Iterative Methods for Queueing Systems and Markov Chains

Wai-Ki CHING

Department of Mathematics

University of Hong Kong

Abstract

Markovian queueing systems are widely used in the modeling of telecommunication systems, manufacturing systems, inventory systems and many other practical systems. Very often, in the system performance analysis, one faces the problem of solving the system steady-state probability distribution of a large number of states. Fast numerical algorithms based on Preconditioned Conjugate Gradient (PCG) method will be presented to solve the problem. Other efficient iterative methods for solving Markov chains will also be discussed.

Introduction

(1) The Birth-and-Death Process.

(2) Examples of Markovian Queueing Systems.

(3) Numerical Algorithm (Preconditioned Conjugate Gradient Method).

(4) Numerical Examples.

(5) Queueing Systems and Markov Chains.

(6) Other Iterative Methods.

1. Birth-and-Death Processes

- We begin our discussion on the theory of birth-and-death processes. The analysis of which is relatively simple and has important applications in queueing theory.

- Let us consider a system that can be represented by a family of random variables $\{N(t)\}$ parameterized by the time variable t. This is called a stochastic process.

- In particular, let us assume that for each t, $N(t)$ is a non-negative integral-valued random variable. For example, in a queue, where $N(t)$ is the number of customers waiting or in service at time t.

- We say that the system is in state E_j at time t if $N(t) = j$. Our aim is then to compute the state probabilities

$$\boxed{P\{N(t) = j\}, \quad j = 0, 1, 2, \cdots .}$$

Definition: A process is called a birth-and-death process if at ant time t

- (1) $P\{E_j \to E_{j+1}$ during $(t, t+h) | E_j$ at $t\} = \lambda_j h + o(h)$ as $h \to 0$ $(j = 0, 1, 2, \cdots)$. Here λ_j (birth rate) is a constant depending on j.

- (2) $P\{E_j \to E_{j-1}$ during $(t, t+h) | E_j$ at $t\} = \mu_j h + o(h)$ as $h \to 0$ $(j = 1, 2, \cdots)$. Here μ_j (death rate) is a constant depending on j.

- (3) $P\{E_j \to E_{j\pm k}$ during $(t, t+h) | E_j$ at $t\} = o(h)$ as $h \to 0$ if $k \geq 2$ $(j = 0, 1, \cdots)$.

Remark: $o(h)$ is a function of h such that

$$\boxed{\lim_{h \to 0} \frac{o(h)}{h} = 0.}$$

Possible examples of $o(h)$ are $o(h) = h^2$ and $o(h) = h \sin(h)$. However, $o(h)$ cannot take the form $\sqrt{h}$ or $h \log(h)$.

Notation: Let $P_j(t) = P\{N(t) = j\}$ and let $\lambda_{-1} = \mu_0 = P_{-1}(t) = 0$.

- It follows from the above three postulates that (where $h \to 0; j = 0, 1, \ldots.$)

$$P_j(t+h) = \underbrace{(\lambda_{j-1}h + o(h))}_{an\ \ arrival} P_{j-1}(t) + \underbrace{(\mu_{j+1}h + o(h))}_{a\ \ departure} P_{j+1}(t) + \underbrace{[1 - ((\lambda_j + \mu_j)h + o(h))]}_{no\ \ arrival\ \ or\ \ departure} P_j(t)$$

$$\boxed{P_j(t+h) = (\lambda_{j-1}h)P_{j-1}(t) + (\mu_{j+1})hP_{j+1}(t) + [1 - (\lambda_j + \mu_j)h]P_j(t) + o(h)}.$$

- Re-arranging terms, we have the following:

$$\boxed{\frac{P_j(t+h) - P_j(t)}{h} = \lambda_{j-1}P_{j-1}(t) + \mu_{j+1}P_{j+1}(t) - (\lambda_j + \mu_j)P_j(t) + \frac{o(h)}{h}.}$$

Letting $h \to 0$, we get the following set of differential-difference equations

$$\frac{d}{dt}P_j(t) = \lambda_{j-1}P_{j-1}(t) + \mu_{j+1}P_{j+1}(t) - (\lambda_j + \mu_j)P_j(t). \qquad (1)$$

- If at time $t = 0$ the system is in State E_i, the initial conditions are then $P_j(0) = \delta_{ij}$ where

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j. \end{cases}$$

- Here the coefficients $\{\lambda_j\}$ and $\{\mu_j\}$ are called the birth and death rates respectively.

- When $\mu_j = 0$ for all j, the process is called a pure birth process; and when $\lambda_j = 0$ for all j, the process is called a pure death process.

- We remark that in the case of either a pure birth process or a pure death process, the equations (1) can be solved by recurrence.

1.1 Pure Birth Process with Constant Rates

- We consider a Pure birth process ($\mu_i = 0$) with constant birth rate $\lambda_j = \lambda$ and initial state E_0.

- The equations in (1) become

$$\boxed{\frac{d}{dt}P_j(t) = \lambda P_{j-1}(t) - \lambda P_j(t) \quad (j = 0, 1, \cdots)}$$

where $P_{-1}(t) = 0$ and $P_j(0) = \delta_{0j}$.

- Here

$$\boxed{j = 0, \quad P_0'(t) = -\lambda P_0(t)}$$

Hence

$$\boxed{P_0(t) = a_0 e^{-\lambda t}.}$$

- From the initial conditions, we get $a_0 = 1$.

- Inductively, we can prove that if

$$\boxed{P_{j-1}(t) = \frac{(\lambda t)^{j-1}}{(j-1)!} e^{-\lambda t}}$$

then the equation

$$\boxed{P_j'(t) = \lambda \left(\frac{(\lambda t)^{j-1}}{(j-1)!} e^{-\lambda t} \right) - \lambda P_j(t)}$$

gives (exercise) the solution (Poisson Distribution)

$$\boxed{P_j(t) = \frac{(\lambda t)^j}{j!} e^{-\lambda t}.} \qquad (2)$$

Theorem: Suppose in a certain process, we let T_i $(i = 1, 2, 3, \cdots)$ be the epoch of the i^{th} occurrence.

- Let $A_i = T_i - T_{i-1}$ $(i = 1, 2, 3, \cdots)$; T_0 = epoch that we start to count the number of occurrences.

- Let $N(t)$ = number of occurrences in a time interval of length t. Then the following statements are equivalent.

(a) The process is Poisson (with coefficient λ).
(b) $N(t)$ is a Poisson random variable with parameter λt, i.e.

$$P\{N(t) = j\} = \frac{(\lambda t)^j}{j!} e^{-\lambda t} \ , \quad j = 0, 1, 2, \ldots .$$

(c) A_i's are mutually independent identically distributed exponential random variables with mean λ^{-1}, i.e.

$$P\{A_i \leq t\} = 1 - e^{-\lambda t} \ , \quad i = 1, 2, \cdots .$$

1.2 Queueing System and Birth-and-Death Process

- We consider a queueing system with one server and no waiting position, with

$$P\{\text{one customer arriving during } (t, t+h)\} = \lambda h + o(h)$$

and

$$P\{\text{service ends in } (t, t+h)|\text{ server busy at } t\} = \mu h + o(h).$$

It is a two-state birth-and-death process with $j = 0, 1$.

