ABSTRACT

Peng, Bin. *Convergence, Rank Reduction and Bounds for the Stationary Analysis of Markov Chains* (Under the direction of William J. Stewart).

With existing numerical methods, the computation of stationary distributions for large Markov chains is still time-consuming, a direct result of the state explosion problem. In this thesis, we introduce a rank reduction method for computing stationary distributions of Markov chains for which low-rank iteration matrices can be formed. We first prove that, for an irreducible Markov chain, a necessary and sufficient condition for convergence in a single iteration is that the iteration matrix have rank one. Since most iteration matrices have rank larger than 1, we also consider the Wedderburn rank-1 reduction formula and develop a rank reduction procedure that takes an initial iteration matrix with rank greater than one and modifies it in successive steps, under the constraint that the exact solution be preserved at each step, until a rank-1 iteration matrix is obtained. When the iteration matrix has rank r, the proposed algorithm has time complexity $O(r^2n)$. Secondly we investigate the relationship among lumpability, weak lumpability, quasi-lumpability and near complete decomposability. These concepts are important in aggregating and disaggregating Markov chains. White's algorithm for identifying all possible lumpable partitions for Markov chains is improved by incorporating lumpability tests on special state orderings. Finally, instead of computing exact stationary distributions, we design stochastic-ordering-based techniques to bound them. Upper bounds can be obtained by using constructive algorithms developed recently. We observe that the more lumpable partitionings, the more accurate the upper bound for the state of interest both with matrix transformation and without matrix transformation. Lastly we combine the approaches of state permutation, matrix transformation and state partitioning to improve the quality of the upper bound for the state of interest.

Convergence, Rank Reduction and Bounds for the Stationary Analysis of Markov Chains

by

Bin Peng

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

OPERATIONS RESEARCH

Raleigh

2004

APPROVED BY:

Chair of Advisory Committee

Biography

Born in Hunan province, China, Bin Peng is one of two brothers and the elder son of Yuanchong Peng and Shengying Mo, both of whom are traditional Chinese farmers. Bin's only younger brother Ru was injured and suffered severe nerve damage in a coal mining accident in 1999 and disastrously lost forever his ability to walk and move. His parents are patiently taking care of him day after day, month after month and year after year.

In July of 1997, Bin Peng graduated from the Department of Mathematics, Xiangtan University in Hunan province, China with a B.S. degree in Economics. He pursued his graduate study in the Department of Statistics & Operations Research at Fudan University (Shanghai, China) and was awarded a Master of Science degree in Operations Research in 2000. Thereafter, he came to North Carolina State University to continue his studies toward his doctoral degree in Operations Research.

Bin Peng was married to Manhong Chai in 2000 and they have a son, Maxwell.

Acknowledgments

First of all, I wish to express my sincere thanks to my advisor, Dr. William J. Stewart for his insight guidance, endless patience and support. Without his expert insight, this research could not have been completed. His support in terms of a NSF graduate research assistant made it possible for me to complete my years of study at North Carolina State University. I am indebted to him.

I am also grateful to my committee members, Drs.Carl Meyer, Harry Perros and Russel King for their valuable times, professional suggestions and gracious services on my committee.

My most special thanks go to my wife Manhong and son Maxwell for their ever-lasting encouragement, support and love.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Since their introduction by the Russian mathematician A.A. Markov(1856-1922) in the early 1900s, Markov chains (MCs) have proven to be a valualbe tool with which to model the performance of complex stochastic systems. More and more applications of their use have been discovered. For example, they play an important role in search engine such as Google [59].

1.1 Motivation

In this thesis, we consider a finite, homogeneous, irreducible Markov chain with transition probability matrix P which possesses a unique stationary distribution vector $\pi > 0$. Usually three steps are followed in conducting a Markovian analysis.

1. Represent the system to be modeled by a high-level formalism such as stochastic Petri nets [16], queuing networks and stochastic automata networks [64] and generate the underlying Markov chains.

2. Solve the stationary distribution vector π.

3. Convert π to meaningful measurements of system performance such as average response time, mean number of customers in systems, mean times for systems failure.

Due to the *state space explosion problem*, the most challenging step in this process is the computation of the unique vector π which satisfies

$$\pi = \pi P, \quad \pi^T e = 1, \pi > 0 \tag{1.1}$$

where e is a column vector of all ones. Although there already exist many numerical methods [64] for solving (1.1), more efficient methods are required. In this dissertation, we design a rank reduction method to solve (1.1) so that the computation time can be lessened; When the computation of the exact solution for (1.1) is unnecessary, we present stochastic-ordering-based bounding techniques which generate an upper bound on the solution so that computation time is reduced to an absolute minimium.

1.2 Contributions

The contributions of the thesis are three fold. Just as the thesis title says, the first is a convergence behavior analysis for iterative numerical methods of computing stationary distributions of MCs. We find a necessary and sufficient condition for convergence in a single iteration for an irreducible M.C. That is, the iteration matrix has rank one. Furthermore we show the condition that the iteration matrix have its spectrum with one eigenvalue equal to 1 and all the others equal to zeros is not adquate.

The second part is the development and experiments of the rank reduction method. Since most iterative matrices have rank larger than 1, we consider the Wedderburn rank-1 reduction formula and design a rank reduction procedure with an initial iteration matrix having rank greater than one and modify it in successive steps, under the constraint that the exact solution be preserved at each step, until a rank-1 iteration matrix is obtained.

The third part lies in developing bounding technique based on strong stochastic ordering. We observe that more lumpable partitionings make the upper bound for the state of interest more accurate no matter whether matrix transformation is used or not. We also combine the approaches of state permutation, matrix transformation and state partitioning to improve the quality of the upper bound for the state of interest.

Pertaining to parts two and three, we also provide some elementary results on how to decompose lumpable MCs into a set of small-size matrices.

Along the way, our observations, definitions, properties and results are included as well.

1.3 Organization

This dissertation is organized as follows. Background on the theory of Markov chains is reviewed in Chapter 2. It covers discrete time Markov chains, continuous time Markov chains, stochastic complement and applications of Markov chains. Chapter 3 describes the different numerical methods for computing stationary distributions of MCs. We present the convergence analysis for the Markov iteration matrices in Chapter 4. This is the basis for designing a new rank reduction algorithm discussed in detail in Chapter 5. In Chapter 6, we focus on the relationship among concepts of lumpability, weak lumpability, quasi-lumpability and near complete decomposability. This chapter also includes our improvements on White's algorithm for identifying all possible lumpable partitions. Instead of computing exact stationary distributions of MCs, we provide bounding technique based on strong stochastic ordering in Chapter 7. Finally, Chapter 8 outlines our conclusions and gives directions for future work.

Chapter 2

Markov Chains

2.1 Introduction

The behavior of a physical system may frequently be represented by enumerating all states that the system may occupy and by indicating how it transitions from one state to another over time. Generally, such a system is represented by a *stochastic process.*

Definition 2.1.1 (Stochastic Process) *A stochastic process is a family of random variables $\{\mathcal{X}(t), t \in T\}$ defined on a given probability space and indexed by the parameter t. If the time index set T is discrete, then the process is a discrete-time process, otherwise the process is a continuous-time process. The values assumed by the random variable $\mathcal{X}(t)$ are called states, and the set of all possible states forms the state space of the process. If the state space is discrete, the process is called a chain.*

One of the simplest and most commonly employed stochastic processes is the *Markov process*: its future evolution depends only on its current state and not on its past history. This memoryless property is formally characterized by the *Markov property.*

Definition 2.1.2 (Markov Process) *A stochastic process is a Markov process which has the Markov property that, for all integers n and for any sequence $t_0,\ t_1,\ \cdots,\ t_n,\ t_{n+1}$ such* *that* $t_0 \leq t_1 \leq \cdots \leq t_{n-1} \leq t_n \leq t_{n+1}$, *we have*

$$P\{\mathcal{X}(t_{n+1}) \leq x \mid \mathcal{X}(t_n) = x_n,\ \mathcal{X}(t_{n-1}) = x_{n-1},\ \cdots,\ \mathcal{X}(t_0) = x_0\}$$

$$= P\{\mathcal{X}(t_{n+1}) \leq x \mid \mathcal{X}(t_n) = x_n\}.$$

An Markov process $\{\mathcal{X}(t), t \in T\}$ is *homogeneous* if its transitions are independent of time t, i.e.,

$$P\{\mathcal{X}(t) \leq x \mid \mathcal{X}(t_n) = x_n\} = P\{\mathcal{X}(t - t_n) \leq x \mid \mathcal{X}(0) = x_n\}.$$

Two kinds of Markov processes, discrete-time Markov chains (DTMCs) and continuous-time Markov chains (CTMCs), are briefly reviewed in the following sections. We only consider finite-state Markov chains in this dissertation.

2.2 Discrete Time Markov Chains

A discrete-time Markov chain (DTMC) is a Markov process with *discrete* state space and which is observed at a *discrete* set of time instants. Without loss of generality, let the time set be $T = \{0,\ 1\ ,\ \cdots,\ \}$ and let successive observations define the random variables $\mathcal{X}_0,\ \mathcal{X}_1,\ \mathcal{X}_2,\ \cdots$ at time instants $0,\ 1,\ 2,\ \cdots$.

Definition 2.2.1 (Discrete-Time Markov Chain) *The Markov process* $\{\mathcal{X}_n\}$, $n = 0,\ 1,\ 2,\ 3\ \cdots$ *is a discrete-time Markov chain if for all* n *and all states* x_n *we have:*

$$P\{\mathcal{X}_{n+1} = x_{n+1} \mid \mathcal{X}_n = x_n,\ \mathcal{X}_{n-1} = x_{n-1},\ \cdots,\ \mathcal{X}_0 = x_0\} = P\{\mathcal{X}_{n+1} = x_{n+1} \mid \mathcal{X}_n = x_n\}.$$

The conditional probabilities $P\{\mathcal{X}_{n+1} = x_{n+1} \mid \mathcal{X}_n = x_n\}$ are called the *single-step transition probabilities*. When it is homogeneous, these probabilities are independent of n and are simply written as

$$p_{ij} = P\{\mathcal{X}_{n+1} = j \mid \mathcal{X}_n = i\}$$

where $x_{n+1} = j$ and $x_n = i$. The matrix P, formed by placing p_{ij} in row i and column j, for all i and j, is called the *transition probability matrix.* Note that the entries of P have the property:

$$\sum_j p_{ij} = 1, \quad 0 \leq p_{ij} \leq 1, \quad i, j \in \{0, \ ,1 \ ,2, \ \cdots\}.$$

When the Markov chain is non-homogeneous, the elements p_{ij} must be replaced with $p_{ij}(n)$ and hence, the matrix P with $P(n)$. The generalization of a single-step transition probability matrix P to an m-step transition probability matrix $P^{(m)}$ with elements $p_{ij}^{(m)} = P\{\mathcal{X}(n+m) = j \mid \mathcal{X}_n = i\}$ may be obtained from the Chapman-Kolmogorov equations as follows.

