On Optimal Multiprogramming for Computation Centers Having Many Users

Kimio Izawa*

Abstract

We consider a method based on the theory of dynamic programming[1] to obtain an optimal policy of partitioning the main memory of a computer system installed at a computation center which has many users.

1. Introduction

We consider an ordinary computer system which enables us to perform jobs under either multiprogramming or uniprogramming environment according to the policy of the computation center. We confine ourselves to discuss the case where the main memory space of the computer is partitioned into domains so that each of them has a fixed size. Suppose that we know the statistical data on the sum of memory occupation times of jobs which arrive in a certain definite period in the form of a function of the size of memory allocated. We assume that, under a multiprogramming environment of degree N, we partition the main memory space of capacity C into N domains such that $N=\Sigma_{i=1}^{n} p_i$, $C=\Sigma_{i=1}^{n} p_i u_i$, and $\alpha(=u_0)\leq u_1 \leq u_2 \leq \cdots \leq u_n$, so that, on domains of capacity u_i, only jobs requiring a memory space of capacity v such that $u_{i-1} \leq v \leq u_i$ are performed; while under a uniprogramming environment jobs such that $u_n \leq v \leq C$ are performed.

The problem is, subject to the above mentioned constraints, to minimize the time needed to perform all the jobs. In this paper, we consider some fundamental aspects of the problem, and show an analytical method to obtain the optimal policy to partition the main memory space into domains.

2. Formalization and analysis

Let $T_N(s)$ be the sum of memory occupation times of jobs requiring memory space of size s under the multiprogramming environment of degree N. Usually, we expect that the following conditions hold:

$$T_1(s) \leq T_2(s) \leq \cdots \leq T_N(s) \leq \cdots \qquad (1)$$

This paper first appeared in Japanese in Joho-Shori (Journal of the Information Processing Society of Japan), Vol. 15, No. 8 (1974), pp. 581~588.

* Computation Center, Osaka University

$$T_1(s) \geq T_2(s)/2 \geq \cdots \geq T_N(s)/N \geq \cdots \tag{2}$$

For simplicity we regard the ratios

$$k_i = T_i(s)/T_1(s) \quad (i=1,2,\ldots) \tag{3}$$

as constants determined in dependence upon the system.

Let P be an infinite sequence $\langle p_1, p_2, \cdots \rangle$ of positive integers, and P_n be $\langle p_1, p_2, \cdots, p_n \rangle$.

For convenience we handle the size of memory space as a real number. We denote C as the capacity of memory space.

When the computer system is operated under the multiprogramming environment of degree N, we partition the main memory space into N domains $D_{ij}^{(P_n)}$ (where $N=p_1+\cdots+p_n$, $\langle p_1, p_2, \cdots, p_n \rangle = P_n$, $j=1,2,\cdots,p_i$, $i=1,2,\cdots,n$) such that, letting the size of $D_{ij}^{(P_n)}$ be $u_{ij}^{(P_n)}$,

$$\begin{aligned} &\alpha \leq u_{11}^{(P_n)} = u_{12}^{(P_n)} = \cdots = u_{1p_1}^{(P_n)} (= u_1^{(P_n)}) \leq u_{21}^{(P_n)} = u_{22}^{(P_n)} = \cdots = u_{2p_2}^{(P_n)} (= u_2^{(P_n)}) \leq \cdots \\ &\leq u_{n1}^{(P_n)} = u_{n2}^{(P_n)} = \cdots = u_{np_n}^{(P_n)} (= u_n^{(P_n)}), \end{aligned} \tag{4}$$

$$p_1 u_1^{(P_n)} + p_2 u_2^{(P_n)} + \cdots + p_n u_n^{(P_n)} = C. \tag{5}$$

Note that a lower bound α ($\alpha \geq 0$) of $u_{ij}^{(P_n)}$ makes sure to satisfy such necessity as each domain should be able to accomodate some system programs, say FORTRAN compiler, in addition to user programs.