- The arrival rates (birth rates) are $\lambda_0 = \lambda$ and $\lambda_j = 0$ for $j \neq 0$ and the departure rates (death rates) are $\mu_j = 0$ when $j \neq 1$ and $\mu_1 = \mu$.

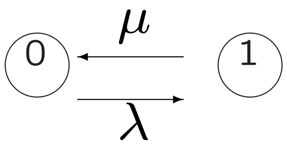


Figure 1.1: The Two-state Birth and Death Process.

- The equations for the birth-and-death process are given by

$$\boxed{\frac{d}{dt}P_0(t) = -\lambda P_0(t) + \mu P_1(t) \quad \text{and} \quad \frac{d}{dt}P_1(t) = \lambda P_0(t) - \mu P_1(t).} \tag{3}$$

One convenient way of solving this set of simultaneous linear differential equations is as follows:

- Adding the equations in (3), we get

$$\frac{d}{dt}[P_0(t) + P_1(t)] = 0,$$

hence $\quad P_0(t) + P_1(t) = \text{constant}.$

- Initial conditions are $P_0(0) + P_1(0) = 1$; thus $P_0(t) + P_1(t) = 1$. Hence we get

$$\boxed{\frac{d}{dt}P_0(t) + (\lambda + \mu)P_0(t) = \mu.}$$

- The solution (exercise) called the transient solution is given by

$$P_0(t) = \frac{\mu}{\lambda + \mu} + (P_0(0) - \frac{\mu}{\lambda + \mu})e^{-(\lambda+\mu)t}.$$

Since $P_1(t) = 1 - P_0(t)$,

$$P_1(t) = \frac{\lambda}{\lambda + \mu} + (P_1(0) - \frac{\lambda}{\lambda + \mu})e^{-(\lambda+\mu)t}. \tag{4}$$

- Consider the state probabilities of the above example when $t \to \infty$, from (4) we have

$$\begin{cases} P_0 & = & \lim_{t\to\infty} P_0(t) = \frac{\mu}{\lambda + \mu} \\ P_1 & = & \lim_{t\to\infty} P_1(t) = \frac{\lambda}{\lambda + \mu}. \end{cases} \tag{5}$$

- We note that $P_0 + P_1 = 1$ and they are called the steady-state probabilities of the system.

Note:

(i) P_0 and P_1 are independent of the initial values $P_0(0)$ and $P_1(0)$.

(ii) P_0 and P_1 satisfy the system of linear equations (generator matrix)

$$\boxed{\begin{pmatrix} -\lambda & \mu \\ \lambda & -\mu \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \text{and} \quad P_0 + P_1 = 1.}$$

- This leads us to the important notion of statistical equilibrium or steady-state. We say that a system is in statistical equilibrium (or stationary) if its state probabilities are constant in time.

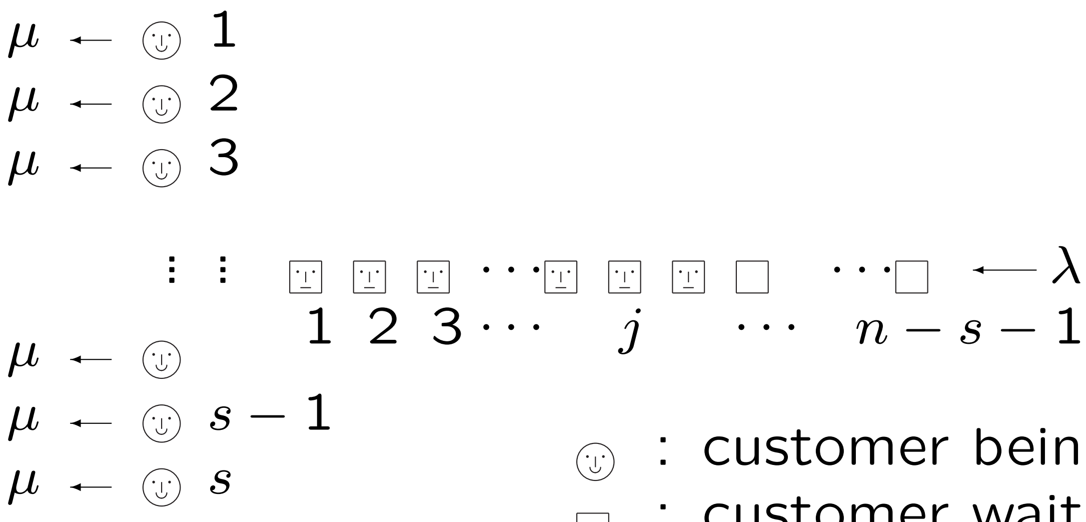
- Note that the system still fluctuate from state to state, unless $P_0(0) = P_0$ or $P_1(0) = P_1$.

- **S. Ross**, *Introduction to Probability Models*, Prentice-Hall, (2003).

2. Examples of Markovian Queueing Systems.

2.1 Single Markovian Queue (M/M/s/n-s-1)

- λ, input rate (Arrival Rate),
- μ, output rate (Service Rate, Production Rate etc).
- Set of possible states (number of customers): $\{0, 1, \ldots, n-1\}$.

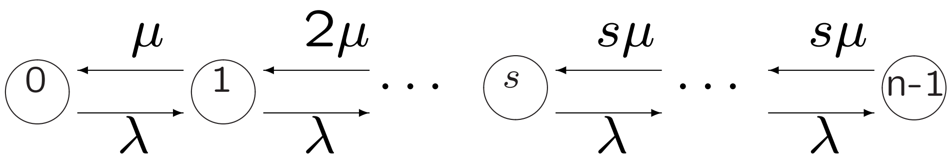


☺ : customer being served

⊡ : customer waiting in queue

□ : empty buffer in queue

2.1.1 The Steady-state Probability Distribution



The Transition Diagram (Markov Chain) of the M/M/s/n-s-1 Queue

- Let p_i be the steady-state probability that there are i customers in the queueing system.

- $\mathbf{p} = (p_0, \ldots, p_{n-1})^t$ is the steady-state probability vector.

- Important for system performance analysis, e.g. average waiting time of the customers in long run.

- **B. Bunday**, *Introduction to Queueing Theory*, Arnold, N.Y., (1996).