Definition 2.2.2 (Chapman-Kolmogorov Equations) *The m-step transition probability matrix $P^{(m)}$ for a DTMC with transition probability matrix P satisfies:*

$$p_{ij}^{(m)} = \sum_{all \ \ k} p_{ik}^{(l)} p_{kj}^{(m-l)}, \quad for \quad 0 < l < m. \tag{2.1}$$

In matrix notation, the Chapman-Kolmogorov equations can be written as

$$P^{(m)} = P^{(l)} P^{(m-l)}.$$

By summing over all possible intermediate states, we have

$$P^{(m)} = PP^{(m-1)} = \cdots = P^m.$$

Definition 2.2.3 (Irreducibility) *A DTMC is irreducible if every state can be reached from every other state, i.e., there exists an integer m for which $p_{ij}^{(m)} > 0$ for every pair of states i and j.*

In studying DTMCs we are often interested in determining the probability that the chain is in a given state at a particular time step. Assume that $\pi_j(m)$ denotes the probability that an Markov chain is in state j at time step m, i.e.,

$$\pi_j(m) = P\{\mathcal{X}(m) = j\}. \tag{2.2}$$

Let the row vector $\pi(m) = (\pi_1(m),\ \pi_2(m),\ \cdots,\ \pi_j(m),\ \cdots)$. Then by use of Chapman-Kolmogorov equations we have

$$\pi(m) = \pi(0)P^m. \tag{2.3}$$

Definition 2.2.4 (Stationary Distribution) *A vector z is said to be the stationary distribution for an Markov chain with transition probability matrix P if and only if*

$$z = zP,\ \sum_j z_j = 1, \quad 0 \le z_j \le 1, \quad j \in \{0,\ ,1\ ,2,\ \cdots\}.$$

This means that if z is chosen as the initial state distribution $\pi_j(0) = z_j$ for all j, then for all m, we have $\pi_j(m) = z_j$.

2.3 Continuous Time Markov Chains

A DTMC has *discrete states* and *discrete times* at which transitions occur. If the transitions can occur at any time, the process is called a *continuous-time Markov chain* (CTMC). Recall that a DTMC uses the matrix P to hold the transition probabilities which define the DTMC. For continuous-time Markov chain however, a matrix Q, called the *infinitesimal generator matrix*, is used to hold the transition rates which define the CTMC. The element $q_{ij},\ i \neq j$ of Q is the rate at which the system moves from state i to state j. The diagonal elements are given by $q_{ii} = -\sum_{j\neq i} q_{ij}$. With this construction, the matrix Q satisfies:

$$\sum_j q_{ij} = 0, \quad i\ = 0,\ 1,\ \cdots.$$

We are concerned with the stationary behavior of the CTMC. Let $\pi(t)$ be the transient probability distribution at time t. Then from the Chapman-Kolmogorov equations, the vector $\pi(t)$ is written as

$$\pi(t) = \pi(0)e^{Qt}, \tag{2.4}$$

where $\pi(0)$ is the initial probability distribution. When the CTMC reaches statistical equilibrium, its probability distribution vector $\pi(t)$ does not change over time, i.e.,

$$\frac{d\pi(t)}{dt} = Q\pi(0)e^{Qt} = 0. \tag{2.5}$$

Thus the stationary distribution vector π can be found by solving the linear system of equations

$$\pi Q = 0, \quad \pi e = 1. \tag{2.6}$$

The uniqueness of the stationary distribution vector π satisfying (2.6) is a result of the irreducibility of the CTMC. Its infinitesimal generator Q has rank $n-1$, where n is the order of the matrix Q. Note that if only the stationary distribution is required, transformations are possible from P to Q or vice versa, by use of the following equations

$$Q = I - P \tag{2.7}$$

and

$$P = I + \Delta t Q \tag{2.8}$$

where

$$\Delta t \leq \frac{1}{max_i |q_{ii}|} \tag{2.9}$$

is a discretization factor. Equation (2.7) gives the procedure for producing a CTMC from a DTMC P. Equation (2.8) discretizes a CTMC so that the DTMC P obtained has transitions which occur at intervals Δt. The value of $\Delta t \leq \frac{1}{max_i |q_{ii}|}$ forces P to be stochastic. For the stationary analysis therefore, we can represent the same Markov chain by either P or Q and its stationary distribution vector is obtained from either $\pi = \pi P$ or $\pi Q = 0$ together with the normalization constraint $\pi e = 1$. Writing $A = I - P^T$ or $A = Q^T$ so that $A\pi^T = 0$

allows for the application of general numerical methods for the solution of linear systems of the form $Ax = b$. We are concerned solely with the stationary distributions of Markov chains in this dissertation.

2.4 Stochastic Complement

The term *stochastic complement* was coined by Meyer [43]. This concept provides insight into the dynamic behavior of nearly completely decomposable systems. Also, it gives us an approach to dividing large Markov chains into sub-chains of small sizes so that each small chain can be solved relatively easily. In this section, we follow Meyer's notations. Consider an Markov chain P with state space S which is partitioned into M subsets

$$S = \{S_1, S_2, \cdots, S_M\}.$$

Definition 2.4.1 (Stochastic Complement[43]) *Assume P is partitioned*

$$P = \begin{pmatrix} P_{11} & P_{12} & \cdots & P_{1M} \\ P_{21} & P_{22} & \cdots & P_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ P_{M1} & P_{M2} & \cdots & P_{MM} \end{pmatrix} \tag{2.10}$$

and let P_i denote the principal block sub-matrix of P obtained by deleting the i^{th} row and j^{th} column of blocks from P and let P_{*i} and P_{i*} designate

$$P_{i*} = (P_{i1}, P_{i2}, \cdots, P_{i,i-1}, P_{i,i+1}, \cdots, P_{iM})$$

and

$$P_{*i} = \begin{pmatrix} P_{1i} \\ \vdots \\ P_{i-1,i} \\ P_{i+1,i} \\ \vdots \\ P_{M,i} \end{pmatrix}.$$

That is, P_{i*} is the i^{th} row of blocks with P_{ii} removed, and P_{*i} is the i^{th} column of blocks with P_{ii} removed. Then the stochastic complement of P_{ii} is defined to be the matrix

$$S_{ii} = P_{ii} + P_{i*}(I - P_i)^{-1}P_{*i}. \tag{2.11}$$

Note that the matrix $I - P$ is a singular M-matrix of rank $n - 1$ (see definition of M-matrix in section 4.1.3). Every principal sub-matrix of $I - P$ other than itself is a non-singular M-matrix. So each matrix $(I - P_i)$ is a non-singular M-matrix and its inverse is a nonnegative matrix, i.e.,

$$(I - P_i)^{-1} \geq 0.$$

Therefore the stochastic complement is well defined. The matrix S_{ii} has interesting properties as follows.

Theorem 2.4.1 (Meyer 1989[43], Thm.2.1, Thm.2.3) *Let P be an irreducible stochastic matrix partitioned as (2.10). Then each stochastic complement*

$$S_{ii} = P_{ii} + P_{i*}(I - P_i)^{-1}P_{*i}$$

is also an irreducible stochastic matrix.

The Markov chain S_{ii} is called also *reduced chain* or *censored chain.* It is a Markov process observed only when the original chain is in some state in subset S_i and skipped whenever the process is in states outside of S_i.

Theorem 2.4.2 (Meyer 1989[43], Thm.2.2) *Let P be an irreducible stochastic matrix partitioned as (2.10), its stationary distribution vector π partitioned conformally as*

$$\pi = (\pi_1, \pi_2, \cdots, \pi_M)$$

and define

$$\phi_i = \frac{\pi_i}{\pi_i e}.$$

Then

$$\phi_i = \phi_i S_{ii}. \tag{2.12}$$

That is, ϕ_i is the stationary probability vector for the irreducible matrix S_{ii}. The vector ϕ_i is called *censored distribution* or the *conditional stationary distribution* of the i^{th} subset of states.

How can we obtain the stationary distribution vector π of the original Markov chain after we form the M censored chains of small size? This will be answered in the next theorem.

Theorem 2.4.3 (Meyer 1989[43], Thm.4.1) *Let P be an irreducible stochastic matrix partitioned as (2.10) and ϕ_i the stationary distribution for stochastic complement S_{ii}. Then the stationary distribution vector π for Markov chain P is written as*

$$\pi = (\xi_1\phi_1,\ \xi_2\phi_2,\ \cdots,\ \xi_M\phi_M)$$

where

$$\xi = (\xi_1,\ \xi_2,\ \cdots,\ \xi_M)$$

is the unique stationary distribution vector for the $M \times M$ irreducible stochastic matrix C whose entries are defined by

$$c_{ij} = \phi_i P_{ij} e. \tag{2.13}$$

The matrix C is referred to as the *coupling matrix* and the scalars $\xi_j,\ j = 1, 2, \cdots, M$ are called the *coupling factors*.

2.5 Applications of Markov chains

Markov chains have been widely and successfully employed in many areas. The following list is just a few of the interesting and successful applications of Markov chains.

- Computer system performance evaluation[70][23][8][38]
- Telecommunication, TCP/IP networks[40][8][6][38]
- Genetics and biology[40]
- Web Search Engine[59][2]
- Manufacturing, production scheduling[37]
- Chemical Engineering[34]
- Social activities such as sports events, public transportations[76]

The applications of Markov chains usually focus on evaluating performance metrics such as mean number of jobs/customers/packets in queue, the utilization of the servers/machines, the throughtput of the communication link and so on, just name a few.

Recall the assumption of the *Markov Property* for Markovian analysis. When the problem in the real world does not naturally fit into a Markov model, then we have to make unrealistic "exponentiality" assumptions but we can still get some insight and/or approximate solution. Let us look at a simple but important example: the use of discrete time Markov chain as web search engines.

Google's Page Ranking: An Example of a DTMC

Most of us use the Google search engine everyday. When looking up a term such as *Markov Chain*, there are thousands of web pages which include this phrase. A good search engine ranks these pages so that the page we need will most-likely fall in the top 10 and thus enables us to quickly find the information needed. Google's solution is to define a DTMC. It first determines all Web pages and links among these pages. Then it constructs a Markov chain transition matrix in which there is one state for each Web page. A transition occurs from state i to state j if and only if page i has a link to page j. If page i has $m > 0$ outgoing links, then the probability on each outgoing transition from state i is $\frac{1}{m}$. For example, assume there are only 10 pages altogether shown in Figure 2.1 which has stationary distribution

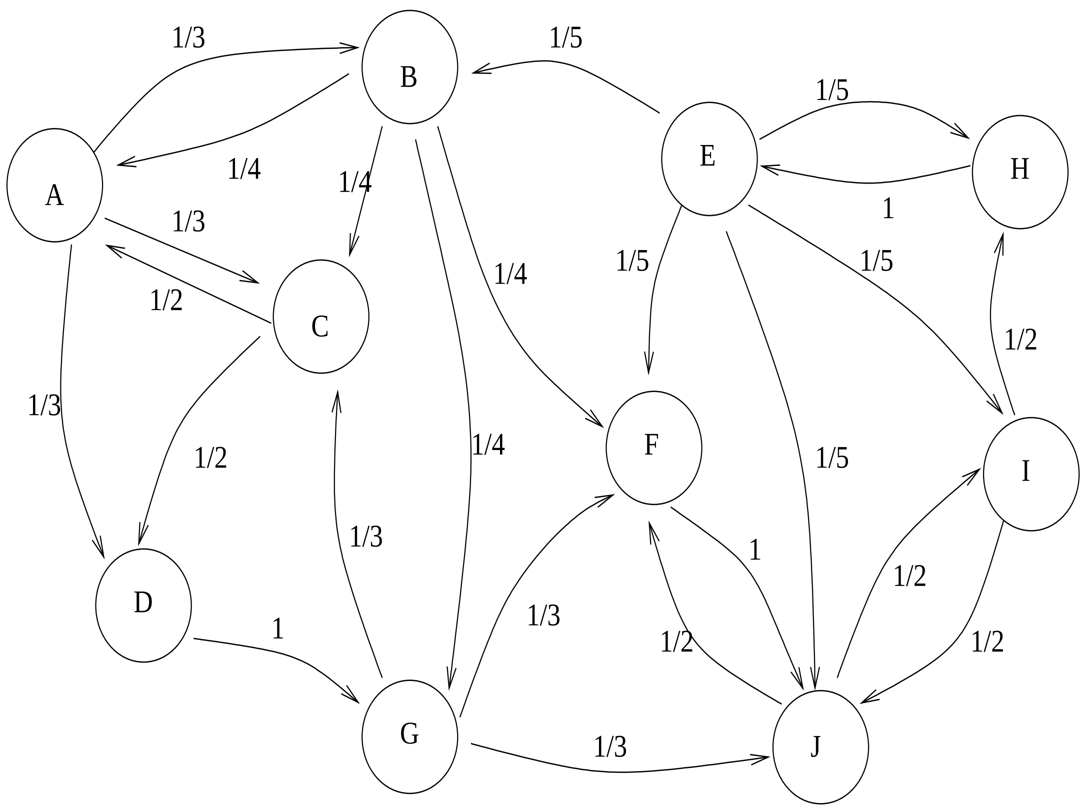


Figure 2.1: Web Page Links

$$\pi = (0.0170, 0.0277, 0.0202, 0.0158, 0.1103, 0.1909, 0.0227, 0.1103, 0.1764, 0.3088).$$

Page A has 3 outgoing links each with probability $\frac{1}{3}$ and has page ranking 0.0170 but page J has 2 outgoing links each with probability $\frac{1}{2}$ and with page ranking equal to 0.3088 which is the largest. Therefore the link for page J lists in the first position. There are some techniques concerning issues of dead ends (some page have no outgoing links and no self loop, thus the MC has no solution) and spider traps (some page has only self-loop outgoing links and thus the MC is not irreducible). For further information on this application, please refer to [2].

