Journal of the Operations Research
Society of Japan
Vol. 26, No. 1, March 1983

AN APPLICATION OF THE LUMPING METHOD TO A LOSS SYSTEM WITH TWO TYPES OF CUSTOMERS

Tsukasa Sato
Nissan Diesel Co. Ltd.

Masao Mori
Tokyo Institute of Technology

(Received May 27, 1982; Revised November 29, 1982)

Abstract This paper applies the lumping method, proposed by Takahashi[4], to get stationary probabilities numerically for a loss system with two types of input streams. By using the method we can evaluate numerically the effect of introducing several servers who are capable to serve both types of customers.

1. Introduction

In a telephone service system for ticket reservation in an airline company, some reception clerks serve only for domestic line passangers and others serve only for international line passangers. In this manner of service there often happens such an unbalanced situation that all clerks for domestic lines are hard at work, while some servers for international lines are idle, or vice-versa. So it is expected that we could improve service availability by introducing several well-trained servers who are capable to deal with any service requirements for both types of customers. In this paper we would examine how these commonly usable servers could decrease the total loss probability.

In the section 2, we will introduce a model to analyze above phenomena, which is described as a Markov process with a large state space. Such the largeness of state space makes it almost impossible even for a powerful computer to get stationary probabilities for this model by using ordinary Gauss-Seidel method or Gauss elimination method. So we provide an abbreviated algorithm of lumping method, originally proposed in Takahashi[4] and Takahashi & Takami[5], to get stationary probabilities in the section 3. In the section 4, we will examine some numerical examples.

2. Model

We consider a loss system with S servers, into which two types of customers are arriving. Each input stream is a Poisson process with rate λ_k $(k=1, 2)$. S_k servers can only deal with the type k customers $(k=1,2)$, and the rests $S_3=S-S_1-S_2$, called commonly usable servers, are highly educated to be able to serve both types of customers. And we assume that commonly usable servers deal with customers overflowed from their own assignmented S_1 or S_2 servers. A type k customer who finds each S_k and S_3 servers are all busy should be lost.

Here each service time is assumed to be independently and exponentially distributed. S_k servers serve with rate μ_k $(k=1,2)$. Commonly usable S_3 servers may be somewhat less efficient on account of complex job, so they are assumed to server with rate μ_3 for type 1 coustomers and with rate μ_4 for type 2 customers.

By setting above, we can easily formulate the model as a Markov process having an appropriate infinitesimal generator Q on the state space $\mathcal{S}=\{(i_1,i_2, i_3,i_4)\ ;\ 0\leqq i_1\leqq S_1,\ 0\leqq i_2\leqq S_2,\ 0\leqq i_3+i_4\leqq S_3,\ i_3,i_4\geqq 0\}$, where i_k denotes the number of servers serving among S_k servers $(k=1,2)$ and i_3 and i_4 denote the number of servers among commonly usable S_3 servers serving type 1 and type 2 customers respectively. Note that the total number of states is $N=(S_1+1)(S_2+1)(S_3+1)(S_3+2)/2$, which is fairly large. For example, $N=29,106$ in the case of $S_1=S_2=20$ and $S_3=10$.

Clearly the process is regular, that is, irreducible and aperiodic. Thus the stationary distribution exists and is given by the unique stochastic vector $\underline{X}$ satisfying $\underline{X}Q=\underline{0}$, where $\underline{0}$ is a N-dimensional zero vector, However the largeness of state apace makes it difficult to get $\underline{X}$ numerically in an ordinaly manner. In the next section we will contrive to calculate $\underline{X}$ by using lumping method.

3. Application of the Lumping Method

Let us assign a number to each state of the Markov process in the lexicographic order, i.e. $1=(0,0,0,0)$, $2=(0,0,0,1),\cdots$, $S_3+1=(0,0,0,S_3)$, $S_3+2=(0,0,1,0),\cdots$, $N-1=(S_1,S_2,S_3-1,1)$ and $N=(S_1,S_2,S_3,0)$. And let us partition the state space $\mathcal{S}$ into groups (or blocks) indexed by (i_1,i_2) as follows:

$$A_1=\{(0,0,i_3,i_4)\ :\ 0\leqq i_3+i_4\leqq S_3\ ,\ \ i_3,i_4\geqq 0\}\equiv\{1,\ 2,\ \cdots,\ m\}$$

$$A_2=\{(0,1,i_3,i_4)\ :\ 0\leqq i_3+i_4\leqq S_3\ ,\ \ i_3,i_4\geqq 0\}\equiv\{m+1,m+2,\ \cdots,\ 2m\}$$