On the domains $D_{ij}^{(P_n)}$ ($j=1,2,\ldots,p_i$) only jobs requiring a memory space of size v such that $u_{i-1}^{(P_n)} \leq v \leq u_i^{(P_n)}$ ($u_0^{(P_n)}=\alpha$) are performed, and the elapsed times needed to perform jobs there are equal to each other. On the one hand jobs requiring a memory space of size v such that $u_n^{(P_n)} \leq v \leq C$ are performed under the uniprogramming environment. We call such a form of operation of the computer system a P_n-typed operation, and the multiprogramming in a P_n-typed operation a P_n-typed multiprogramming.

The sum T_x, obtained periodically, of memory occupation times of jobs requiring a memory space of size v such that $\alpha \leq v \leq x$ ($x \leq C$) is given as follows: under the multiprogramming environment of degree N

$$T_x = \int_\alpha^x f_N(v)\,dv \tag{6}$$

The integrand f_N is assumed to have the following properties:

(F1) The function $f_N(x)$ ($N=1,2,\ldots$) is bounded on the interval $[\alpha, C]$, and has there finite discontinuity points at most. So that it is integrable, and further the integral of $f_N(x)$ from a to b ($\alpha \leq a \leq b \leq C$) becomes positive, as $f_N(x)$ is positive at any continuity point $x \in [\alpha, C]$.

(F2) There exist constants k_N ($N=1,2,\ldots$) such that

$$f_N(x)=k_N f_1(x),\ x\in[\alpha,C] \tag{7}$$

$$k_1(=1)\leq k_2\leq k_3\leq\cdots \tag{8}$$

$$k_1\geq k_2/2\geq k_3/3\geq\cdots \tag{9}$$

Note that T_x is a continuous function of x in the interval $[\alpha,C]$.

We define $J_i^{(P_n)}(C)$ (or $J_i^{(P_n)}$) as

$$J_i^{(P_n)}=\frac{1}{p_i}\int_{u_{i-1}}^{u_i} f_N(x)dx \tag{10}$$

Here, $J_i^{(P_n)}$ represents the time needed to perform all the jobs of a definite period which are assigned to the domain $D_{ij}^{(P_n)}$ $(1\leq j\leq p_i)$ under the P_n-typed multiprogramming environment. The length of time when jobs are multiprocessed is $\max_{i=1,2,\ldots,n}[J_i^{(P_n)}]$, while the length of time when jobs are uniprocessed, $R^{(P_n)}(C)$ (or $R^{(P_n)}$) defined as

$$R^{(P_n)}=\int_{u_0}^{C} f_1(x)dx. \tag{11}$$

First, we consider the problem of optimal partitioning of memory space for P_n-typed multiprogramming. In other words this problem is to minimize the time of multiprogrammed operation.

[Problem I] Find a solution $[u_i, J_i^{(P_n)}\ (i=1,2,\ldots,n)]$ which minimize the target function $\max_{i=1,2,\ldots,n}[J_i^{(P_n)}]$ subject to the condition:

$$(R)\qquad \begin{cases} p_1u_1+p_2u_2+\cdots+p_nu_n=C, \\ \alpha\leq u_1\leq u_2\leq\cdots\leq u_n \end{cases}$$

Next, we discuss some properties of Problem I.

[Proposition 1] $[u_i, J_i^{(P_n)}\ (i=1,2,\ldots,n)]$ which satisfies the condition (R) is a solution for Problem I, if and only if it satisfies the condition:

$$(E)\qquad J_1^{(P_n)}=J_2^{(P_n)}=\cdots=J_n^{(P_n)}$$

[Proposition 2] There exists a solution for Problem I, if and only if

$$C\geq N\alpha \tag{12}$$

And when a solution exists, it is unique.

The next proposition gives us a method to solve the problem.