Chapter 3

Numerical Methods for Computing Stationary Distributions

In this section we review the most commonly employed numerical methods for solving linear systems of the form $Ax = b$.

3.1 Direct Method

The direct methods, including Gauss elimination and LU decomposition, compute solutions to the system $Ax = b$ in a fixed number of operations. This class of methods works well for Markov chain problems having small and medium-sized state spaces. Their chief disadvantage is the *fill-in* phenomenon which results, during the reduction phase, from the creation of nonzero elements in positions that previously contained zeros. This situation makes the organization of a compact storage scheme more difficult. Also, it may exhaust the available memory when the amount of fill-in is extensive. We review only the LU decomposition below, since it is this version that we shall use. The basic idea of LU decomposition is to rewrite a matrix A as the product of a nonsingular lower triangular matrix L and a nonsingular triangular matrix U, i.e.,

$$A = LU.$$

The solution of the system $Ax = b$ is then given by

$$x = U^{-1}L^{-1}b$$

and is obtained by using forward substitution to compute y as $y = L^{-1}b$ and then backward substitution to compute x as $x = U^{-1}y$. Finding an LU decomposition of A is equivalent to solving the linear equations

$$\sum_{k=1}^{i} l_{ik}u_{kj} = a_{ij} \quad for\ i \leq j,$$

and

$$\sum_{k=1}^{j} l_{ik}u_{kj} = a_{ij} \quad for\ i > j.$$

These specify n^2 equations in the $n^2 + n$ unknowns $l_{ik},\ i = 1, 2, \cdots, n;\ k = 1, 2, \cdots, i$ and $u_{kj},\ j = 1, 2, \cdots, n;\ k = 1, 2, \cdots, j$. Therefore we may set $l_{ii} = 1,\ 1 \leq i \leq n$ (called the *Doolittle* Decomposition) or $u_{ii} = 1,\ 1 \leq i \leq n$ (called the *Crout* Decomposition). When we apply this method to Markov chain problems, we have $A = Q^T$ or $A = I - P^T$ and $x = \pi^T$. Although an LU decomposition does not exist for all matrices A, it does exist for the transition matrices that correspond to *irreducible* Markov chains. The computational complexity of an LU decomposition is $O(2n^3/3)$.

3.2 Iterative Methods

Suppose the matrix A is split as $A = M - N$ in which M is nonsingular. Then the system of equations,

$$Ax = (M - N)x = b,$$

may be written in iterative form as

$$x^{(k+1)} = M^{-1}Nx^{(k)} + M^{-1}b, \quad k = 0, 1, \cdots .$$

The matrix $T = M^{-1}N$ is called the *iteration* matrix. When the system of equations is *non-homogeneous*, i.e., $b \neq 0$, then this iterative scheme will converge for any initial vector $x^{(0)}$ if and only if the spectral radius of T, $\rho(T)$, is strictly less than one. With continuous-time Markov chains, the system of equations is *homogeneous* (i.e., $b = 0$) and many of the commonly used iterative schemes have iteration matrices with $\rho(T) = 1$. We begin with a brief discussion of the *power* method.

The Power Method

The *Power method* is the simplest iterative method. Given an initial guess $\pi^{(0)}$, the stationary distribution vector π, which is the left-hand eigenvector corresponding to the unit eigenvalue of the stochastic matrix P in (1.1), is successively approximated by $\pi^{(0)},\ \pi^{(1)},\ \cdots,$ $\pi^{(k)}, \pi^{(k+1)},\ \cdots$, where

$$\pi^{(k+1)} = P^T \pi^{(k)} \quad \text{for } k = 0, 1, 2, \cdots .$$

The sequence of approximations $\pi^{(k)},\ k = 0, 1, 2, \cdots$, is expected to approach the exact solution π in the limit as $k \to \infty$. We shall let $T_p = P^T$ and refer to it as iteration matrix for the *power* method. For irreducible Markov chains, the unit eigenvalue is simple. If the Markov chain is also aperiodic, then no eigenvalue other than this unit eigenvalue exists on the unit circle. Let us assume for the moment that the eigenvalues of P, the transition probability matrix of an irreducible, aperiodic Markov chain, satisfy the relationship

$$1 = \lambda_1 > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|.$$

Let the corresponding eigenvectors be given by $x_1,\ x_2,\ \cdots,\ x_n$, i.e.,

$$P^T x_i = \lambda_i x_i, \quad i = 1,\ 2, \cdots,\ n.$$

When the initial approximation, $\pi^{(0)}$, can be written as a linear combination of the eigenvectors of P^T, i.e., when

$$\pi^{(0)} = \sum_{i=1}^{n} \eta_i x_i,$$

then it is easy to show that the rate of the convergence of the *power method* depends on the ratios $|\lambda_i|/|\lambda_1|,\ i = 2,\ 3, \ldots,\ n.$ This is due to the fact that

$$\pi^{(k+1)} = (P^T)^k \pi^{(0)} = \sum_{i=1}^{n} \eta_i \lambda_i^k x_i = \lambda_1^k \left(\eta_1 x_1 + \sum_{i=2}^{n} \eta_i (\frac{\lambda_i}{\lambda_1})^k x_i \right)$$

and the iterative process converges to the stationary distribution vector $\pi = x_1$, when the summation inside the parentheses goes to zero. The rate at which this goes to zero depends on the aforementioned ratios, and in particular, on the ratio $|\lambda_2|/|\lambda_1| = |\lambda_2|$. It is easily seen that the magnitude of the sub-dominant eigenvalue, λ_2, determines the convergence rate. The smaller the magnitude of the λ_2, the faster the convergence. In particular, the power method will converge very slowly when $|\lambda_2| \approx 1$.

The Method of Jacobi

In the method of Jacobi as well as in the next two iterative methods we consider, the matrix A is split as

$$A = M - N = D - (L + U)$$

where D, L and U are respectively the diagonal, negated strictly lower and negated strictly upper triangular parts of the matrix A. The *Jacobi* method is then essentially identical to the power method but applied to a different iteration matrix. The iteration matrix for this method is defined as

$$T_{jcb} = D^{-1}(L + U)$$

and the iterative scheme is therefore defined as

$$\pi^{(k+1)} = T_{jcb}\pi^{(k)}, \quad k = 1,\ 2,\ \cdots,\ n$$

or, in scalar form,

$$\pi_i^{(k+1)} = \frac{1}{d_{ii}} \sum_{j \neq i}^{n} (l_{ij} + u_{ij}) \pi_j^{(k)}, \quad k = 1,\ 2,\ \cdots,\ n.$$

The iteration matrix T_{jcb} has 1 as its dominant eigenvalue.

Gauss-Seidel

The point Gauss-Seidel method is identical to the *Jacobi* method except that the computation of the components of $\pi^{(k+1)}$ makes use of the most recently available component approximations. This is given in scalar form as

$$\pi_j^{(k+1)} = \frac{1}{d_{jj}} \left(\sum_{i=1}^{j-1} l_{ji} \pi_i^{(k+1)} + \sum_{i=j+1}^{n} u_{ji} \pi_i^{(k)} \right), \quad j = 1,\ 2,\ \cdots,\ n,$$

or in matrix form as

$$\pi^{(k+1)} = T_{gs}\pi^{(k)} \tag{3.1}$$

with $T_{gs} = M^{-1}N = (D - L)^{-1}U$. This is the *forward* Gauss-Seidel method. Similarly, a *backward* Gauss-Seidel method whose iteration matrix is $T_{gs} = (D - U)^{-1}L$ may also be defined.

Successive-Over-Relaxation (SOR)

The *SOR* method introduces a *relaxation* parameter, ω, into the iterative formula. This method is defined by the relation

$$\pi_j^{(k+1)} = (1 - \omega)\pi_j^{(k)} + \omega \{ \frac{1}{d_{jj}} (\sum_{i=1}^{j-1} l_{ji} \pi_i^{(k+1)} + \sum_{i=j+1}^{n} u_{ji} \pi_i^{(k)}) \}, j = 1,\ 2,\ \cdots,\ n,$$

or in matrix form as

$$\pi^{(k+1)} = T_{sor}\pi^{(k)} = (1-\omega)\pi^{(k)} + \omega\left\{D^{-1}(L\pi^{(k+1)} + U\pi^{(k)})\right\}.$$

For convergence, it is necessary, but not sufficient, that ω lie in between zero and 2. When $\omega < 1$, the method is sometimes referred to as *under relaxation*. The *SOR* method reduces to the method of Gauss-Seidel when $\omega = 1$.

Block Gauss-Seidel (BGS)

Block methods are often appropriate when the state space can be meaningfully partitioned into groups of states in such a way that states within a group communicate frequently with each other and only infrequently with the states of other groups. Markov chains having this property are said to be *nearly-completely-decomposable (NCD)*. Block methods may also be appropriate in other cases. Given a partition of the state space, this may be used to induce a partitioning of the transition matrix and the probability vector. We assume a total of N groups and write

$$\pi = (\pi_1,\ \pi_2,\ \ldots,\ \pi_N)$$

and

$$Q = \begin{pmatrix} Q_{11} & Q_{12} & \cdots & Q_{1N} \\ Q_{21} & Q_{22} & \cdots & Q_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ Q_{N1} & Q_{N2} & \cdots & Q_{NN} \end{pmatrix}.$$

We now introduce the block splitting:

$$Q^T = D_N - (L_N + U_N)$$

where D_N, L_N and U_N are respectively the block diagonal, negated strictly lower block and negated strictly upper block triangular matrices of Q^T. Their block structures are as follows:

$$D_N = \begin{pmatrix} D_{11} & 0 & \cdots & 0 \\ 0 & D_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & D_{NN} \end{pmatrix},$$

$$L_N = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ L_{21} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_{N1} & L_{N2} & \cdots & 0 \end{pmatrix}, \quad U_N = \begin{pmatrix} 0 & U_{12} & \cdots & U_{1N} \\ 0 & 0 & \cdots & U_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix},$$

where $D_{ii} = Q_{ii},\ i = 1, 2, \ldots N$, $L_{ij} = -Q_{ij}\ i > j$ and $U_{ij} = -Q_{ij},\ i < j$. Block iterative methods may now be constructed. For example, the forward block Gauss-Seidel method, given in matrix form, is

$$\pi^{(k+1)} = T_{bgs}\pi^{(k)} = \left\{(D_N - L_N)^{-1}U_N\right\}\pi^{(k)}.$$

A backward block version is obtained with $T_{bgs} = (D_N - U_N)^{-1}L_N$. Writing out the individual equations, instead of in block matrix format, we have

$$D_{ii}\pi_i^{(k+1)} = (\sum_{j=1}^{i-1} L_{ij}\pi_j^{(k+1)} + \sum_{j=i+1}^{N} U_{ij}\pi_j^{(k)}), \quad i = 1,\ 2,\ \cdots,\ N.$$

The BGS method necessitates the solution of N systems of linear equations at each iteration. It was shown by Courtois [19] that BGS always converges for irreducible and aperiodic Markov chains.