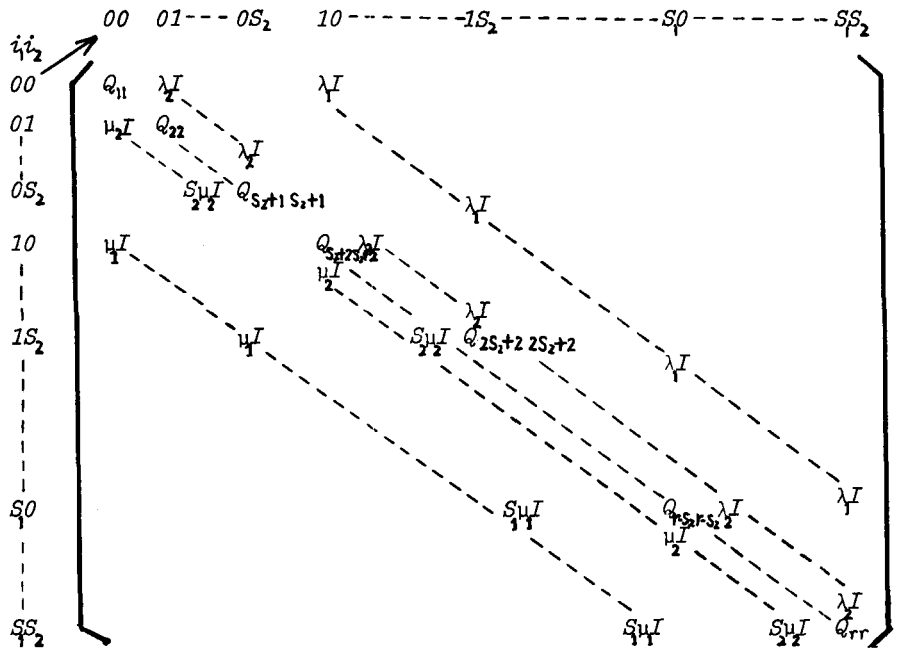
$$\vdots$$

$$A_r = \{(S_1, S_2, i_3, i_4) : 0 \leq i_3 + i_4 \leq S_3, \quad i_3, i_4 \geq 0\} \equiv \{N-m+1, N-m+2, \cdots, N\}$$

where $r=(S_1+1)(S_2+1)$ is the number of groups (or blocks). We call each A_i a lumped state. And group size (or block size) of each lumped state is $m=(S_3+1)(S_3+2)/2$.

In the manner of above numbering of the state space, the infinitesimal generator Q is written as

(3.1) $Q =$



where each entry designated by blank is zero and Q_{ii} for $i=(i_1, i_2)$ is represented by

$$Q_{ii} = \begin{cases} T-(i_1\mu_1 + i_2\mu_2)I & \text{for } 0 \leq i_1 \leq S_1-1 \text{ and } 0 \leq i_2 \leq S_2-1 \\ T + \Lambda_1-(S_1\mu_1 + i_2\mu_2)I & \text{for } 0 \leq i_2 \leq S_2-1 \text{ and } i_1=S_1 \\ T + \Lambda_2-(i_1\mu_1 + S_2\mu_2)I & \text{for } 0 \leq i_1 \leq S_1-1 \text{ and } i_2=S_2 \\ T + \Lambda_1 + \Lambda_2-(S_1\mu_1 + S_2\mu_2)I & \text{for } i_1=S_1 \text{ and } i_2=S_2 \end{cases}$$

Here Q_{ii}, T, Λ_1 and Λ_2 are $m \times m$ submatrices representing transition rate within each group, I is the $m \times m$ identical matrix. T, Λ_1 and Λ_2 are just defined for computational convenience. For instance, in the case of $S_3=3$, T, Λ_1 and Λ_2 are given aby the following matrices:

$T =$

$(i\ i)$	$(0,0)$	$(0,1)$	$(0,2)$	$(0,3)$	$(1,0)$	$(1,1)$	$(1,2)$	$(2,0)$	$(2,1)$	$(3,0)$
$(0,0)$	$-(\lambda_1+\lambda_2)$									
$(0,1)$	μ_4	$-(\lambda_1+\lambda_2+\mu_4)$								
$(0,2)$		$2\mu_4$	$-(\lambda_1+\lambda_2+2\mu_4)$							
$(0,3)$			$3\mu_4$	$-(\lambda_1+\lambda_2+3\mu_4)$						
$(1,0)$	μ_3				$-(\lambda_1+\lambda_2+\mu_3)$					
$(1,1)$		μ_3			μ_4	$-(\lambda_1+\lambda_2+\mu_3+\mu_4)$				
$(1,2)$			μ_3			$2\mu_4$	$-(\lambda_1+\lambda_2+\mu_3+2\mu_4)$			
$(2,0)$					$2\mu_3$			$-(\lambda_1+\lambda_2+2\mu_3)$		
$(2,1)$						$2\mu_3$		μ_4	$-(\lambda_1+\lambda_2+2\mu_3+\mu_4)$	
$(3,0)$								$3\mu_3$		$-(\lambda_1+\lambda_2+3\mu_3)$