[Proposition 3] Let $[u_i^{(P_n)}, J_i^{(P_n)}\ (i=1,2,\ldots,n)]$ be a solution for Problem I of P_n-typed multiprogramming. It follows that $[u_i^{(P_n)}, (k_{(N-p_n)}/k_N)J_i^{(P_n)}\ (i=1,\ldots,n-1)]$ is a solution for Problem I of P_{n-1}-typed multiprogramming where the capacity of memory is $C-p_nu_n^{(P_n)}$. In other words, in case where $n\geq 2$, denoting the solution for the latter as $[u_i^{(P_{n-1})}(C-p_nu_n^{(P_n)}(C)),\ J_i^{(P_{n-1})}(C-p_nu_n^{(P_n)}(C))\ (i=1,2,\ldots,n-1)]$, we have

$$u_i^{(P_{n-1})}(C-p_nu_n^{(P_n)}(C))=u_i^{(P_n)}(C), \tag{13}$$

$$J_i^{(P_{n-1})}(C-p_nu_n^{(P_n)}(C))=(k_{(N-p_n)}/k_N)J_i^{(P_n)}(C)$$
$$=(k_{(N-p_n)}/k_N)J_n^{(P_n)}(C) \tag{14}$$

Proposition 3 shows that the problem can be solved as a multistage decision process.

The recurrence equation for Problem I of P_n-typed multiprogramming is as follows:

(1) In case where n=1.

$$\left.\begin{aligned} J_1^{(P_1)}(C)&=\frac{1}{p_1}\int_{\alpha}^{u_1^{(P_1)}(C)} f_{P_1}(x)dx,\\ u_1^{(P_1)}(C)&=\frac{C}{p_1},\\ R^{(P_1)}(C)&=\int_{C/p_1}^{C} f_1(x)dx. \end{aligned}\right\} \tag{15}$$

(2) In case where $n\geq 2$.

$$\left.\begin{aligned} J_n^{(P_n)}(C)&=\frac{k_N}{k(N-p_n)}J_{n-1}^{(P_{n-1})}(C-p_nu_n^{(P_n)}(C)),\\ u_{n-1}^{(P_n)}(C)&=u_{n-1}^{(P_{n-1})}(C-p_nu_n^{(P_n)}(C)). \end{aligned}\right\} \tag{16}$$

A solution for the recurrence equation (15) and (16), $[u_n=u_n^{(P_n)}(C), u_{n-1}=u_{n-1}^{(P_{n-1})}(C-p_nu_n), u_{n-2}=u_{n-2}^{(P_{n-2})}(C-p_nu_n-p_{n-1}u_{n-1}),\ldots, u_1=u_1^{(P_1)}(C-p_nu_n-p_{n-1}u_{n-1}-\cdots-p_2u_2)$, and $J_n^{(P_n)}(C)]$ is the required solution for Problem I.

The time spent to perform all the jobs which arrive in a definite period is $J^{(P_n)}=J_n^{(P_n)}+R^{(P_n)}$ in case of the P_n-typed operation.

3. Example

An example will show how to solve the problem analytically.

[Example 1] Decide the optimal type of operation for a computation center where $f_1(x)=K$(constant)>0 $(x\varepsilon[\alpha,C])$.

(Solution)

(1.1) For n=1, the solution is obtained from (15) as follows:

$$\left.\begin{aligned} u_1^{(P_1)}(C)&=\frac{C}{p_1},\\ J_1^{(P_1)}(C)&=\frac{1}{p_1}\int_{\alpha}^{C/p_1} k_{p_1}Kdx=\frac{k_{p_1}}{p_1^2}K(C-p_1\alpha), \end{aligned}\right\} \tag{17}$$

$$R^{(P_1)}(C)=\int_{C/p_1}^{C} Kdx=K\frac{(p_1^2-p_1)C}{p_1^2}. \tag{18}$$

(1.2) For n=2, letting u_1 and u_2 denote $u_1^{(P_2)}(C)$ and $u_2^{(P_2)}(C)$, respectively, from (16) we have

$$u_1=u_1^{(P_1)}(C-p_2u_2)=(C-p_2u_2)/p_1. \tag{19}$$

From (10) we have

$$\begin{aligned} J_2^{(P_2)}(C)&=\frac{1}{p_2}\int_{u_1}^{u_2} f_{(p_1+p_2)}(x)dx\\ &=\frac{k(p_1+p_2)}{p_1p_2}K\{(p_1+p_2)u_2-C\}. \end{aligned} \tag{20}$$