A final remark concerning all the iterative methods mentioned above is that they are all essentially applications of the *power* method but with different iteration matrices. It follows then that their convergence rates are also determined by the sub-dominant eigenvalues of their iteration matrices.

3.3 Projection Methods

Projection methods consist of creating small-dimension subspaces from which the best approximate solution is taken. These two main steps, i.e.,

- Construct subspaces
- Extract approximate solutions from subspaces

are repeated until convergence is reached.

Here we focus on Krylov projection methods for which the constructed subspaces are Krylov subspaces. Specifically, Krylov projection methods start with an initial approximate solution $x^{(0)}$ and find the best approximation

$$x^{(k)} = x^{(0)} + r$$

where r is selected from the Krylov subspace $\mathcal{K}_k$,

$$\mathcal{K}_k(A, v) = span\{v, Av, A^2v, \cdots, A^{k-1}v\},$$

for some vector v. The sequence of vectors $\{x^{(i)}\}, i = 0, 1, \cdots$, are expected to converge to the solution in a finite number of iterations.

In order to introduce the *conjugate gradient* method, we first review the gradient or steepest descent method. For a symmetric and positive definite (SPD) matrix A, the solution to the linear system $Ax = b$ is equivalent to minimizing the quadratic function

$$F(z) = \frac{1}{2}z^T Az - z^T b. \tag{3.2}$$

Its minimum can be found by use of a gradient/steepest descent method which, starting with an initial approximation $x^{(0)}$, consists of two steps in each iteration: (1) choose a direction d, and (2) choose how far to proceed in that direction, i.e., the step size α. For a SPD matrix A, the residual vector defines the search direction. The steepest descent method is shown in Figure 3.1.

```
Gradient/Steepest descent method:
1. Do until convergence
2.      Compute step size $\alpha$
3.      Update solution $x = x + \alpha r$
4.      Update direction $r = b - Ax$
```

Figure 3.1: Gradient/Steepest Descent Method

The value of α in each iteration is calculated as

$$\alpha_k = \frac{{r^{(k)}}^T r^{(k)}}{{r^{(k)}}^T A r^{(k)}}$$

which minimizes function $F(x + \alpha r)$ when evaluated at the point $z = x + \alpha r$.

In order to speed up convergence, widely employed scheme, the *conjugate gradient* method, chooses search directions $d^{(k)}$ under the condition that successive search direction vectors $d^{(k)}$ and $d^{(k-1)}$ are A-orthogonal or conjugate (i.e,. ${d^{(k)}}^T A d^{(k-1)} = 0$). This guarantees that successive residual vectors $r^{(k-1)}$, $r^{(k)}$ are mutually orthogonal (i.e., ${r^{(k)}}^T r^{(k)} = 0$). The corresponding step sizes α_k are determined by

$$\alpha_k = \frac{{r^{(k)}}^T r^{(k)}}{{d^{(k)}}^T A d^{(k)}}$$

and the scalars

$$\beta_k = \frac{{r^{(k)}}^T r^{(k)}}{{r^{(k-1)}}^T r^{(k-1)}}$$

are used to update the search direction vector. Take the initial search direction $d^{(0)} = b - Ax^{(0)}$, then the *conjugate gradient* method is illustrated in Figure 3.2.

The conjugate gradient method is indeed a Krylov subspace method because, at any iteration k, it can be shown that

$$r^{(k)} \in \mathcal{K}_k(A, r^{(0)}) = span\{r^{(0)}, Ar^{(0)} A^2 r^{(0)}, \cdots, A^k r^{(0)}\}$$

```
Conjugate Gradient Method:
1. Do until convergence
2.      Compute step size $\alpha$
3.      Update solution $x = x + \alpha d$
4.      Update residual $r = r - \alpha A d$
5.      Compute scalar $\beta$
6.      Update direction $d = d - \beta d$
```

Figure 3.2: Conjugate Gradient Method

and

$$d^{(k)} \in \mathcal{K}_k(A, d^{(0)}) = span\{d^{(0)}, Ad^{(0)} A^2 d^{(0)}, \cdots, A^k d^{(0)}\}.$$

Note that when the conjugate gradient method is applied to compute the stationary distribution of a Markov chain, the matrix A corresponds to the matrix $I - P^T$ (for DTMCs) or to Q^T (for CTMCs).

There are many other variants of Krylov subspace methods such as generalized minimum residual (GMRES), bi-conjugate gradient stabilized (BiCGSTAB) and so on. We recommend a study on Stewart's book[64] for further details on how these projection methods are applied to compute stationary distributions of Markov chains.

3.4 Decompositional Methods

In order to solve large scale Markov chains, it is appealing to apply divide and conquer strategy. The system is divided into subsystems, each of which is analyzed separately and then a global solution is constructed from the partial solutions. One such technique is called *Aggregation/Disaggregation Method* and is stated below. Assume P is partitioned as

$$P = \begin{pmatrix} P_{11} & P_{12} & \cdots & P_{1M} \\ P_{21} & P_{22} & \cdots & P_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ P_{M1} & P_{M2} & \cdots & P_{MM} \end{pmatrix} \tag{3.3}$$

and let P_i denote the principal block submatrix of P obtained by deleting the i^{th} row and j^{th} column of blocks from P and let P_{*i} and P_{i*} designate

$$P_{i*} = (P_{i1}, P_{i2}, \cdots, P_{i,i-1}, P_{i,i+1}, \cdots, P_{iM})$$

and

$$P_{*i} = \begin{pmatrix} P_{1i} \\ \vdots \\ P_{i-1,i} \\ P_{i+1,i} \\ \vdots \\ P_{M,i} \end{pmatrix},$$

then the exact Aggregation/Disaggregation method is illustrated in Figure 3.3.

Input:

$$P = \begin{pmatrix} P_{11} & P_{12} & \cdots & P_{1M} \\ P_{21} & P_{22} & \cdots & P_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ P_{M1} & P_{M2} & \cdots & P_{MM} \end{pmatrix}.$$

Output: The steady state distribution $\pi = (\pi_1, \cdots, \pi_M)$.

```
1.  For $I = 1, 2, \cdots, M$,
2.      Compute stochastic complement: $S_{II} = P_{II} + P_{I*}(I - P_I)^{-1} P_{*I}$
3.      Calcuate the censored distribution: $\phi_I = \phi_I S_{II},\ \phi_I e = 1,\ \phi_I > 0$.
4.  End for
5.  Form the coupling matrix: $C[I, J] = \phi_I P_{IJ} e,\ I, J = 1, 2, \cdots, M$.
6.  Solve the censored Markov chain: $\xi = \xi C,\ \xi e = 1,\ \xi > 0$.
7.  Compute the distribution for the chain $P$: $\pi = (\xi_1 \phi_1,\ \xi_2 \phi_2,\ \cdots, \xi_M \phi_M)$.
8.  Stop.
```

Figure 3.3: Exact Aggregation/Disaggregation Method

The idea of this method is first of all, to uncouple (or divide) the original chain into several smaller independent chains and to calculate the stationary distribution of each chain. Then the coupling matrix is formed and its stationary distribution is determined. Finally the overall solution to the original chain is formed. One of disadvantages for this method is the computation of M matrix inverses. This method is called exact aggregation/disaggregation because the exact solution to the original chain can be obtained after one aggregation and one disaggregation. Later on we will see iterative aggregation/disaggregation method which usually take many iterations of aggregations and disaggregations till convergence.

Input:

$$P = \begin{pmatrix} P_{11} & P_{12} & \cdots & P_{1M} \\ P_{21} & P_{22} & \cdots & P_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ P_{M1} & P_{M2} & \cdots & P_{MM} \end{pmatrix}.$$

Output: The steady state distribution $\pi = (\pi_1, \cdots, \pi_M)$.

```
1.  Initialization: $\pi^{(0)} = (\pi_1^{(0)}, \cdots, \pi_M^{(0)})$, set $m = 1$.
2.  **For** each $I = 1, 2, \cdots, M$
3.      Compute: $\phi_I^{(m-1)} = \pi_I^{(m-1)} / ||\pi_I^{(m-1)}||_1$
4.  **End for**
5.  Form the coupling matrix: $C[I, J]^{(m-1)} = \phi_I^{(m-1)} P_{IJ} e,\ I, J = 1, 2, \cdots, M$.
6.  Solve the censored Markov chain: $\xi^{(m-1)} = \xi^{(m-1)} C^{(m-1)},\ \xi^{(m-1)} e = 1,\ \xi^{(m-1)} > 0$.
7.  Form the row vector: $z^{(m)} = (\xi_1^{(m-1)} \phi_1^{(m-1)},\ \xi_2^{(m-1)} \phi_2^{(m-1)},\ \cdots, \xi_M^{(m-1)} \phi_M^{(m-1)})$.
8.  **For** each $K = 1, 2, \cdots, M$
9.      Compute $\pi_K^{(m)}$: $\pi_K^{(m)} = \pi_K^{(m)} P_{KK} + \sum_{J<K} \pi_J^{(m)} P_{JK} + \sum_{J>K} z_J^{(m)} P_{JK}$
10. **End for**
11. Test for convergence:
12.     **If** not convergence, set $m = m + 1$, go to step 2.
12.     **Else** Stop.
```

Figure 3.4: Iterative Aggregation/Disaggregation Method

For NCD Markov chains, the KMS (Koury, McAllister and Stewart) method [63][65] has been successfully used and is shown in Figure 3.4. The KMS method first forms the aggregation matrix with the current approximate solution and calculate its stationary distribution. Then M subsystems of equations are solved by use of a block Gauss-Seidel step. This defines disaggregation step. If the overall solution obtained is not sufficiently accurate, then the whole process is repeated with the solution currently obtained being the next initial approximate solution. The experimental results in [65] show rapid convergence for NCD Markov chains.

Based on the ways in which the aggregation is performed and the ways in which the disaggregation is executed, many variants for Iterative Aggregation/Disaggregation (IAD) have been successfully designed and implemented, See [14][51][17] [72][12][32] [13][24]. Here we would like to briefly introduce Feinberg & Chiu's method since it carefully exploits the structure of the transition diagram.

Feinberg & Chiu's Method

This method takes advantages of the *single-input* state structure for each group of states. That is, all forward and backward transitions between any two groups of states go through a particular state, namely the *single-input state*. One simple example of the transition diagram is illustrated in Figure 3.5 where the set of states containing state A have all incoming transitions from other sets of states through state A and the set of states containing state B have all incoming transitions from other sets of states through state B. There are no restrictions on all internal transitions for each subset of states.

