p_{h0}^{ik}(t)\mu_1\Delta t g_l$.

(d) Other cases rarely occur, so that the probability for them is $o(\Delta t)$.

As the above cases are mutually exclusive, we sum up the probabilities, and obtain the following difference equation:

$$\begin{aligned}p_{i0}^{kl}(t+\Delta t)&=p_{i0}^{kl}(t)(1-\mu_1\Delta t)\\&+\sum_{h=1}^{N-i}p_{hi}^{kl}(t)\mu_2\Delta t(1-\mu_2\Delta t)+\sum_{h=1}^{N-i}p_{ih}^{kl}(t)\mu_2\Delta t(1-\mu_2\Delta t)\\&+\sum_{h=N-i+1}^{N}p_{h0}^{ik}(t)\mu_1\Delta t g_l+o(\Delta t)\end{aligned}\qquad(2)$$

where $1\leq i\leq N$, $N-i+1\leq k\leq N$, $1\leq l\leq N$.

(2) Suppose that, at time $t+\Delta t$, tasks of sizes i and j reside on the memory space, and the first two tasks on the top of the queue are of sizes k and l, respectively, in other words, the system is in state $Q_{ij}^{kl}(t+\Delta t)$ ($1\leq i\leq N-1$, $1\leq j\leq N-i$, $1\leq k\leq N$, $1\leq l\leq N$). The system can reach this state through one of the four ways of transition:

(a) Any transition does not occur between t and $t+\Delta t$. The probability for this case is $p_{ij}^{kl}(t)(1-\mu_2\Delta t)^2$.

(b) At time t tasks of sizes i and h ($1\leq h\leq N-i$) are on the memory space, and tasks of sizes j and k are on the top of the queue. Between t and $t+\Delta t$ the task of size h terminates, and the memory is repacked. Then, the task of size j is moved onto the memory

space, the task of size k advances to the top so that a task of size l appears at the second place. The probability for this case is $\Sigma_{h=1}^{N-i} p_{hi}^{jk}(t)\mu_2\Delta t(1-\mu_2\Delta t)g_l + \Sigma_{h=1}^{N-i} p_{ih}^{jk}(t)\mu_2 \Delta t(1-\mu_2\Delta t)g_l$.

(c) At time t only a task of size h ($N-i+1 \leq h \leq N$) resides on the memory space, and tasks of sizes i and j are on the top of the queue. Between t and $t+\Delta t$ the task of size h terminates, tasks of sizes i and j are moved onto the memory space, and tasks of sizes k and l appear on the top of the queue. The probability for this case is $\Sigma_{h=N-i+1}^{N} p_{h0}^{ij}(t)\mu_2\Delta t g_k g_l$.

(d) Other cases rarely occur, so the probability is $o(\Delta t)$.

Also, as the above cases are mutually exclusive, we obtain the following difference equation:

$$p_{ij}^{kl}(t+\Delta t) = p_{ij}^{kl}(t)(1-\mu_2\Delta t)^2 + \sum_{h=1}^{N-i} p_{hi}^{jk}(t)\mu_2\Delta t(1-\mu_2\Delta t)q_l + \sum_{h=1}^{N-i} p_{ih}^{jk}(t)\mu_2\Delta t(1-\mu_2\Delta t)q_l + \sum_{h=N-i+1}^{N} p_{h0}^{ij}(t)\mu_1\Delta t q_k q_l + o(\Delta t) \tag{3}$$

where $1 \leq i \leq N-1$, $1 \leq j \leq N-i$, $1 \leq k \leq N$, $1 \leq l \leq N$.

Rewriting (2) and (3), and letting $\Delta t \to 0$, we have simultaneous differential equations. Then, assuming that the system reaches an equilibrium state as $t \to \infty$, and the limits: $\lim_{t\to\infty} p_{i0}^{kl}(t) = p_{i0}^{kl}$, $\lim_{t\to\infty} p_{ij}^{kl}(t) = p_{ij}^{kl}$ exist, we have the simultaneous linear equations:

$$\mu_1 p_{i0}^{kl} = \mu_2 \sum_{h=1}^{N-i}(p_{hi}^{kl} + p_{ih}^{kl}) + \mu_1 q_l \sum_{h=N-i+1}^{N} p_{h0}^{ik} \tag{4}$$

$$2\mu_2 p_{ij}^{kl} = \mu_2 q_l \sum_{h=1}^{N-i}(p_{hi}^{jk} + p_{ih}^{jk}) + \mu_1 q_k q_l \sum_{h=N-i+1}^{N} p_{h0}^{ij} \tag{5}$$

The solution relevant to the problem is as follows:

$$p_{i0}^{kl} = \frac{1}{\mu_1} q_l q_k q_l K_{2,N}, \tag{6}$$

($1 \leq i \leq N$, $N-i+1 \leq k \leq N$, $1 \leq l \leq N$)

$$p_{ij}^{kl} = \frac{1}{2\mu_2} q_l q_j q_k q_l K_{2,N}, \tag{7}$$

($1 \leq i \leq N-1$, $1 \leq j \leq N-i$, $1 \leq k \leq N$, $1 \leq l \leq N$)

where, because p_{i0}^{kl}, p_{ij}^{kl} are probabilities, $K_{2,N}$ must be as follows:

$$K_{2,N} = \frac{1}{\frac{1}{2\mu_2} G * G(N) + \frac{1}{\mu_1}(1 - G * G(N))} \tag{8}$$

(the symbol $*$ denotes convolution)