From (16) we have

$$\begin{aligned} J_2^{(P_2)}(C)&=\frac{k(p_1+p_2)}{k_{p_1}}J_1^{(P_1)}(C-p_2u_2)\\ &=\frac{k(p_1+p_2)}{p_1^2}K(C-p_2u_1-p_1\alpha). \end{aligned} \tag{21}$$

From (20) and (21) we have

$$u_2=\frac{(p_1+p_2)C-p_1p_2\alpha}{p_1^2+p_1p_2+p_2^2}. \tag{22}$$

Substituting (22) into (19), (20), and (11), we obtain

$$u_1=\frac{p_1C+p_2^2\alpha}{p_1^2+p_1p_2+p_2^2}, \tag{23}$$

$$J_2^{(P_2)}(C)=\frac{k(p_1+p_2)}{p_1^2+p_1p_2+p_2^2}K\{C-(p_1+p_2)\alpha\}, \tag{24}$$

$$R^{(P_2)}(C)=K\frac{(p_1^2+p_1p_2+p_2^2-p_1-p_2)C+p_1p_2\alpha}{p_1^2+p_1p_2+p_2^2}. \tag{25}$$

(1.3) The general solution is as follows:

$$u_n=\frac{\{\sum_{i=1,2,\cdots,n}p_i\}C-\{\sum_{i,j=1,2,\cdots n;\ i<j}p_ip_j\}\alpha}{\sum_{i,j=1,2,\cdots,n,\ i\leq j}p_ip_j}, \tag{26}$$

$$u_{n-1}=\frac{\{\sum_{i=1,2,\cdots,n-1}p_i\}C-\{\sum_{i,j=1,2,\cdots,n;\ i<j}p_ip_j-p_n(\sum_{i=1,2,\cdots,n}p_i)\}\alpha}{\sum_{i,j=1,2,\cdots,n:\ i\leq j}p_ip_j}, \tag{27}$$

$$u_{n-2}=\frac{\{\sum_{i=1,2,\cdots,n-2}p_i\}C-\{\sum_{i,j=1,2,\cdots,n;\ i<j}p_ip_j-(p_n+p_{n-1})(\sum_{i=1,2,\cdots,n}p_i)\}\alpha}{\sum_{i,j=1,2,\cdots,n;\ i\leq j}p_ip_j}, \tag{28}$$

$$\vdots$$

$$u_1=\frac{p_1C-\{\sum_{i,j=1,2,\cdots,n;\ i<j}p_ip_j-(p_n+p_{n-1}+\cdots+p_2)(\sum_{i=1,2,\cdots,n}p_i)\}\alpha}{\sum_{i,j=1,2,\cdots,n;\ i\leq j}p_ip_j}, \tag{29}$$

$$J_n^{(P_n)}(C)=\frac{k(p_1+p_2+\cdots+p_n)}{\sum_{i,j=1,2,\cdots,n;\ i\leq j}p_ip_j}K\{C-(\sum_{i=1,2,\cdots,n}p_i)\alpha\}, \tag{30}$$

$$R^{(P_n)}(C)=K\frac{\{\sum_{i,j=1,2,\cdots,n;\ i\leq j}p_ip_j-\sum_{i=1,2,\cdots,n}p_i\}C+\{\sum_{i,j=1,2,\cdots,n:\ i<j}p_ip_j\}\alpha}{\sum_{i,j=1,2,\cdots,n;\ i\leq j}p_ip_j}. \tag{31}$$

4. Conclusion

Some numerical values calculated from the above result show us that, if we adopt multiprogramming in the proper form, we can considerably save computer time required to process jobs.

1) R. Bellman: "Dynamic Programming", Princeton Univ. Press, Princeton, New Jersey, (1957)