If we number all groups of states from 1 to M, then the transition matrix P or Q has all its off-diagonal blocks with only 1 nonzero column shown below. Note that this matrix does not correspond to the Markov chain shown in Figure 3.5. Otherwise the transition diagram 3.5 will be much more complicated.

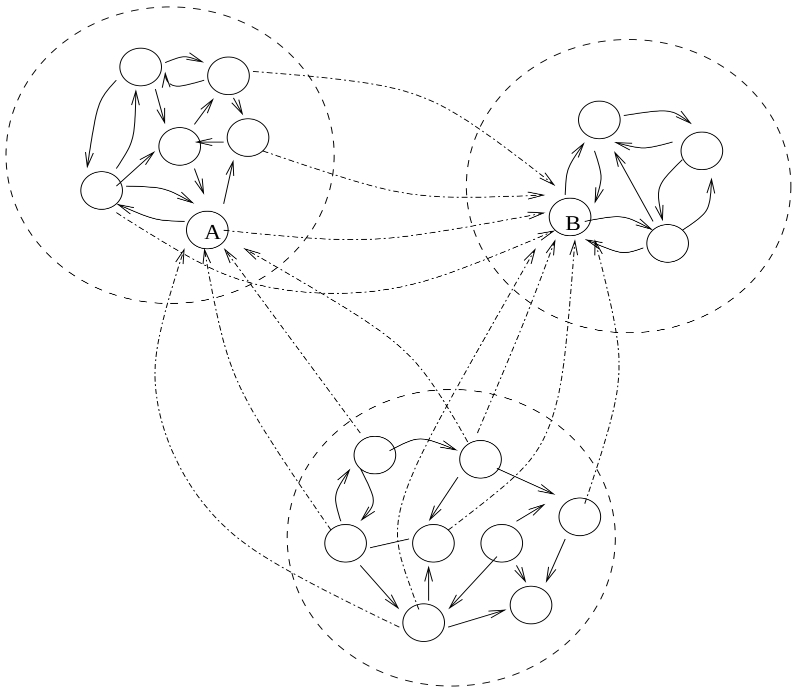


Figure 3.5: Single-input State Structure

$$P = \left(\begin{array}{cccc|cccc|cccc}
X & X & X & X & & X & & & X & & & \\
X & X & X & X & & X & & & X & & & \\
X & X & X & X & & X & & & X & & & \\
X & X & X & X & & X & & & X & & & \\
\hline
X & & & & X & X & X & X & X & & & \\
X & & & & X & X & X & X & X & & & \\
X & & & & X & X & X & X & X & & & \\
X & & & & X & X & X & X & X & & & \\
\hline
X & & & & & X & & & X & X & X & X \\
X & & & & & X & & & X & X & X & X \\
X & & & & & X & & & X & X & X & X \\
X & & & & & X & & & X & X & X & X
\end{array}\right)$$

Assume the Markov chain P is partitioned into the form of equation (3.3) following the *single-input state* structure. Assume also that the *single-input state* in each subset S_i

is the last state s_i, $|S_i| = s_i$, based on some state ordering. Then pseudo code for Feinberg & Chiu's method is illustrated in Figure 3.6.

Input:

$$P = \begin{pmatrix} P_{11} & P_{12} & \cdots & P_{1M} \\ P_{21} & P_{22} & \cdots & P_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ P_{M1} & P_{M2} & \cdots & P_{MM} \end{pmatrix},$$

Output: The steady state distribution $\pi = (\pi_1, \cdots, \pi_M)$.

```
1.  For $I = 1, 2, \cdots, M$,
2.      Compute aggregated matrix: $\tilde{P_{II}} = P_{II} + \sum_{J=1,J\neq I}^{M} P_{IJ}$
3.      Calcuate the distribution: $\phi_I = \phi_I \tilde{P_{II}},\ \phi_I e = 1,\ \phi_I > 0$.
4.  End for
5.  Form the intertransition matrix:
6.      $K[I,J] = \phi_I P_{IJ} e,\ I \neq J,\ I,J = 1,2,\cdots,M$.
6.      $K[I,I] = 1 - \sum_{j=1,j\neq I}^{M} K[I,j]$.
7.  Solve the Markov chain: $\xi = \xi K,\ \xi e = 1,\ \xi > 0$.
8.  Compute the overall distribution for $P$: $\pi = (\xi_1\phi_1,\ \xi_2\phi_2,\ \cdots, \xi_M\phi_M)$.
9.  Stop.
```

Figure 3.6: Feinberg & Chiu's Method

Note that when the subsets do not have the same number of states, the summation in step 2 of Figure 3.6 must be computed carefully. Since we assume that the *single-input state* is numbered last in each subset, we can add all off-diagonal nonzero columns into the last column of each diagonal block to form the aggregation matrix $\tilde{P_{II}},\ I = 1, 2, \cdots, M$. The basic idea and the steps are the same as those of exact A/D method. It was proved that the final solution is exact. Based on the *single-input state* structure however, this algorithm does not involve the computation of matrix inverses when forming each aggregated matrix $\tilde{P_{II}},\ I = 1, 2, \cdots, M$. It terminates in a finite number of steps and when the size of each subset is equal to $\sqrt{n}$, its time complexity is $O(n^2)$.

Chapter 4

Convergence Behavior of Markov Iteration Matrices

Our central theme in this chapter is an investigation into the condition under which iterative methods would converge in one iteration. Our objective is to obtain both sufficient and necessary conditions. Some backgrounds on nonnegative matrices, M-Matrices and matrix splittings are reviewed first.

4.1 Nonnegative Matrices

In what follows, we shall use the following notations for any matrix $A \in \mathbf{R^{n\times n}}$ of size $n \times n$.

- $A \geq 0$ if $a_{ij} \geq 0$ for all i and j .
- $A > 0$ if $a_{ij} \geq 0$ for all i, j and $A \neq 0$.
- $A \gg 0$ if $a_{ij} > 0$ for all i and j .

A matrix $A \in \mathbf{R^{n\times n}}$ is said to be *nonnegative* if $A \geq 0$. When $A \gg 0$, then A is said to be *positive*. Some of major results concerning nonnegative matrices are contained in the theorem of Perron-Frobenius (See Berman and Plemmons [7]). Let us introduce the concept of the *spectral radius* of a matrix A as the number $\rho(A) = max_i|\lambda_i(A)|$ where $\lambda_i(A)$ denotes the i^{th} eigenvalue of A. The *spectrum* $\sigma(A)$ of A is the set of its distinct eigenvalues. For any natural matrix norm $||A||$, we have $\rho(A) \leq ||A||$. Some parts of the results presented in Perron-Frobenius theorem are illustrated below.

Theorem 4.1.1 (Perron-Frobenius) *Let $A \geq 0$ be an irreducible square matrix of order n. Then,*

- *A has a positive real eigenvalue, λ_1, equal to its spectral radius.*
- *There is a positive eigenvector $x > 0$ associated with $\rho(A)$, i.e.,*

$$Ax = \lambda_1 x \quad and \quad x > 0.$$

- *$\rho(A)$ increases when any entry of A increases.*
- *$\rho(A)$ is a simple eigenvalue of A, i.e., λ_1 is a simple root of*

$$|\lambda I - A| = 0.$$

4.1.1 Stochastic Matrices

The matrices P and Q for DTMC and CTMC have many helpful and interesting properties to which we now turn. A matrix $P \in \mathbf{R}^{\mathbf{n}\times\mathbf{n}}$ is said to be a *stochastic* matrix if it satisfies the following conditions

$$\sum_{all\ j} p_{ij} = 1\ \forall i; \qquad P_{ij} \geq 0\ \ \forall i, j.$$

Some properties of stochastic matrices are as follows.

Proposition 4.1.1 *Every stochastic matrix P has an eigenvalue equal to unity.*

Proof A stochastic matrix P has row sums equal to 1. So we have

$$Pe = e$$

where $e = (1,\ 1,\ \cdots,\ 1)^T$. It immediately follows that P has a unit eigenvalue. $\square$

Proposition 4.1.2 *The eigenvalues of a stochastic matrix P must have modulus less than or equal to* 1*.*

Proof A stochastic matrix P has row sums equal to 1. So we have

$$||P||_\infty = 1.$$

It is known also that

$$\rho(P) \leq ||P||_\infty$$

and immediately it follows that

$$\rho(P) \leq 1.$$

$\square$

Proposition 4.1.3 *The stochastic matrix P of an irreducible Markov chain possesses a simple unit eigenvalue.*

Proof This follows in a straightforward manner from Propositions 4.1.1, 4.1.2 and the Perron-Frobenius theorem. $\square$

Proposition 4.1.4 *The right-hand eigenvector corresponding to a unit eigenvalue $\lambda_1 = 1$ of a stochastic matrix P is given by $e = (1,\ 1,\ \cdots,\ 1)^T$.*

Proof A stochastic matrix P has row sums equal to 1. So $Pe = e = \lambda_1 e$. $\square$

Proposition 4.1.5 *The vector π is the stationary probability vector of a stochastic matrix P if and only if it is a left-hand eigenvector corresponding to a unit eigenvalue.*

Proof From the definition of stationary distribution, we have

$$\pi = \pi P$$

and thus $\pi P = \lambda_1 \pi$ with $\lambda_1 = 1$. The converse is obvious as well. □

4.1.2 Matrix Splittings

Definition 4.1.1 (Regular and Weak Regular Splitting) *A splitting $A = M - N$ is called a regular splitting if $M^{-1} \geq 0$ and $N \geq 0$. It is called a weak regular splitting if $M^{-1} \geq 0$ and $M^{-1}N \geq 0$.*

It is obvious that a regular splitting is also a weak regular splitting, but the converse is not true.

Definition 4.1.2 (Convergent and Semiconvergent Matrices) *A matrix T is said to be convergent whenever $\lim_{k\to\infty} T^k = 0$. It is said to be semiconvergent whenever $\lim_{k\to\infty} T^k$ exists. This limit need not be zero.*

Definition 4.1.3 (Convergent and Semiconvergent Splittings) *A splitting $A = M - N$ is said to be convergent if $M^{-1}N$ is convergent. It is said to be semiconvergent if $M^{-1}N$ is semiconvergent.*

4.1.3 M-Matrices

Now we review the definition and properties of an M-matrix. More detailed information and proofs may be found in *Nonnegative Matrices in the Mathematical Sciences* by Berman and Plemmons[7]. See also pages 169–173 [64].

Definition 4.1.4 (M-Matrix) *If a finite matrix A with non-positive off-diagonal elements and nonnegative diagonal elements can be expressed in the form*

$$A = sI - G, \quad s > 0, \quad G \leq 0 \text{ and } s \geq \rho(G),$$

then A is called an M-Matrix.

A Z-matrix is a matrix with all off-diagonal elements nonpositive. A Z-matrix can be written as $A = sI - G$ with $G \geq 0$. For instance, we can take $s = \max_{i=1}^{n} a_{ii}$. If $s \geq \rho(G)$, then matrix M is also an M-matrix. When $s > \rho(G)$, the matrix M is non-singular and when $s = \rho(G)$, it is singular.

Theorem 4.1.2 *Let A be a Z-matrix. Then the following are equivalent.*

1. *A is a nonsingular M-matrix.*
2. *$A^{-1} \geq 0$.*
3. *Every real eigenvalue of A is positive.*
4. *All of the principal matrices of A are positive.*
5. *$A + \alpha I$ is nonsingular for all $\alpha \geq 0$.*
6. *$Av \geq 0$ for some nonnegative vector $v \geq 0$.*

It is straightforward that the negated infinitesimal generator $-Q$ of a CTMC is an M-Matrix. An important property of M-Matrix is stated as follows (See[64]).