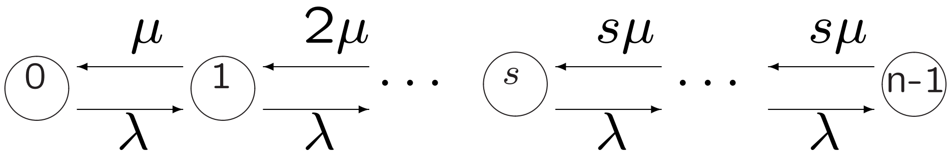
- The steady-state probability p_i is governed by the Kolmogorov equations which can be obtained by equating the expected outgoing rate and the expected incoming rate at each of the state:

Out-going Rate

$$\begin{tikzcd}
p_{i-1} & p_i \arrow[l, "i\mu"] \arrow[r, "\lambda"] & p_{i+1}
\end{tikzcd}$$

Incoming Rate

$$\begin{tikzcd}
p_{i-1} \arrow[r, "\lambda"] & p_i & p_{i+1} \arrow[l, "(i+1)\mu_1"]
\end{tikzcd}$$



The Transition Diagram (Markov Chain) of the M/M/s/n-s-1 Queue

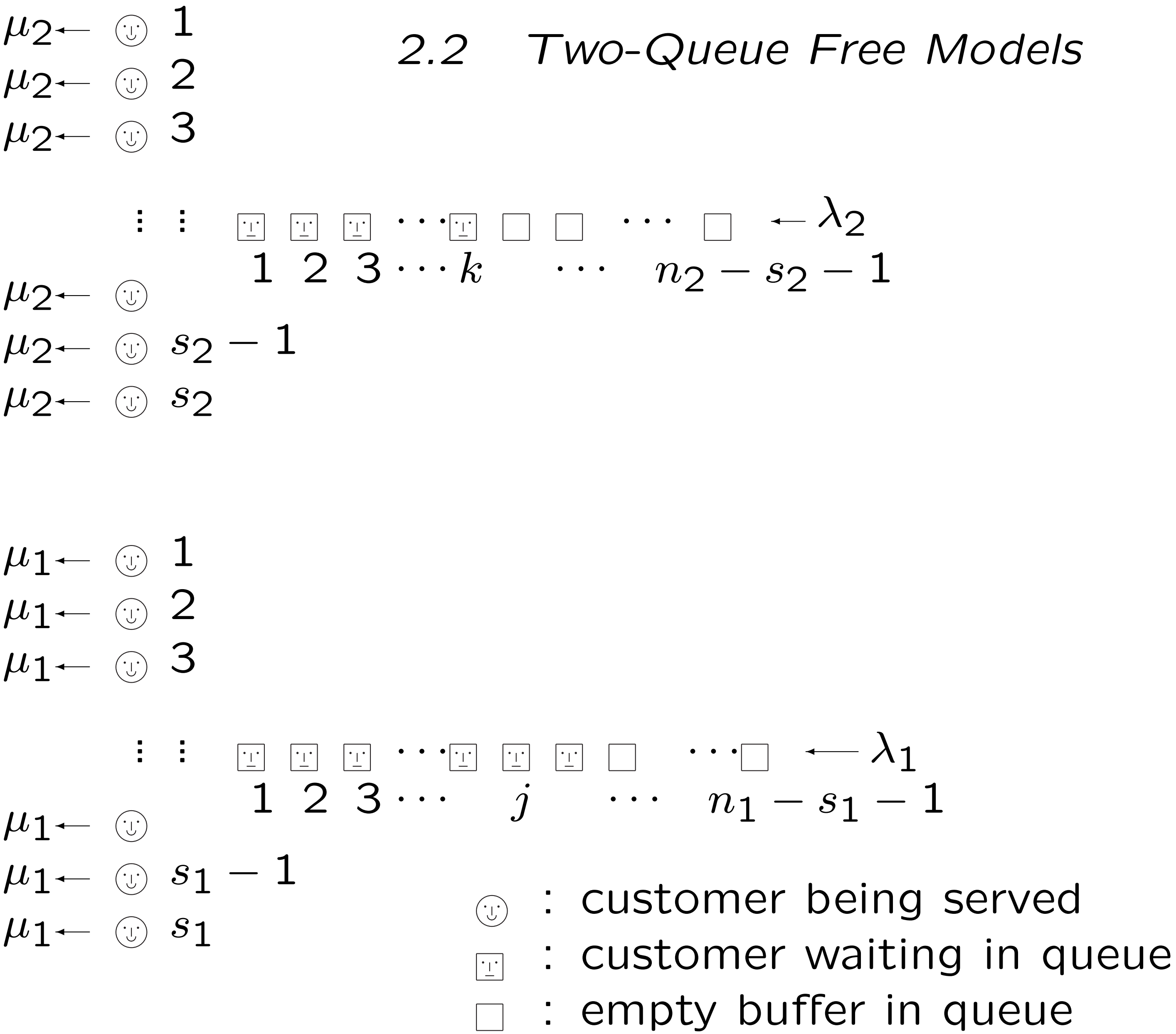
- We are solving:

$$\left\{\begin{array}{cl} A_0\mathbf{p_0} & = \mathbf{0}, \\ \sum p_i & = 1, \\ p_i & \geq 0. \end{array}\right.$$

- A_0, the generator matrix, is given by the $n \times n$ tridiagonal matrix:

$$A_0 = \begin{pmatrix} \lambda & -\mu & & & & & & 0 \\ -\lambda & \lambda+\mu & -2\mu & & & & & \\ & -\lambda & \lambda+2\mu & -3\mu & & & & \\ & & \cdot & \cdot & \cdot & & & \\ & & -\lambda & \lambda+s\mu & -s\mu & & & \\ & & & \cdot & \cdot & \cdot & & \\ & & & & -\lambda & \lambda+s\mu & -s\mu & \\ 0 & & & & & -\lambda & s\mu & \end{pmatrix}.$$

2.2 Two-Queue Free Models



☺ : customer being served

▣ : customer waiting in queue

□ : empty buffer in queue

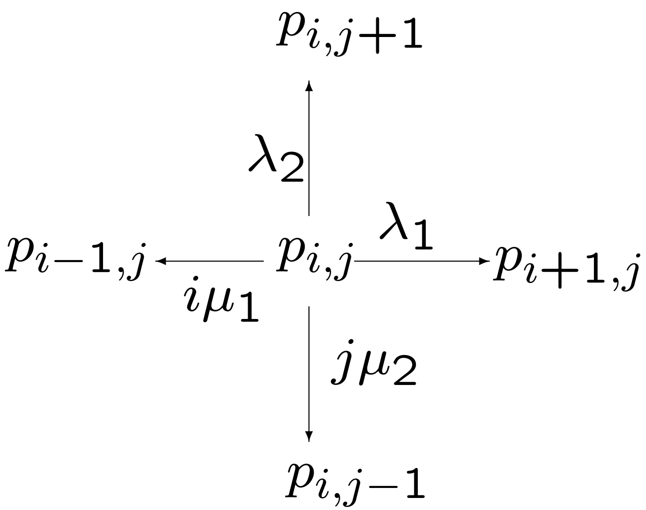
- Let $p_{i,j}$ be the probability that there are i customers in queue 1 and j customers in queue 2.