$\Lambda_1 =$

(i,i)	$(0,0)$	$(0,1)$	$(0,2)$	$(0,3)$	$(1,0)$	$(1,1)$	$(1,2)$	$(2,0)$	$(2,1)$	$(3,0)$
$(0,0)$	0	0	0	0	λ_1	0	0	0	0	0
$(0,1)$		0	0	0	0	λ_1	0	0	0	0
$(0,2)$			0	0	0	0	λ_1	0	0	0
$(0,3)$				λ_1	0	0	0	0	0	0
$(1,0)$					0	0	0	λ_1	0	0
$(1,1)$						0	0	0	λ_1	0
(1.2)							λ_1	0	0	0
$(2,0)$								0	0	λ_1
$(2,1)$									λ_1	0
$(3,0)$										λ_1

and

$\Lambda_2 =$

(i,i)	$(0,0)$	$(0,1)$	$(0,2)$	$(0,3)$	$(1,0)$	$(1,1)$	$(1,2)$	$(2,0)$	$(2,1)$	$(3,0)$
$(0,0)$	0	λ_2	0	0	0	0	0	0	0	0
$(0,1)$		0	λ_2	0	0	0	0	0	0	0
$(0,2)$			0	λ_2	0	0	0	0	0	0
$(0,3)$				λ_2	0	0	0	0	0	0
$(1,0)$					0	λ_2	0	0	0	0
$(1,1)$						0	λ_2	0	0	0
$(1,2)$							λ_2	0	0	0
$(2,0)$								0	λ_2	0
$(2,1)$									λ_2	0
$(3,0)$										λ_2

Here we write Q concisely as

(3.2) $Q = (Q_{ij})$,

where Q_{ij} is a $m \times m$ submatrix representing transition rate from the group A_i to the group A_j. And also we split the stationary distribution vector $\underline{X}$ into

(3.3) $\underline{X} = (\underline{X}_1, \underline{X}_2, \cdots, \underline{X}_r)$

where each $\underline{X}_i$ is a m-dimensional row vector corresponding to A_i.

Let $\underline{e}$ be the m-dimensional column vector with all entries equal to 1, and $\underline{\pi}=(\omega_i)$ be the r-dimensional row vector whose i-th entry is $\omega_i=\underline{X}_i \cdot \underline{e}$. The entry ω_i denotes the probability that the process is in the group A_i in the steady state. Let $\underline{\beta}_i = \frac{1}{\omega_i}\underline{X}_i \equiv (\beta_{ik}; k \in A_i)$. Then $\underline{\beta}_i \cdot \underline{e}=1$ and the entry β_{ik} denotes the conditional probability that the process is in the state k in the steady state given that the process is in the group A_i.

In the general lumping method both the vector $\underline{\pi}$ and $\underline{\beta}_i$ should be calucu-lated iteratively. And the total number of iterations increases rapidly as the number of groups r and the group size m increase.

However in our model the marginal distribution $\underline{\pi}=(\omega_i)$ is given by

$$(3.4)\quad \omega_i = \left[\frac{1}{i_1!}\left(\frac{\lambda_1}{\mu_1}\right)^{i_1} \Big/ \sum_{j=0}^{S_1}\frac{1}{j!}\left(\frac{\lambda_1}{\mu_1}\right)^{j}\right] \cdot \left[\frac{1}{i_2!}\left(\frac{\lambda_2}{\mu_2}\right)^{i_2} \Big/ \sum_{j=0}^{S_2}\frac{1}{j!}\left(\frac{\lambda_2}{\mu_2}\right)^{j}\right]$$

for the lumped state $A_i = \{(i_1, i_2, i_3, i_4),\ 0 \le i_3+i_4 \le S_3,\ i_3, i_4 \ge 0\}$, which is easily obtained by considering the independent behaviour of the solely usable S_1 and S_2 servers for type 1 and type 2 customers respectively. So we can omitt the procedure to calculate the vector $\underline{\pi}$. Thus the abbreviated algorithm is given in the following. Here we will designate the conditional probability vectors at the nth iteration as $\underline{\beta}_i(n)$.