Theorem 4.1.3 *Any M-matrix A has the property that*

1. *$A^{-1} \geq 0$.*
2. *If $\tilde{A}$ is obtained from A by setting any off-diagonal element to zero, then $\tilde{A}$ is also an M-Matrix.*

Lemma 4.1.1 *A matrix T is semiconvergent if and only if*

1. *$\rho(T) \leq 1$;*
2. *if $\rho(T) = 1$, then all the Jordan blocks associated with 1 for T are one-by-one;*
3. *if $\rho(T) = 1$, then $\lambda \in \sigma(T)$ and $|\lambda| = 1$ implies $\lambda = 1$.*

Proof The proof may be found in [7]. □

Theorem 4.1.4 *For an irreducible singular M-matrix A, we have*

1. *index*$(A) = 1$.

2. *A has rank* $n - 1$.

3. *There is a positive vector* $x > 0$ *such that* $Ax \geq 0$.

4. *A is* almost monotone*, i.e., if* $Ax \geq 0$*, then* $Ax = 0$.

5. *Each principal sub-matrix of A other than A itself is a nonsingular M-matrix.*

4.2 Motivating Examples

Based on the definitions and theorems reviewed above, we are now ready to investigate conditions for convergence in a single iteration. Let us look at an example first.

Example 1:

Consider the 5-state CTMC shown graphically in Figure 4.1.

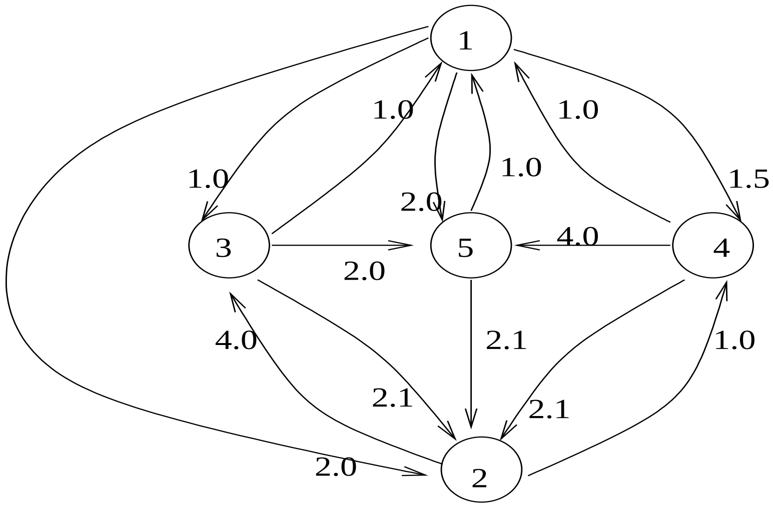


Figure 4.1: A 5-state Continuous Time Markov Chain

Its transition rate matrix Q is

$$Q = \begin{pmatrix} -6.50 & 2.00 & 1.00 & 1.50 & 2.00 \\ 0.00 & -5.00 & 4.00 & 1.00 & 0.00 \\ 1.00 & 2.10 & -5.10 & 0.00 & 2.00 \\ 1.00 & 2.10 & 0.00 & -7.10 & 4.00 \\ 1.00 & 2.10 & 0.00 & 0.00 & -3.10 \end{pmatrix}$$

and its rank is $rank(Q) = 4$. It is also easy to verify that Q is irreducible and aperiodic. Splitting Q^T as $Q^T = D - L - U$ where D, L and U are respectively the diagonal, negated lower triangular and upper triangular part of the matrix Q^T, gives

$$D = \begin{pmatrix} -6.50 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & -5.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & -5.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & -7.10 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & -3.10 \end{pmatrix},$$

$$L = \begin{pmatrix} 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ -2.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ -1.00 & -4.00 & 0.00 & 0.00 & 0.00 \\ -1.50 & -1.00 & 0.00 & 0.00 & 0.00 \\ -2.00 & 0.00 & -2.00 & -4.00 & 0.00 \end{pmatrix} \; and \, U = \begin{pmatrix} 0.00 & 0.00 & -1.00 & -1.00 & -1.00 \\ 0.00 & 0.00 & -2.10 & -2.10 & -2.10 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \end{pmatrix}.$$

It follows that the iteration matrix T_{gs} for the forward Gauss-Seidel method is

$$T_{gs} = (D-L)^{-1}U = \begin{pmatrix} 0.00 & 0.00 & 0.1538 & 0.1538 & 0.1538 \\ 0.00 & 0.00 & 0.4815 & 0.4815 & 0.4815 \\ 0.00 & 0.00 & 0.4078 & 0.4078 & 0.4078 \\ 0.00 & 0.00 & 0.1003 & 0.1003 & 0.1003 \\ 0.00 & 0.00 & 0.4918 & 0.4918 & 0.4918 \end{pmatrix}.$$

The spectrum of this iteration matrix is $\sigma(T_{gs}) = \{1.0,\ 0.0\}$ with 1.0 being simple. The eigenvalues (the diagonal elements of the matrix Λ below) and eigenvectors (the column vectors in matrix V below) of T_{gs} are respectively

$$\Lambda = \begin{pmatrix} 0.0 & & & & \\ & 0.0 & & & \\ & & 0.0 & & \\ & & & 1.0 & \\ & & & & 0.0 \end{pmatrix} \quad \textit{and } V = \begin{pmatrix} 1.00 & 0.00 & -0.2425 & 0.1874 & -0.2051 \\ 0.00 & 1.00 & -0.9701 & 0.5866 & -0.8816 \\ 0.00 & 0.00 & 0.0000 & 0.4968 & -0.2904 \\ 0.00 & 0.00 & 0.0000 & 0.1222 & -0.0194 \\ 0.00 & 0.00 & 0.0000 & 0.5992 & 0.3099 \end{pmatrix}.$$

The reader may wish to verify that, given the three initial starting vectors

$$x^{(0)} = \begin{pmatrix} 0.20 \\ 0.20 \\ 0.20 \\ 0.20 \\ 0.20 \end{pmatrix},\ y^{(0)} = \begin{pmatrix} 0.05 \\ 0.05 \\ 0.20 \\ 0.30 \\ 0.40 \end{pmatrix} \ \textit{and } z^{(0)} = \begin{pmatrix} 0.40 \\ 0.60 \\ 0.00 \\ 0.00 \\ 0.00 \end{pmatrix},$$

the iterative Gauss-Seidel scheme converges to the exact solution

$$\pi = (0.0941,\ 0.2944,\ 0.2494,\ 0.0613,\ 0.3007)$$

in a single iteration with starting vector either $x^{(0)}$ or $y^{(0)}$, but not $z^{(0)}$. In fact it never converges with the initial vector $z^{(0)}$, since after the first iteration, the new approximation becomes the zero vector (*all* components are zero) and remains like this in all successive iterations. We notice as well that it takes 11 iterations for the backward Gauss-Seidel to obtain the same solution π above when the initial vector is $x^{(0)}$ or $y^{(0)}$ and 10 iterations when the initial vector is $z^{(0)}$.

Example 2: An $M/G/1$-Like Queue

The underlying Markov chain for a truncated $M/G/1$-like queueing model has its infinitesimal generator in upper Hessenberg form. In other words, it has the following nonzero structure (where "X" indicates a nonzero element)

$$Q = \left(\begin{array}{ccccccc|ccccccc}
X & X & X & \cdots & X & X & X & X & X & X & \cdots & X & X & X \\
X & X & X & \cdots & X & X & X & X & X & X & \cdots & X & X & X \\
0 & X & X & \cdots & X & X & X & X & X & X & \cdots & X & X & X \\
0 & 0 & X & \cdots & X & X & X & X & X & X & \cdots & X & X & X \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & 0 & X & X & X & X & X & \cdots & X & X & X \\
\hline
0 & 0 & 0 & \cdots & 0 & 0 & X & X & X & X & \cdots & X & X & X \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & X & X & X & \cdots & X & X & X \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & X & X & \cdots & X & X & X \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & X & \cdots & X & X & X \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & X & X
\end{array}\right).$$

The block Gauss-Seidel method converges in a single iteration for Markov chains like this, whose transition matrix is upper Hessenberg and which is partitioned into four blocks as indicated above. For a particular example, given an Markov chain

$$Q^T = D_N - (L_N + U_N) = \left(\begin{array}{c|c} D_{11} & -U_{12} \\ \hline -L_{21} & D_{22} \end{array}\right) =$$

$$\left(\begin{array}{rrrr|rrrr}
-11.10 & 4.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
1.00 & -14.90 & 5.50 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
1.20 & 1.60 & -18.50 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 \\
1.00 & 1.50 & 2.00 & -8.70 & 3.00 & 0.00 & 0.00 & 0.00 \\
\hline
1.00 & 1.20 & 2.10 & 1.10 & -8.80 & 2.20 & 0.00 & 0.00 \\
2.40 & 2.10 & 2.40 & 1.60 & 1.20 & -5.70 & 3.00 & 0.00 \\
2.10 & 1.40 & 2.50 & 3.10 & 2.80 & 2.20 & -5.60 & 3.10 \\
2.40 & 3.10 & 4.00 & 2.70 & 1.80 & 1.30 & 2.60 & -3.10
\end{array}\right),$$

the iteration matrix for the forward block Gauss-Seidel is

$$T_{bgs} = (D_N - L_N)^{-1} U_N = \begin{pmatrix}
0.00 & 0.00 & 0.00 & 0.00 & 0.00053206597368 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00147648307697 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00390318761329 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.34604059377192 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 1.00000000000000 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 3.82220673054407 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 6.67305606366524 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 8.08857771766120 & 0.00 & 0.00 & 0.00
\end{pmatrix}.$$

It converges in a single iteration with the initial approximation $x^{(0)}$ below.

$$x^{(0)} = \begin{pmatrix} 0.12500000 \\ 0.12500000 \\ 0.12500000 \\ 0.12500000 \\ 0.12500000 \\ 0.12500000 \\ 0.12500000 \\ 0.12500000 \end{pmatrix} ; \quad \pi = \begin{pmatrix} 0.00002668897986 \\ 0.00007406191911 \\ 0.00019578792999 \\ 0.01735775429195 \\ 0.05016103487381 \\ 0.19172584510574 \\ 0.33472739792441 \\ 0.40573142897513 \end{pmatrix}$$

The exact solution π is shown above. These examples lead us to investigate conditions under which the Gauss-Seidel method for computing stationary distributions of finite irreducible Markov chains converges in a single iteration. We would also like to investigate the role played by the initial vector approximation in such convergence. And finally, we would like to know if iterative methods other than Gauss-Seidel also have this property.

4.3 Spectrum Condition

Our convergence analysis of the power method might suggest that iteration matrices having a single unit eigenvalue and all other eigenvalues equal to zero might be an appropriate place to begin our investigations.

Spectrum of Iteration Matrix: $\lambda_1 = 1, \lambda_2 = 0, \ldots, \lambda_n = 0$.

Given an iteration matrix T, with eigensolution $Ty_i = \lambda_i y_i,\ y_i \neq 0,\ i = 1,\ 2\ ,\ \ldots,\ n$ and an initial approximation $x^{(0)}$ which we assume can be written as a linear combination of the eigenvectors, we have

$$\begin{aligned}
x^{(k)} &= Tx^{(k-1)} = T^2x^{(k-2)} \;= \cdots = T^k x^{(0)} \\
&= \sum_{i=1}^{n} \eta_i \lambda_i^k y_i \\
&= \lambda_1^k \left(\eta_1 y_1 + \sum_{i=2}^{n} \eta_i (\frac{\lambda_i}{\lambda_1})^k y_i \right) \\
&= \eta_1 y_1.
\end{aligned}$$

Then the computed solution satisfies

$$x^* = \frac{x^{(1)}}{||x^{(1)}||_1} = \frac{y_1}{||y_1||_1}$$

which is exactly what we are looking for. After one iteration we have computed the solution, y_1, exactly. However it turns out that this condition is not sufficient for Gauss-Seidel method to converge in a single iteration for any positive initial $x^{(0)} > 0$.