- Set of possible states (number of customers in each queue):

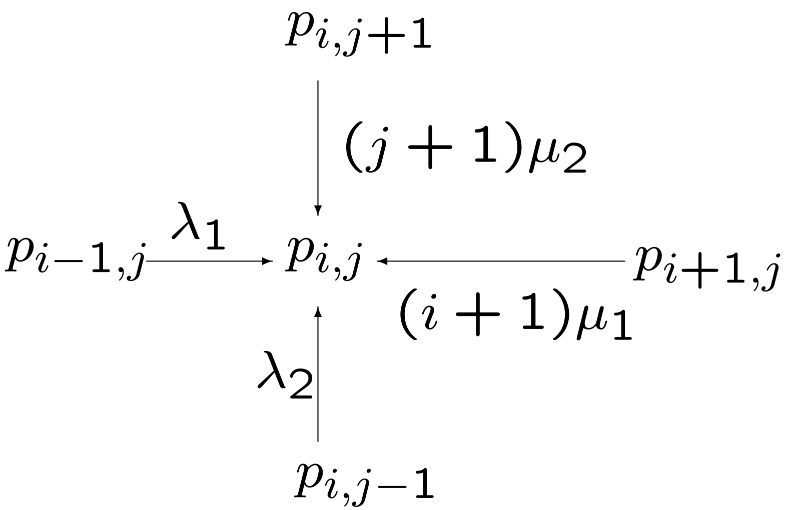
$$\{(i,j) : i = 0,1,\ldots,n_1, \quad j = 0,1,\ldots,n_2\}.$$

- The Kolmogorov equations for the two-queue network:

Out-going Rate



Incoming Rate



- Again we have to solve

$$\begin{cases} A_1\mathbf{p} & = \mathbf{0}, \\ \sum p_{ij} & = 1, \\ p_{ij} & \geq 0. \end{cases}$$

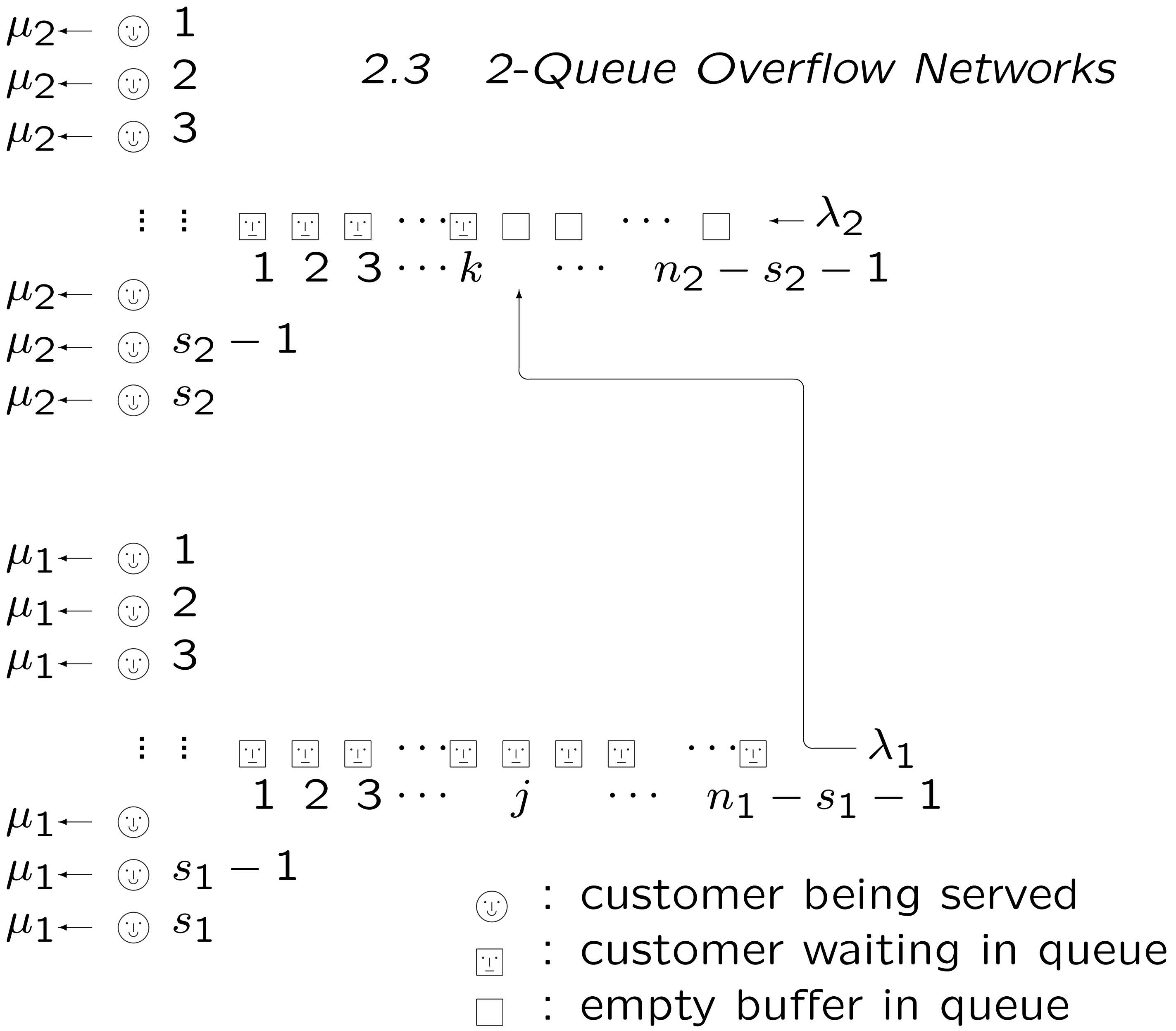
- The generator matrix A_1 is separable (no interaction between the queues): $A_1 = A_0 \otimes I + I \otimes A_0$.

- Kronecker tensor product of two matrices $A_{n\times r}$ and $B_{m\times k}$:

$$A_{n\times r} \otimes B_{m\times k} = \begin{pmatrix} a_{11}B & \cdots & \cdots & a_{1n}B \\ a_{21}B & \cdots & \cdots & a_{2n}B \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1}B & \cdots & \cdots & a_{mn}B \end{pmatrix}_{nm\times rk} .$$

- It is easy to check that the underlying Markov chain of the queueing system is irreducible and the unique solution is $\mathbf{p} = \mathbf{p_0} \otimes \mathbf{p_0}$.

2.3 2-Queue Overflow Networks


μ_2
1
2
3
$s_2 - 1$
s_2
λ_2
$1\ 2\ 3 \cdots k \quad \cdots \quad n_2 - s_2 - 1$
μ_1
1
2
3
$s_1 - 1$
s_1
λ_1
$1\ 2\ 3 \cdots \quad j \quad \cdots \quad n_1 - s_1 - 1$

☺ : customer being served

▣ : customer waiting in queue

□ : empty buffer in queue

- **L. Kaufman**, SIAM J. Sci. Statist. Comput., 4 (1982).