Algorithm

〈Step 1〉 Start with arbitrary initial stochastic vectors

$\underline{\beta}_i(0)$, $i=1, 2, \cdots, r$.

↓

〈Step 2〉 At the nth iteration, solve the equation

$$(3.5)\quad \overline{\beta}_i(n)Q_{ii} + \sum_{j=1}^{i-1}\omega_j\underline{\beta}_j(n)Q_{ji} + \sum_{j=i+1}^{r}\omega_j\underline{\beta}_j(n-1)Q_{ji} = \underline{0}_m$$

for $\overline{\beta}_i(n)$, where $\underline{0}_m$ is the m-dimentional zero vector.

NO

↓

〈Step 3〉 Normalize $\overline{\beta}_i(n)$ as

$$(3.6)\quad \underline{\beta}_i(n) = \overline{\beta}_i(n) \,/\, (\overline{\beta}_i(n)\cdot\underline{e})$$

↓

〈Step 4〉 For a given sufficiently small $\varepsilon>0$, if

YES

$$(3.7)\quad |\beta_{ik}(n-1)-\beta_{ik}(n)| < \varepsilon \quad \textit{for all } i, k$$

↓

〈Step 5〉 Set the stationary distribution as

$\underline{X} = (\omega_i\underline{\beta}_i(n))$.

In this algorithm the total computational time depends on how to solve the equation (3.5) effectively. Ordinarilly we solve it by Gauss-Seidel iteration method. Fortunately, however, many of diagonal matrices Q_{ii} are triangular, so in such a case we can directly solve the equation by Gauss elimination method. By this little idea we can reduce computational time much.

4. Numerical Examples

In our problem, we are interested mainly in the values of the following loss probabilities:

P_1 : probability a type *1* customer is lost,

P_2 : probability a type *2* customer is lost,

$P_t = \dfrac{\lambda_1 P_1 + \lambda_2 P_2}{\lambda_1 + \lambda_2}$: probability an arbitrary customer is lost.

We call P_t the total loss rate of the system.

Table 1 shows some values of loss probabilities for the case of $S=10$. For a given set of parameters λ_i and μ_i, the optimal assignment of servers to minimize the total loss rate P_t without introducing commonly usable servers is $(S_1=6,\ S_2=4)$. Each loss probabilities in this case are $(P_1=0.05216,\ P_2=0.09524,\ P_t=0.06939)$. Then we assume *3* servers to be trained as commonly usables servers, but their service speed is assumed to be not so high as that of other servers dealing with only one type of jobs: here we set μ_3 and μ_4 as *90%* of μ_1 and μ_2 respectively. The optimal assignment for the case turns to be $(S_1=4,\ S_2=3,\ S_3=3)$, where loss probabilities are $(P_1=0.03898,\ P_2=0.03432,\ P_t=0.03712)$. It is noteworthy that these values are all smaller than corresponding probabilities for the case of $(S_1=6,\ S_2=4,\ S_3=0)$. That is to say, we can improve system availabity/performance both for customers and the manager of the system by training a few as commonly usable servers.

Here we mention computational times. We used HITAC 8350, which was installed about ten years ago. In getting probabilities we set $\varepsilon=10^{-6}$. For the case of $(S=10;\ S_1=5,\ S_2=2,\ S_3=3;$ no. of blocks$=(S_1+1)(S_2+1)=18,$ block size$=(S_3+1)(S_3+2)/2=10)$, CPU time is *72* seconds. For the case of $(S=30;\ S_1=17,\ S_2=10,\ S_3=3;$ no. of blocks$=(17+1)(10+1)=198,$ block size$=(3+1)(3+2)/2=10)$, CPU time swells up to *59* minutes on account of increasing number of iterations in our algorithm. However, when our algorithm was checked by using ACOS 1000 at Tohoku University, it was reported that CPU times for $S=10$ are negligible small and the latter case takes *80* seconds. In practice we could set ε as $10^{-3}\sim10^{-5}$according to the order of numerical value of probabilities actually

Table 1. Loss probabilities

$S_1+S_2+S_3=10$,
$\lambda_1=3.0$, $\lambda_2=2.0$,
$\mu_1=1.0$, $\mu_2=1.0$, $\mu_3=0.9$, $\mu_4=0.9$