Example 3: A Counter Example

Given a CTMC with infinitesimal generator [35]

$$Q = \begin{pmatrix} -1.00 & 0.50 & 0.50 & 0.00 \\ 0.50 & -1.00 & 0.00 & 0.50 \\ 0.50 & 0.00 & -1.00 & 0.50 \\ 0.00 & 0.50 & 0.50 & -1.00 \end{pmatrix}$$

and Gauss-Seidel iteration matrix

$$T_{gs} = \begin{pmatrix} 0.00 & 0.50 & 0.50 & 0.00 \\ 0.00 & 0.25 & 0.25 & 0.50 \\ 0.00 & 0.25 & 0.25 & 0.50 \\ 0.00 & 0.25 & 0.25 & 0.50 \end{pmatrix},$$

the spectrum of T is given by $\sigma(T_{gs}) = \{1.00,\ 0.00\}$ and 1.00 is simple. The corresponding eigenvectors are

$$\begin{pmatrix} 0.50 \\ 0.50 \\ 0.50 \\ 0.50 \end{pmatrix}, \begin{pmatrix} 1.00 \\ 0.00 \\ 0.00 \\ 0.00 \end{pmatrix}, \begin{pmatrix} -1.00 \\ 0.00 \\ 0.00 \\ 0.00 \end{pmatrix} \; and \; \begin{pmatrix} -1.00 \\ 0.00 \\ 0.00 \\ 0.00 \end{pmatrix}.$$

With the initial approximation $x^{(0)} = (0.10\ 0.20\ 0.30\ 0.40)^T$, the Gauss-Seidel method does *not* converge in one iteration. In the previous analysis of the convergence of the power method, it was assumed that the initial approximation $x^{(0)}$ could be written as a linear combination of the eigenvectors of the iteration matrix. In this example, the initial approximation $x^{(0)} = (0.10\ 0.200\ 0.30\ 0.40)^T$ *can not* be expressed as a linear combination of the eigenvectors. Observe that only the first eigenvector, y_1, has nonzero components in positions 2, 3 and 4 and they are identical ($= 0.50$). In these positions in the other eigenvectors, the components are all zero. However, the initial vector, $x^{(0)}$, has unequal components in positions 2, 3 and 4 and hence cannot be written as a linear combination of the eigenvectors $y_i,\ i = 1, 2, 3, 4$. It is for this reason that the initial vector does not converge in one iteration (In fact, 2 iterations are required). With a different initial approximation, for instance, $x^{(0)} = (0.10\ 0.30\ 0.30\ 0.30)^T$ with identical components in the last three positions, the method does indeed converge in one iteration. In this case, the vector $x^{(0)}$ can be written as the liner combination

$$x^{(0)} = 0.60\ y_1 + 0.20\ y_4$$

where $y_i,\ i = 1,\ 2,\ 3,\ 4$ are the eigenvectors of T. Therefore, we have to investigate other conditions.

4.4 Rank-1 Condition

Before showing that an iteration matrix T having rank one is both a necessary and a sufficient condition for convergence in a single iteration, we prove a number of lemmas that we will need.

Lemma 4.4.1 *Any rank-1 matrix A can be decomposed into the product of two nonzero column vectors, $A = uv^T$.*

Proof The *Rank Normal Form*, see pages 136–137[44] for example, says that if A is an $n \times n$ matrix such that $rank(A) = 1$, then there exist nonsingular matrices P and Q such that

$$PAQ = \begin{pmatrix} I_{r,r} & 0_{r,n-r} \\ 0_{n-r,r} & 0_{n-r,n-r} \end{pmatrix}.$$

When matrix A is rank 1, we have

$$PAQ = \begin{pmatrix} 1_{1,1} & 0_{1,n-1} \\ 0_{n-1,1} & 0_{n-1,n-1} \end{pmatrix} = e_1 e_1^T$$

where e_1 is a column vector whose first component is one and whose remaining components are zero. Let $(P^{-1})_{*1}$ be the first column of P^{-1} and $(Q^{-1})_{1*}$ the first row of Q^{-1}. Then we have

$$A = P^{-1}e_1e_1^TQ^{-1} = (P^{-1}e_1)(e_1^TQ^{-1}) = (P^{-1})_{*1}(Q^{-1})_{1*} = uv^T.$$

The lemma follows by setting $u = (P^{-1})_{*1}$ and $v^T = (Q^{-1})_{1*}$. □

Lemma 4.4.2 *Any rank-1 matrix A has at most one nonzero eigenvalue λ_1 which is simple if it is nonzero.*

Proof To prove this lemma, we use the following property of the coefficients of characteristic equations (Page 494[44]). Let

$$|A - \lambda I| = p(\lambda) = \lambda^n + c_1\lambda^{n-1} + c_2\lambda^{n-2} + \cdots + c_{n-1}\lambda + c_n = 0$$

be the characteristic equation associated with the matrix A. Then

$$c_k \ = \ (-1)^k \sum (all\ k \times k\ principal\ minors).$$

Since all principal minors corresponding to blocks of size two or greater in a rank-1 matrix are 0, we have

$$p(\lambda) = \lambda^n + c_1\lambda^{n-1} = \lambda^{n-1}(\lambda + c_1) = \lambda^{n-1}(\lambda - \sum_{i=1}^{n} a_{ii})$$

which, when set to zero, i.e., $p(\lambda) = 0$, has roots $\lambda_1 = \sum_{i=1}^{n} a_{ii}$ and $\lambda_2 = \lambda_3 = \cdots = \lambda_n = 0$. If $\lambda_1 \neq 0$, then λ_1 must be simple since the remaining $n-1$ eigenvalues of A are all equal to zero. □

Lemma 4.4.3 *For a Markov-Type system $Ax = 0$ with splitting $A = D-L-U$, where D, L and U are respectively the diagonal, negated lower triangular and negated upper triangular part of A, the iteration matrix T_{gs} for the forward (respectively backward) point Gauss-Seidel method has spectrum $\sigma(T_{gs}) = \{1, 0\}$ if the matrix U (respectively L) has rank one. Furthermore,the dominant eigenvalue $\lambda_1 = 1$ is simple if the Markov chain is irreducible.*

Proof We prove the forward version only. The Gauss-Seidel iteration matrix, $T_{gs} = (D-L)^{-1}U$, has rank one when the matrix U has rank one. In this case, from the previous lemma, matrix T_{gs} has at most one nonzero eigenvalue. Also, since π satisfies $A\pi = 0$, it also satisfies $\pi = T_{gs}\pi$. Thus $\lambda_1 = 1$ is an eigenvalue of the iteration matrix T_{gs} corresponding to the stationary distribution vector π. We therefore conclude that $\sigma(T_{gs}) = \{1, 0\}$ i.e., all the eigenvalues of T_{gs} are equal to 0 except the single nonzero eigenvalue 1. The fact that the eigenvalue $\lambda_1 = 1$ is simple can be inferred also from the previous lemma. □

We point out in the following corollary that matrix T_{gs} is idempotent.

Corollary 4.4.1 *The Gauss-Seidel iterative matrix $T_{gs} = (D-L)^{-1}U$ for an irreducible Markov chain A is idempotent, where $A = D-L-U$, $(D-L)$ is nonsingular and $rank(U) = 1$.*

Proof When $rank(N) = 1$, then $rank(T_{gs}) = 1$. So with lemma 4.4.1, matrix T_{gs} has full rank factorization $T_{gs} = uv^T$ with nonzero column vectors u and v. With lemma 4.4.3, its nonzero eigenvalue is 1. That means $v^T u = 1$ since $T_{gs}u = uv^T u = (v^T u)u$ $(u \neq 0)$. Therefore $T_{gs}^2 = (uv^T)(uv^T) = u(v^T u)v^T = uv^T = T_{gs}$. □

Theorem 4.4.1 *In an irreducible Markov-Type system $Ax = 0$ with the splitting $A = D - L - U$, where D, L and U are respectively the diagonal, lower negated triangular and upper negated triangular parts of A, if the rank of the matrix U (respectively L) is equal to* 1*, the forward (respectively backward) point Gauss-Seidel method with iteration matrix T_{gs} converges in one iteration for any positive initial vector $x^{(0)} > 0$.*

Proof Notice first that matrix $-(D - L)$ is an M-matrix (see page 171[64]) and any M-matrix has the property that its inverse is nonnegative, so $(D-L)^{-1} \leq 0$ and $(D-L)^{-1}U \geq 0$. Note also that each row of matrix $(D - L)^{-1}U$ must have at least one positive element otherwise the solution π does not satisfy the fix-point equation $x = (D-L)^{-1}Ux$. Therefore any initial positive vector $x^{(0)} > 0$, multiplying by matrix $T_{gs} = (D - L)^{-1}U$, results in a positive vector $x^{(1)}$, i.e.

$$x^{(1)} = T_{gs}x^{(0)} > 0.$$

Let x be the normalized $x^{(1)}$, i.e.,

$$x = \frac{x^{(1)}}{||x^{(1)}||_1} = \frac{T_{gs}x^{(0)}}{||T_{gs}x^{(0)}||_1} > 0. \tag{4.1}$$

This implies that vector x is a probability vector with all components being positive. Secondly we have $T_{gs}^2 = T_{gs}$ by corollary 4.4.1. Finally we prove that the positive probability vector (4.1) satisfies the fix-point equation $x = T_{gs}x$. This can be seen from the fact

$$T_{gs}x = T_{gs}\frac{T_{gs}x^{(0)}}{||T_{gs}x^{(0)}||_1} = \frac{(T_{gs})^2x^{(0)}}{||T_{gs}x^{(0)}||_1} = \frac{T_{gs}x^{(0)}}{||T_{gs}x^{(0)}||_1} = x.$$

Thus we proved the theorem. □

This theorem explains why the first motivating example in Section 3 converges in just one iteration for forward Gauss-Seidel with initial vector $x^{(0)}$ or $y^{(0)}$. It is due to the rank-1 property of the iteration matrix T_{gs}. We wish to stress that this rank-1 condition for single iteration convergence also holds for a *block* Gauss-Seidel splitting. Indeed, it holds for *all* regular splittings $A = M - N$ where $M^{-1} \geq 0$ and $N \geq 0$, as long as the iteration matrix $T = M^{-1}N$ has rank 1 because the lemmas, corollaries and theorems given above hold also for rank-1 iteration matrix T. Before concluding this section, we prove that this rank-1 condition is also a *necessary* condition for single iteration convergence when the Markov chain is irreducible.