- The generator matrix A_2 is given by

$$A_2 = A_0 \otimes I + I \otimes A_0 + \begin{pmatrix} \mathbf{0} & \\ & 1 \end{pmatrix} \otimes R_0,$$

where

$$R_0 = \lambda_1 \begin{pmatrix} 1 & & & & 0 \\ -1 & 1 & & & \\ & \ddots & \ddots & & \\ & & -1 & 1 & \\ 0 & & & -1 & 0 \end{pmatrix}$$

describes the overflow discipline of the queueing system.

- In fact, we may write

$$A_2 = A_1 + \begin{pmatrix} \mathbf{0} & \\ & 1 \end{pmatrix} \otimes R_0.$$

- Unfortunately analytic solution for the steady-state distribution $\mathbf{p}$ is not available.

- The generator matrices are sparse and have block structures.

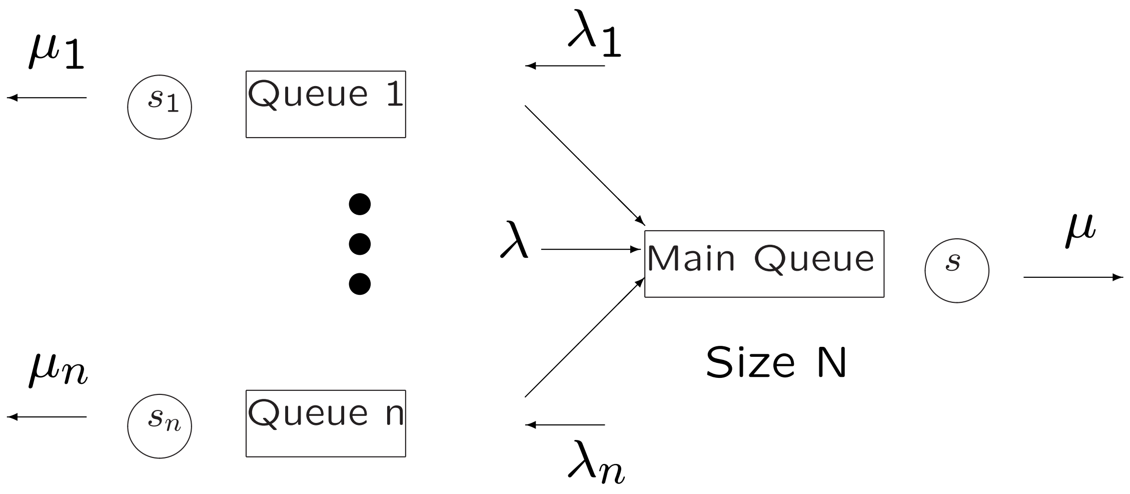
- Direct method (LU decomposition will result in dense matrices L and U) is not efficient in general.

- Fast algorithm should make use of the block structures and the sparsity of the generator matrices.

- Block Gauss-Seidel (BGS) is an usual approach for mentioned queueing problems. Its convergence rate is not fast and increase linearly with respect to the size of the generator matrix in general.

- **R. Varga**, *Matrix Iterative Analysis*, Prentice-Hall, N.J., (1963).

2.4 The Telecommunication System



- **K. Hellstern**, *The Analysis of a Queue Arising in Overflow Models*, IEEE Trans. Commun., 37 (1989).

- **W. Ching, R. Chan and X. Zhou**, *Circulant Preconditioners for Markov Modulated Possion Processes and Their Applications to Manufacturing Systems*, SIAM J. Matrix Anal., 17 (1997).

- We may regard the telecommunication network as a (MMPP/M/$s/s+N$) queueing system.

- An MMPP is a Poisson Process whose instantaneous rate itself is a stationary random process which varies according to an irreducible n-state Markov chain (When $n = 1$, it is just the Poisson Process).

- Important in analysis of blocking probability and system utilization.

- **M. Neuts**, *Matrix-Geometric Solutions in Stochastic Models*, Johns Hopkins University Press, M.D., (1981).

- **J. Flood**, *Telecommunication Switching Traffic and Networks*, Prentice-Hall, N.Y., (1995).

Generator matrix is given by:

$$A_3 = \begin{pmatrix} Q+\Gamma & -\mu I & & & & 0 \\ -\Gamma & Q+\Gamma+\mu I & -2\mu I & & & \\ & \ddots & \ddots & \ddots & & \\ & & -\Gamma & Q+\Gamma+s\mu I & -s\mu I & \\ & & \ddots & \ddots & \ddots & \\ & & & -\Gamma & Q+\Gamma+s\mu I & -s\mu I \\ 0 & & & & -\Gamma & Q+s\mu I \end{pmatrix},$$

($(N+1)$-block by $(N+1)$-block), where

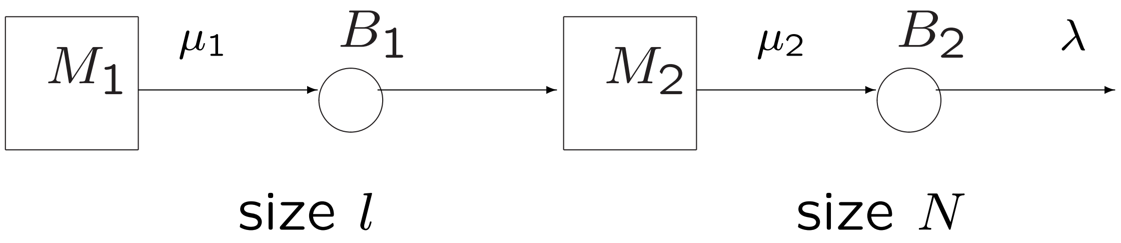
$$\Gamma = \Lambda + \lambda I_{2^n},$$

$$Q = (Q_1 \otimes I_2 \otimes \cdots \otimes I_2) + (I_2 \otimes Q_2 \otimes I_2 \otimes \cdots \otimes I_2) + \cdots + (I_2 \otimes \cdots \otimes I_2 \otimes Q_n),$$

$$\Lambda = (\Lambda_1 \otimes I_2 \otimes \cdots \otimes I_2) + (I_2 \otimes \Lambda_2 \otimes I_2 \otimes \cdots \otimes I_2) + \cdots + (I_2 \otimes \cdots \otimes I_2 \otimes \Lambda_n),$$

$$Q_j = \begin{pmatrix} \sigma_{j1} & -\sigma_{j2} \\ -\sigma_{j1} & \sigma_{j2} \end{pmatrix} \qquad \text{and} \qquad \Lambda_j = \begin{pmatrix} \lambda_j & 0 \\ 0 & 0 \end{pmatrix}.$$

2.5 The Manufacturing System of Two Machines in Tandem



- Search for optimal buffer sizes l and N $(N >> l)$, which minimizes (1) the average running cost, (2) maximizes the throughput, or (3) minimizes the blocking and the starving rate.