This result is extended to the case where the maximum degree of multiprogramming is M ($2 \leq M \leq N$) as follows:

$$\left.\begin{aligned} p_{i_1 0 \cdots 0}^{j_1 j_2 \cdots j_M} &= \frac{1}{\mu_1} q_{i_1} \left(\prod_{k=1}^{M} q_{j_k}\right) K_{M,N} \\ p_{i_1 i_2 0 \cdots 0}^{j_1 j_2 \cdots j_M} &= \frac{1}{2\mu_2} q_{i_1} q_{i_2} \left(\prod_{k=1}^{M} q_{j_k}\right) K_{M,N} \\ &\cdots\cdots \qquad\qquad \cdots\cdots \\ p_{i_1 i_2 \cdots i_M}^{j_1 j_2 \cdots j_M} &= \frac{1}{M\mu_M} \left(\prod_{k=1}^{M} q_{i_k}\right)\left(\prod_{k=1}^{M} q_{j_k}\right) K_{M,N} \end{aligned}\right\} \tag{9}$$

where $K_{M,N}=\dfrac{1}{\frac{1}{M\mu_M}G^{M*}(N)+\sum_{m=1}^{M-1}\frac{1}{m\mu_m}[G^{m*}(N)-G^{(m+1)*}(N)]}$ (10)

(the symbol $m*$ is defined as $G^{1*}(N)=G(N)$, $G^{m*}(N)=G*(G^{(m-1)*})(N)$.)

The stationary probability $p_{m,M}$ of the degree m of multiprogramming is given by

$$p_{m,N}=\sum_{i_1=1}^{N-(m-1)}\sum_{i_2=1}^{N-i_1-(m-2)}\cdots\sum_{i_m=1}^{N-(i_1+\cdots+i_{m-1})}\sum_{j_1=N-(i_1+\cdots+i_m)+1}^{N}\sum_{j_M=1}^{N}p_{i_1\cdots i_m 0\cdots 0}^{j_1\cdots j_m j_{m+1}\cdots j_M}$$
$$=\frac{1}{m\mu_m}\cdot K_{M,N}\{G^{m*}(N)-G^{(m+1)*}(N)\} \quad (11)$$

where $m=1,2,\ldots,M-1$, and

$$p_{M,N}=\sum_{i_1=1}^{N-(M-1)}\sum_{i_2=1}^{N-i_1-(M-2)}\cdots\sum_{i_M=1}^{N-(i_1+\cdots+i_{M-1})}$$
$$\sum_{j_1=1}^{N}\cdots\sum_{j_M=1}^{N}p_{i_1i_2\cdots i_M}^{j_1j_2\cdots j_M}=\frac{1}{M\mu_M}K_{M,N}G^{M*}(N) \quad (12)$$

The mean degree $\bar{M}_{M,N}$ of multiprogramming is, then,

$$\bar{M}_{M,N}=\sum_{m=1}^{M}m\,p_{m,N}=K_{M,N}\left[\frac{1}{\mu_M}G^{M*}(N)+\sum_{m=1}^{M-1}\frac{1}{\mu_m}\{G^{m*}(N)-G^{(m+1)*}(N)\}\right] \quad (13)$$

The stationary probability $p_{n/N}$ of the number n of blocks in use is given by

$$p_{n/N}=\sum_{k=1}^{n}\sum_{i_1+\cdots+i_k=n}\sum_{j_1=N-n+1}^{N}\sum_{j_2=1}^{N}\cdots\sum_{j_M=1}^{N}p_{i_1i_2\cdots i_k0\cdots0}^{j_1j_2\cdots\cdots j_M}=\sum_{k=1}^{n}\frac{1}{k\mu_k}K_{M,N}g^{k*}(n)\{1-G(N-n)\} \quad (14)$$

in the case where $n=1,2,\ldots,M-1$; and

$$p_{n/N}=\sum_{k=1}^{M-1}\sum_{i_1+\cdots+i_k=n}\sum_{j_1=N-n+1}^{N}\sum_{j_2=1}^{N}\cdots\sum_{j_M=1}^{N}p_{i_1i_2\cdots i_k0\cdots0}^{j_1j_2\cdots\cdots j_M}+\sum_{i_1+\cdots+i_M=n}\sum_{j_1=1}^{N}\cdots\sum_{j_M=1}^{N}p_{i_1i_2\cdots i_M}^{j_1j_2\cdots j_M}$$
$$=\sum_{k=1}^{M-1}\frac{1}{k\mu_k}K_{M,N}g^{k*}(n)\{1-G(N-n)\}+\frac{1}{M\mu_M}K_{M,N}g^{M*}(n) \quad (15)$$

in the case where $n=M, M+1, \ldots, N$.

The mean $\bar{U}_{M,N}$ (the utilization of the memory space) is given by

$$\bar{U}_{M,N}=\sum_{n=1}^{N}n\,p_{n/N} \quad (16)$$

The output rate $\bar{R}_{M,N}$ of the system (i.e. the number of tasks which terminate in a unit of time) is given by

$$\bar{R}_{M,N}=\sum_{m=1}^{M}m\mu_m p_{m,N}=K_{M,N} \quad (17)$$

In particular, if $M=1$ (uniprogramming), we have $\bar{R}_{1,N}=\mu_1$.

4. Conclusion

If we consider some particular distributions for $G(j)$, we can obtain numerical results which are greatly useful. For example, in the case of the uniform distribution $G(j)=j/N$ $(j=1,2,\ldots,N)$: if $\mu_1=2\mu_2=3\mu_3=\cdots$, then we have $\bar{U}_{\infty,\infty}=e-2=0.71828\ldots$; if $\mu_1=\mu_2=\mu_3=\cdots$, then we have $\bar{U}_{\infty,\infty}=1/2(e-2)=0.69610\ldots$; and others.

1) T. Betteridge: "An Analytic Storage Allocation Model", Acta-Informatica, 3, 2, p. 101 (1974)