S_1	S_2	S_3	P_1	P_2	P_t
9	1	0	.00270	.16667	.26829
8	1	1	.00557	.41740	.17030
7	1	2	.00976	.23754	.10087
6	1	3	.01453	.12722	.05961
5	1	4	.01887	.07094	.03970
4	1	5	.02238	.04615	.03188
3	1	6	.02532	.03560	.02943
2	1	7	.02809	.03068	.02913
1	1	8	.03089	.02811	.02977
0	1	9	.03382	.02667	.03069
8	2	0	.00813	.40000	.16488
7	2	1	.01324	.22383	.09748
6	2	2	.01878	.11724	.05816
5	2	3	.02364	.06372	.03968
4	2	4	.02750	.04055	.03272
3	2	5	.03076	.03078	.03077
2	2	6	.03385	.02625	.03081
1	2	7	.03701	.02387	.03175
0	2	8	.04033	.02252	.03321
7	3	0	.02186	.21053	.09323
6	3	1	.02870	.10695	.06000
5	3	2	.03446	.05596	.04306
4	3	3	.03898	.03432	.03712
3	3	4	.04285	.02539	.03587
2	3	5	.04662	.02129	.03649
1	3	6	.05054	.01913	.03798
0	3	7	.05470	.01789	.03998
6	4	0	.05216	.09524	.06939
5	4	1	.05925	.04688	.05430
4	4	2	.06484	.03703	.04972
3	4	3	.06982	.01913	.04954
2	4	4	.07485	.01562	.05116
1	4	5	.08022	.01379	.05365
0	4	6	.08598	.01273	.05668
5	5	0	.11005	.03670	.08071
4	5	1	.11697	.01914	.07784
3	5	2	.12352	.01260	.07915
2	5	3	.13048	.00986	.08223
1	5	4	.13808	.00849	.08624
0	5	5	.14635	.00770	.09089

required. Then the number of iterations calculating β_i (Step 2→Step 4) might decrease, which leads to reduce the computational time much. So the lumping method would be applicable to a problem with proper size of state space.

5. Concluding Remarks

The lumping method enables us to calculate stationary probabilities of a model even with a fairly large state space. Especially it is very efficient for a model of which transition rate matrix is block tridiagonal([4]). However, in solving a Markov model with a general form of infinitesimal generator, usually this method requires a great number of iterations and takes much computational times, even though it makes possible to get probabilities. In our model, fortunately, the marginal probability vector $\underline{\pi}$ is available, which makes it handy to apply the method.

In another paper [3], we tried to partition the state space into groups indexed by i_1 only. So the number of groups is S_1+1 and each group size is $(S_2+1)(S_3+1)(S_3+2)/2$. Grouping this way, the generator Q is represented as a block tridiagonal matrix. This generator is similar to one of Ph/M loss system, but those are slightly different. So we cannot apply the method of Neuts[2] directly. We transformed Q into a block didiagonal matrix, which enable us easily to solve that system of linear equations. However, largeness of block size requires large memory of computer.

Machihara[1] studies a similar problem appeared in telephone exchange network. He concentrates to analyze the phenomena happened at commonly usable servers, into which two types of interruped Poisson streams arrive. The approach is reported very efficient to get distributions.

Acknowledgements

The authors sincerely express their thanks to Professor Y. Takahashi of Tohoku University for his suggesting the problem and troublesome testing of our computer program using ACOS 1000. And they are also grateful to the refrees for their kind comments.

References

[1] Machihara, F.; Ph +Ph /M/S/K no Afure-ko ni tsuite. *Abstract of Autumn meeting, Operations Res. Soc. of Japan,* (1981) (in Japanese).

[2] Neuts, M. F.: *Matrix-Geometric Solutions in Stochastic Models —— An algorithmic Approach.* The Johns Hopkins University Press, 1981.

[3] Sato, T. and M. Mori: Kyoyo-gata Madoguchi no Koka ni tsuite. *Seminar Report of Institute of Mathematical Analysis at Kyoto University,* No.452 (1982) (in Japanese)

[4] Takahashi, Y.: A Lumping Method for Numerical Calculations of Stationary Distributions of Markov Chains. *Res. Rep. on Information Sciences B-18* (1975), Dept. of Information Science, Tokyo Institute of Technology.

[5] Takahashi, Y. and Y. Takami: A Numerical Method for the Steady-State Probabilities of a GI/G/C Queueing System in a General Class. *J. Operations Soc. of Japan,* vol.19 (1976), 147-157.

Masao MORI: Dept. of Management Science, Tokyo Institute of Technology, Oh-okayama, Meguro-ku Tokyo 152, Japan.