- **G. Yamazaki, T. Kawashima and H. Sakasegawa**, *Reversibility of Tandem Blocking Queueing Systems*, Manag., Sci., 31 (1985).

- **W. Ching**, *Iterative Methods for Manufacturing Systems of Two Stations in Tandem*, Applied Maths. Letters, 11 (1998).

The generator matrix is of the form:

$$A_4 = \begin{pmatrix} \Lambda + \mu_1 I & -\Sigma & & & 0 \\ -\mu_1 I & \Lambda + D + \mu_1 I & -\Sigma & & \\ & \ddots & \ddots & \ddots & \\ & & -\mu_1 I & \Lambda + D + \mu_1 I & -\Sigma \\ 0 & & & -\mu_1 I & \Lambda + D \end{pmatrix},$$

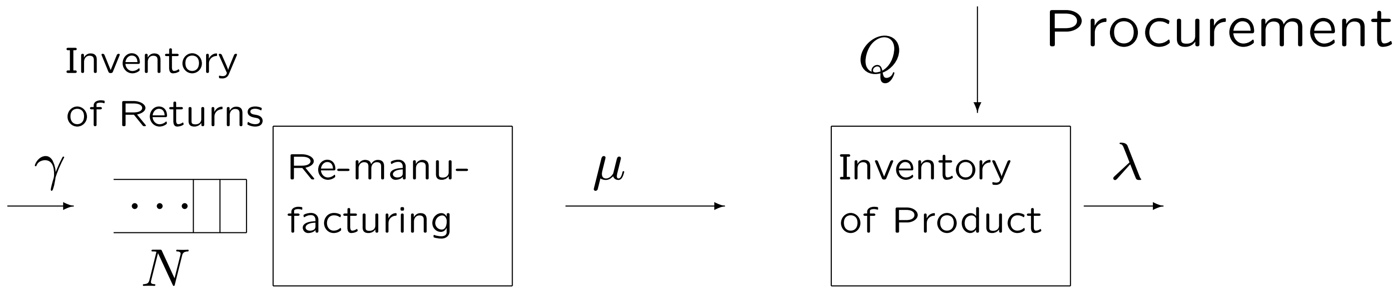
($(l+1)$-block by $(l+1)$-block), where

$$\Lambda = \begin{pmatrix} 0 & -\lambda & & 0 \\ & \lambda & \ddots & \\ & & \ddots & -\lambda \\ 0 & & & \lambda \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 0 & & & 0 \\ \mu_2 & \ddots & & \\ & \ddots & \ddots & \\ 0 & & \mu_2 & 0 \end{pmatrix},$$

and

$$D = \mathrm{Diag}(\mu_2, \cdots, \mu_2, 0).$$

2.6 *The Re-Manufacturing System*



- There are two types of inventory to manage: the serviceable product and the returned product. The re-cycling process is modelled by an M/M/1/N queue.

- The serviceable product inventory level and the outside procurements are controlled by an (r, Q) continuous review policy. Here r is the outside procurement level and Q is the procurement quantity. We assume that $N >> Q$.

- **M. Fleischmann**, *Quantitative Models for Reverse Logistics*, (501) LNEMS, Berlin, Springer (2001).

- **W. Ching and W. Yuen**, *Iterative Methods for Re-manufacturing Systems*, Int. J. Appl. Maths (2002).

- The generator matrix is given by

$$A_5 = \begin{pmatrix} B & -\lambda I & & & 0 \\ -L & B & -\lambda I & & \\ & \ddots & \ddots & \ddots & \\ & & -L & B & -\lambda I \\ -\lambda I & & & -L & B_Q \end{pmatrix},$$

where

$$L = \begin{pmatrix} 0 & \mu & & & 0 \\ & 0 & \ddots & & \\ & & \ddots & \ddots & \\ & & & \ddots & \mu \\ 0 & & & & 0 \end{pmatrix}, \quad B = \lambda I_{N+1} + \begin{pmatrix} \gamma & & & & 0 \\ -\gamma & \gamma+\mu & & & \\ & \ddots & \ddots & & \\ & & \ddots & \gamma+\mu & \\ 0 & & & -\gamma & \mu \end{pmatrix},$$

and

$$B_Q = B - \mathrm{Diag}(0, \mu, \ldots, \mu).$$

3 Numerical Algorithm (Preconditioned Conjugate Gradient (PCG) Method)

- Conjugate Gradient (CG) Method to solve $A\mathbf{x} = \mathbf{b}$.

- Need preconditioning to accelerate convergence rate.

- **O. Axelsson**, *Iterative Solution Methods*, Cambridge University Press, (1996).

- In the Preconditioned Conjugate Gradient(PCG) method with preconditioner C, CG method is applied to solve

$$C^{-1}A\mathbf{x} = C^{-1}\mathbf{b}$$

instead of

$$A\mathbf{x} = \mathbf{b}.$$

- A Good preconditioner C is a matrix satisfying:

(a) Easy and fast to construct;

(b) The preconditioner system $C\mathbf{x} = \mathbf{f}$ can be solved very fast;

(c) The preconditioned matrix $C^{-1}A$ has singular values clustered around one[1].

Note:
(1) One sufficient condition for a sequence of matrices B_n (size $n \times n$) has singular values clustered around 1 : the number of singular values of B_n different from 1 is bounded above and independent of the matrix size of B_n.

3.1 Circulant-based Preconditioners

- Circulant matrices are Toeplitz matrices (constant diagonal entries) such that each column is a cyclic shift of its preceding column.

- Class of circulant matrices denoted by $\mathcal{F}$.

- $C \in \mathcal{F}$ implies C can be diagonalized by Fourier matrix F:

$$C = F^* \Lambda F.$$

Hence

$$C^{-1}\mathbf{x} = F^* \Lambda^{-1} F\mathbf{x}.$$

- Eigenvalues of a circulant matrix has analytic form, therefore enhance the spectrum analysis of the preconditioned matrix.

- $C^{-1}\mathbf{x}$ can be done in $O(n \log n)$ where n is the size of the matrix.

- **P. Davis**, *Circulant Matrices*, John Wiley and Sons, N.J. (1985).

3.2 Our Circulant-based Preconditioner

$$A = \begin{pmatrix} \lambda & -\mu & & & & & 0 \\ -\lambda & \lambda+\mu & -2\mu & & & & \\ & \cdot & \cdot & \cdot & & & \\ & & -\lambda & \lambda+s\mu & -s\mu & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & -\lambda & \lambda+s\mu & -s\mu \\ 0 & & & & & -\lambda & s\mu \end{pmatrix}.$$

$$s(A) = \begin{pmatrix} \lambda+s\mu & -s\mu & & & & & -\lambda \\ -\lambda & \lambda+s\mu & -s\mu & & & & \\ & \cdot & \cdot & \cdot & & & \\ & & -\lambda & \lambda+s\mu & -s\mu & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & -\lambda & \lambda+s\mu & -s\mu \\ -s\mu & & & & & -\lambda & \lambda+s\mu \end{pmatrix}.$$

We have rank$(A - s(A)) = s + 1$.

3.2.1 *The Telecommunication System*

•$A_3 = I \otimes Q + A \otimes I + R \otimes \Lambda$ where.

$$R = \begin{pmatrix} 1 & & & & 0 \\ -1 & 1 & & & \\ & -1 & \ddots & & \\ & & \ddots & 1 & \\ 0 & & & -1 & 0 \end{pmatrix}.$$

•$s(A_3) = s(I) \otimes Q + s(A) \otimes I + s(R) \otimes \Lambda$.

$$s(I) = I \quad \text{and} \quad s(R) = \begin{pmatrix} 1 & & & & -1 \\ -1 & 1 & & & \\ & -1 & \ddots & & \\ & & \ddots & 1 & \\ 0 & & & -1 & 1 \end{pmatrix}.$$

3.2.2 *The Manufacturing System of Two Machines in Tandem*

- Circulant-based approximation of A_4 : $s(A_4) =$

$$\begin{pmatrix} s(\Lambda)+\mu_1 I & -s(\Sigma) & & & 0 \\ -\mu_1 I & s(\Lambda)+s(D)+\mu_1 I & -s(\Sigma) & & \\ & \ddots & \ddots & \ddots & \\ & & -\mu_1 I & s(\Lambda)+s(D)+\mu_1 I & -s(\Sigma) \\ 0 & & & -\mu_1 I & s(\Lambda)+s(D) \end{pmatrix},$$

($(l+1)$-block by $(l+1)$-block), where

$$s(\Lambda) = \begin{pmatrix} \lambda & -\lambda & & 0 \\ & \lambda & \ddots & \\ & & \ddots & -\lambda \\ -\lambda & & & \lambda \end{pmatrix}, \quad s(\Sigma) = \begin{pmatrix} 0 & & & \mu_2 \\ \mu_2 & \ddots & & \\ & \ddots & \ddots & \\ 0 & & \mu_2 & 0 \end{pmatrix},$$

and

$$s(D) = \mathrm{Diag}(\mu_2, \cdots, \mu_2, \mu_2).$$

3.2.3 The Re-manufacturing System

- circulant-based approximation of A_5:

$$s(A_5) = \begin{pmatrix} s(B) & -\lambda I & & & 0 \\ -s(L) & s(B) & -\lambda I & & \\ & \ddots & \ddots & \ddots & \\ & & -s(L) & s(B) & -\lambda I \\ -\lambda I & & & -s(L) & s(B_Q) \end{pmatrix},$$

where

$$s(L) = \begin{pmatrix} 0 & \mu & & & 0 \\ & 0 & \ddots & & \\ & & \ddots & \ddots & \\ & & & \ddots & \mu \\ \mu & & & & 0 \end{pmatrix}, \quad s(B) = \lambda I_{N+1} + \begin{pmatrix} \gamma+\mu & & & -\gamma \\ -\gamma & \gamma+\mu & & \\ & \ddots & \ddots & \\ 0 & & -\gamma & \gamma+\mu \end{pmatrix},$$

and

$$s(B_Q) = s(B) - \mu I.$$

3.4 Stochastic Automata Networks

- In fact, all the generator matrices A take the form

$$A = \sum_{i=1}^{m} \bigotimes_{j=1}^{n} A_{ij},$$

where A_{i1} is relatively huge in size.

- Our preconditioner is defined as

$$C = \sum_{i=1}^{m} s(A_{i1}) \bigotimes_{j=2}^{n} A_{ij}.$$

- We note that

$$[F \bigotimes_{j=2}^{n} I]^* C [F \bigotimes_{j=2}^{n} I] = \sum_{i=1}^{m} \Lambda_{i1} \bigotimes_{j=2}^{n} A_{ij} = \bigoplus_{k=1}^{\ell} \left[\sum_{i=1}^{m} \lambda_{i1}^{k} \bigotimes_{j=2}^{n} A_{ij} \right]$$

which is a block-diagonal matrix.

- One of the advantages of our preconditioner is that it can be inverted in parallel by using a parallel computer easily. This would therefore save a lot of computational cost.

- Theorem: If all the parameters stay fixed then the preconditioned matrix has singular values clustered around one. Thus we expect our PCG method converges very fast.

- $A_{i1} \approx$ Toeplitz except for rank $(s+1)$ perturbation
  $\approx s(A_{i1})$ except for rank $(s+1)$ perturbation.

- **R. Chan and W. Ching**, *Circulant Preconditioners for Stochastic Automata Networks*, Numerise Mathematik, (2000).

4. Numerical Results

- Since generator A is non-symmetric, we used the generalized CG method, the Conjugate Gradient Squared (CGS) method. This method does not require the multiplication of $A^T\mathbf{x}$.

- Our proposed method is applied to the following systems.

(1) The Telecomunications System.

(2) The Manufacturing Systems of Two Machines in Tandem.

(3) The Re-Manufacturing System.

- **P. Sonneveld**, *A Fast Lanczos-type Solver for Non-symmetric Linear Systems* SIAM J. Sci. Comput., 10 (1989).

- Stopping Criteria: $\frac{||\mathbf{r}_n||_2}{||\mathbf{r}_0||_2} < 10^{-10}$; $||\mathbf{r}_n||_2 = n$th step residual.

4.1 The Telecomunications System

- n, number of external queues; N, size of the main queue.

- Cost per Iteration:

I	C	BGS
$O(n2^nN)$	$O(n2^nN \log N)$	$O((2^n)^2N)$

- Number of Iterations:

$s=2$	$n=1$			$n=4$		
N	I	C	BGS	I	C	BGS
32	155	8	171	161	13	110
64	$**$	7	242	$**$	13	199
128	$**$	8	366	$**$	14	317
256	$**$	8	601	$**$	14	530
512	$**$	8	$**$	$**$	14	958

- '$**$' means greater than 1000 iterations.

4.2 *The Manufacturing Systems of Two Machines in Tandem*

- l, size of the first buffer; N, size of the second buffer.

- Cost per Iteration:

I	C	BGS
$O(lN)$	$O(lN \log N)$	$O(lN)$

- Number of Iterations:

	$l=1$			$l=4$		
N	I	C	BGS	I	C	BGS
32	34	5	72	64	10	72
64	129	7	142	139	11	142
128	$**$	8	345	$**$	12	401
256	$**$	8	645	$**$	12	$**$
1024	$**$	8	$**$	$**$	12	$**$

- '$**$' means greater than 1000 iterations.

4.3 The Re-Manufacturing System

- Q, size of the serviceable inventory; N, size of the return inventory.

- Cost per iteration:

I	C	BGS
$O(QN)$	$O(QN \log N)$	$O(QN)$

- Number of Iterations:

	$Q=2$			$Q=3$			$Q=4$		
N	I	C	BGS	I	C	BGS	I	C	BGS
100	246	8	870	$**$	14	1153	$**$	19	1997
200	$**$	10	1359	$**$	14	$**$	$**$	19	$**$
400	$**$	10	$**$	$**$	14	$**$	$**$	19	$**$
800	$**$	10	$**$	$**$	14	$**$	$**$	19	$**$

- '$**$' means greater than 2000 iterations.

5. Queueing Systems and Markov Chains

- Given a generator matrix A of a queueing system:

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & \cdots & a_{nn} \end{pmatrix}$$

we have

$$a_{ii} = -\sum_{k=1,k\neq i}^{n} a_{ki} > 0 \quad \text{for} \quad i = 1, 2, \ldots, n.$$

- Now if we define a diagonal matrix

$$D = \text{Diag}(a_{11}, a_{22}, \ldots, a_{nn})$$

then the matrix $I - AD^{-1}$ is a transition probability matrix with each column sum being equal to one.

- To solve for

$$A\mathbf{x} = \mathbf{0}$$

it is equivalent to solve

$$AD^{-1}(D\mathbf{x}) = \mathbf{0}.$$

Let $\mathbf{y} = D\mathbf{x}$ then the above linear system is equivalent to the following

$$AD^{-1}\mathbf{y} = \mathbf{0}$$

or

$$(I - AD^{-1} - I)\mathbf{y} \equiv (P - I)\mathbf{y} = 0$$

or

$$P\mathbf{y} = \mathbf{y}$$

where P is a transition probability matrix of a certain Markov chain.

6. Other Iterative Methods
6.1 Power Method

- Power method is a popular method for solving the steady-state probability distribution of a Markov chain.

- In fact, the power method is an iterative method for solving the largest eigenvalue in modulus (the dominant eigenvalue) and its corresponding eigenvector.

- The idea of the power method can be briefly explained as follows. Given an $n \times n$ matrix A and suppose that there is a single eigenvalue of maximum modulus and the eigenvalues $\lambda_1, \lambda_2, \cdots, \lambda_n$ be labeled such that

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|.$$

For the case of transition probability matrix, $|\lambda| = 1$.

- Suppose further that there is a linearly independent set of n unit eigenvectors. This means that there is a basis

$$\left\{\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \ldots, \mathbf{u}^{(n)}\right\}$$

such that

$$A\mathbf{u}^{(i)} = \lambda_i \mathbf{u}^{(i)}, \quad i = 1, 2, \ldots, n, \quad \text{and} \quad \|\mathbf{u}^{(i)}\| = 1.$$

Then begin with an initial vector $\mathbf{x}^{(0)}$, one may write

$$\mathbf{x}^{(0)} = a_1\mathbf{u}^{(1)} + a_2\mathbf{u}^{(2)} + \cdots + a_n\mathbf{u}^{(n)}.$$

- Now we iterate the initial vector with the matrix A as follows:

$$\begin{aligned}
A^k\mathbf{x}^{(0)} &= a_1 A^k\mathbf{u}^{(1)} + \ldots + a_n A^k\mathbf{u}^{(n)} \\
&= a_1\lambda_1^k\mathbf{u}^{(1)} + \ldots + a_n\lambda_n^k\mathbf{u}^{(n)} \\
&= \lambda_1^k\left\{a_1\mathbf{u}^{(1)} + \left(\frac{\lambda_2}{\lambda_1}\right)^k a_n\mathbf{u}^{(2)} + \ldots + \left(\frac{\lambda_n}{\lambda_1}\right)^k a_n\mathbf{u}^{(n)}\right\} \\
&= a_1\mathbf{u}^{(1)} + \lambda_2^k a_n\mathbf{u}^{(2)} + \ldots + \lambda_n^k a_n\mathbf{u}^{(n)}.
\end{aligned}$$

Since

$$|\lambda_i| < 1 \quad \text{for } i = 2, \ldots, n$$

we have

$$\lim_{k\to\infty} |\lambda_i|^k = 0 \quad \text{for } i = 2, \ldots, n.$$

Hence we have

$$A^k \mathbf{x}^{(0)} \approx a_1 \lambda_1^k \mathbf{u}^{(1)}.$$

- To get an approximation for λ_1 and $\mathbf{u}^{(1)}$, we can introduce a normalization in the iteration:

$$\mathbf{r}_{k+1} = \frac{A^{k+1}\mathbf{x}^{(0)}}{\|A^k \mathbf{x}^{(0)}\|_2}$$

then we have

$$\lim_{k\to\infty} \mathbf{r}_{k+1} = \lim_{k\to\infty} \frac{a_1 \lambda_1^{k+1} \mathbf{u}^{(1)}}{\|a_1 \lambda_1^k \mathbf{u}^{(1)}\|_2} = \lambda_1 \mathbf{u}^{(1)}.$$

- The main computational cost of this method comes from the matrix-vector multiplication of the form $A\mathbf{x}$.

- It is clear from the above analysis that the convergence rate of the power method depends on the ratio of

$$|\lambda_2/\lambda_1|$$

where λ_1 and λ_2 are respectively the largest and the second largest eigenvalue in modulus of the matrix P.

- Since $\lambda_1 = 1$ in the Markov chain problem, so it only depends on $|\lambda_2|$.

6.2. Extrapolation Method

- The idea of extrapolation method here is, instead of solving the steady-state distribution $\mathbf{x}$ of a Markov chain satisfying $P\mathbf{x} = \mathbf{x}$ we consider the solution of the transition probability matrix:

$$M(c) = cP + (1 - c)\mathbf{u}\mathbf{1}^t \quad \text{where} \quad c \in [0, 1] \tag{6}$$

- Here $\mathbf{1}$ is the column vector of all ones and $\mathbf{u}$ is a given positive probability distribution vector. Therefore $\mathbf{u}\mathbf{1}^t$ is a rank one transition probability matrix. Moreover, $\mathbf{x} = \mathbf{x}(1)$.

- It is known that the second largest eigenvalue in modulus of the matrix (6) is bounded above by c.

- **S. Haveliwala and and S. Kamvar**, *The Second Eigenvalue of the Google Matrix*, Stanford University, Technical Report (2003).

- We recall that the convergence rate of the power method when apply to solving the steady-state probability distribution vector $\mathbf{x}(c)$ of

$$M(c)\mathbf{x}(c) = \mathbf{x}(c) \tag{7}$$

depends on c and the smaller the value of c, the faster the convergence rate will be.

- We note that it is very efficient to solve the steady-state probability distribution vectors of (7) for small values of c, say

$$\mathbf{x}(0.1), \mathbf{x}(0.2), \ldots, \mathbf{x}(0.5).$$

Extrapolation methods can then be developed to get an approximate for $\mathbf{x}(1)$.

- **C. Brezinski, M. Redivo-Zaglia and S. Serra-Capizzano**, *Extrapolation Methods for PageRank Computations*, C.R. Acad. Sci. Paris, Ser. I, 340 (2005).

Thank you for Your Attention

The Slides can be Obtained at

http://hkumath.hku.hk/ wkc/teaching.html