Theorem 4.4.2 *Consider an irreducible Markov chain A with regular splitting $A = M - N$. If the iterative scheme $x^{(k+1)} = Tx^{(k)}$ with $T = M^{-1}N$ converges in a single iteration for* **any** *initial vector $x^{(0)} > 0$, then $rank(T) = rank(N) = 1$.*

Proof When the iterative scheme $x^{(k+1)} = Tx^{(k)}$ converges in one iteration for any initial positive vector $x^{(0)} > 0$, we have $x^{(1)} = Tx^{(0)}$ (assume $x^{(0)}$ and $x^{(1)}$ are normalized positive vector) and $x^{(1)} = Tx^{(1)}$. So

$$Tx^{(0)} = x^{(1)} = Tx^{(1)} = T(Tx^{(0)}) = T^2x^{(0)},$$

or

$$(T - T^2)x^{(0)} = 0$$

for any initial positive vector $x^{(0)}$. This indicates that the subspace of the general solution to the homogeneous linear system $(T - T^2)x = 0$ has dimension n. The basis for the subspace, for example, can be formed by the n linear independent columns in the Vandermonde matrix

$$V^{n\times n} = \begin{pmatrix} 1 & v_1 & v_1^2 & \cdots & v_1^{n-1} \\ 1 & v_2 & v_2^2 & \cdots & v_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & v_{n-1} & v_{n-1}^2 & \cdots & v_{n-1}^{n-1} \\ 1 & v_n & v_n^2 & \cdots & v_n^{n-1} \end{pmatrix}$$

where $v_i \neq v_j$, for $\forall\ i \neq j$. It thus implies $rank(Ker(T - T^2)) = n$. So $rank(T - T^2) = n - n = 0$. Or $T - T^2 = 0$, i.e, $T = T^2$. In other words, the iteration matrix T is idempotent. Now we prove the rank of T is 1. Assume $rank(T) = r$. Let $Rng(T)$ and $Ker(T)$ be the range and kernel of T respectively. Assume all columns $x_i's$ in matrix $X = [x_1, x_2, \cdots, x_r]$ form the basis for $Rng(T)$ and columns $y_j's$ in matrix $Y = [y_1, y_2, \cdots, y_{n-r}]$ be the basis for $ker(T)$. Then all columns in matrix $B^{n\times n} = [X|Y]$ are linearly independent[44]. That is, $B^{n\times n}$ is nonsingular. And we know $Tx_i = x_i$ for $i = 1, 2, \cdots, r$ and $Ty_j = 0$ for $j = 1, 2, \cdots, n - r$. Then

$$TB = T[X|Y] = [TX|TY] = [X|0]$$

and consequently

$$T = [X|0]B^{-1} = [X|Y] \begin{pmatrix} I_r & 0 \\ 0 & 0 \end{pmatrix} B^{-1} = B \begin{pmatrix} I_r & 0 \\ 0 & 0 \end{pmatrix} B^{-1}$$

where I_r is an identity matrix of order r(see pages 385–386[44]). Now we prove $r = 1$. Since T is similar to I_r, they have the same set of eigenvalues, i.e., $\sigma(I_r) = \{1, 0\}$ with the algebraic multiplicity of 1 being r. But we know from the Perron-Frobenius Theorem which states that the spectral radius of an nonnegative matrix is one of its eigenvalues and has algebraic multiplicity 1 (see page 27[64] and page 673[44]), that the nonnegative matrix T has its unit eigenvalue simple, i.e, $r = 1$. It follows immediately that $rank(T) = rank(I_r) =$

$rank(I_1) = 1.$ □

Note that it is true that the rank-1 iteration matrix $T = M^{-1}N$ of a regular splitting of an irreducible Markov chain is idempotent. But the converse is not necessarily true, i.e, not every idempotent matrix can be an iteration matrix for an irreducible Markov chain. This is because an idempotent matrix can have its eigenvalue 1 with algebraic multiplicity greater than 1, an iteration matrix for an irreducible Markov chain, however, has its eigenvalue 1 being simple (see page 175[64]).

Recall that in example 3, the condition $\sigma(T_{gs}) = \{1, 0\}$ with 1 being simple was not sufficient for the iteration to converge in one iteration. Instead it converged in 2 iterations. This was not just by chance. In that example, $rank(T_{gs}) = 2$ and the rank of its kernel is $rank(Ker(T_{gs})) = 4 - 2 = 2$. This means that the Jordan canonical form for T has one 1×1 Jordan block for the eigenvalue 1, one 1×1 Jordan block and one 2×2 Jordan block for eigenvalue 0, i.e.,

$$T_{gs} = B \left(\begin{array}{c|cc|c} 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \end{array} \right) B^{-1}.$$

Then

$$T_{gs}^2 = B \left(\begin{array}{c|cc|c} 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \end{array} \right)^2 B^{-1} = B \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} B^{-1}$$

and

$$T_{gs}^3 = T_{gs} T_{gs}^2 = B \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} B^{-1} = T_{gs}^2.$$

Hence

$$T_{gs}^2 = T_{gs}^3 = T_{gs}^4 = T_{gs}^5 = \cdots$$

and the iterative process converges in 2 iterations. It is interesting to note that the above fact can be generalized when the iteration matrix T has $rank(T) = r > 1$ and $\sigma(T) = \{1, 0\}$. This is stated in the following theorem.

Theorem 4.4.3 *For an irreducible Markov chain A with regular splitting $A = M - N$, if the matrix $T = M^{-1}N$ has spectrum $\sigma(T) = \{1, 0\}$ and $rank(T) = r > 1$, then the iterative scheme $x^{(k+1)} = Tx^{(k)}$ converges in no more than r iterations.*

Proof Recall that for every matrix $A \in \mathcal{C}^{n\times n}$ with distinct eigenvalues $\sigma(A) = \{\lambda_1, \lambda_2, \cdots, \lambda_s\}$, there is a nonsingular matrix P such that (see page 590[44])

$$P^{-1}AP = J = \begin{pmatrix} J(\lambda_1) & 0 & \cdots & 0 \\ 0 & J(\lambda_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J(\lambda_s) \end{pmatrix}$$

where $J(\lambda_j)$ is the *Jordan segment* for eigenvalue $\lambda_j \in \sigma(A)$; each segment $J(\lambda_j)$ is made up of $t_j = dim(Ker(A - \lambda_j I^{n\times n}))$ *Jordan blocks* $J_i(\lambda_j)$

$$J(\lambda_j) = \begin{pmatrix} J_1(\lambda_j) & 0 & \cdots & 0 \\ 0 & J_2(\lambda_j) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J_{t_j}(\lambda_j) \end{pmatrix} \quad \textit{with} \quad J_i(\lambda_j) = \begin{pmatrix} \lambda_j & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_j \end{pmatrix}$$

and the largest Jordan block in $J(\lambda_j)$ is of size $k_j \times k_j$ where $k_j = index(\lambda_j)$ (see page 590[44]). So when matrix $T = M^{-1}N$ has spectrum $\sigma(T) = \{1, 0\}$, it has two *Jordan segments*, $J(1)$ of size 1×1 for eigenvalue 1 (1 is simple by Perron-Frobenius Theorem) and the other $J(0)$ of size $(n-1) \times (n-1)$ for eigenvalue 0. And when $rank(T) = r > 1$, Jordan segment $J(0)$ has $t_j = index(0) = dim(Ker(T - 0I^{n\times n})) = dim(Ker(T)) = n - r$

Jordan blocks which are actually nilpotent matrices with $1's$ on the super-diagonal and $0's$ elsewhere:

$$P^{-1}TP = J = \begin{pmatrix} J(1) & 0 \\ 0 & J(0) \end{pmatrix} \quad with \;\; J(1) = I_{1\times 1}$$

and

$$J(0) = \begin{pmatrix} J_1(0) & 0 & \cdots & 0 \\ 0 & J_2(0) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J_{n-r}(0) \end{pmatrix}$$

with

$$J_i(0) = \begin{pmatrix} 0 & 1 & & \\ & 0 & \ddots & \\ & & \ddots & 1 \\ & & & 0 \end{pmatrix}.$$

Assume the matrix $J_{n-r}(0)$ has the largest size k_0 among these nilpotent matrices $J_i(0)$ and we prove that k_0 is no larger than r, otherwise assume $k_0 > r$, then we have the following contradiction.

$$\begin{aligned} \sum_{i=1}^{n-r} |J_i(0)| &> |J_1(0)| + |J_2(0)| + \cdots + |J_{n-r-1}(0)| + r \\ &\geq 1 + 1 + \cdots + 1 + r \\ &= (n-r-1) + r \\ &= n - 1 = |J(0)| \end{aligned}$$

where $|\cdot|$ stands for the number of rows or columns of a square matrix. We know, for any nilpotent matrix $J_i(0)$ of size $k \times k$, $J_i(0)^k = J_i(0)^{k+1} = J_i(0)^{k+2} = \cdots = 0$. Thus

$$\begin{aligned}
T^{k_0} &= \left(\begin{array}{c|ccc} 1 & 0 & 0 & 0 \\ \hline 0 & J_1(0) & 0 & 0 \\ 0 & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & J_{n-r}(0) \end{array}\right)^{k_0} \\
&= \left(\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline 0 & J_1(0)^{k_0} & \cdots & 0 \\ 0 & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J_{n-r}(0)^{k_0} \end{array}\right) \\
&= \left(\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline 0 & 0 & \cdots & 0 \\ 0 & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{array}\right)
\end{aligned}$$

which is equal to T^{k_0+1}. Thus the iterative scheme $x^{(k+1)} = Tx^{(k)}$ converges in $k_0 \leq r$ iterations. It converges in no more than r iterations for an Markov chain where r is the rank of the matrix T. □

Finally, observe that with an iteration matrix T having $rank(T) > 1$, an iterative scheme may still converge in a single iteration for some selected initial vectors. However, it will never converge in a single iteration for **any** initial positive vector.

Chapter 5

A Rank Reduction Method For Markov Chains

We have just shown that all iteration matrices based on regular splitting $A = M - N$ such as point and block Gauss-Seidel, converge in a single step when the iteration matrix $M^{-1}N$ has rank one. This leads us to ask whether it is possible to reduce the rank of an iteration matrix with rank $r > 1$ to one and then take advantage of the single step convergence property for the resulting iteration matrix? This is what we seek to achieve in this section.

5.1 Rank Reduction Formula

We first introduce the Wedderburn rank reduction formula and some other formulas needed.

Theorem 5.1.1 (Wedderburn Rank-one Reduction Formula) *Let $A \in \mathbf{R}^{n\times n}$. If $x \in \mathbf{R}^n$ and $y \in \mathbf{R}^n$ are vectors such that $\omega = y^T Ax \neq 0$, then the matrix $B = A - \omega^{-1}Axy^T A$ has rank exactly one less than the rank of A.*

Successive use of this idea provides quite general factorizations of the matrix A. For more detailed information, see HouseHolder [33], or Chu, Funderlic and Golub [15] and the references therein.

Theorem 5.1.2 (Sherman-Morrison Formula[44]) *If $A \in \mathbf{R}^{n\times n}$ is nonsingular and if u and v are $n \times 1$ column vectors such that $1 + v^T Au \neq 0$, then the sum $A + uv^T$ is nonsingular, and*

$$(A + uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^T A^{-1}}{1 + v^T A^{-1}u}.$$

Note that when A is the identity matrix I, we have

$$(I + uv^T)^{-1} = I - \frac{uv^T}{1 + v^T u} \quad \text{when} \;\; 1 + v^T u \neq 0.$$

5.2 Rank Reduction for Computing Stationary Distributions of MCs

We are now ready to derive a rank reduction method for computing stationary distributions of Markov chains. Let $A = M - N$ and $T = M^{-1}N$ and assume $rank(T) = r > 1$. Let e_i be the i^{th} column of the $(n \times n)$ identity matrix. Given two vectors $e_{i_1},\ e_{j_1} \in \mathbf{I}^{n\times n}$ such that

$$\omega = e_{j_1}^T T e_{i_1} = T[j_1, i_1] \neq 0$$

and

$$e_{j_1}^T T^2 e_{i_1} = T[j_1, *]T[*, i_1] \neq \omega,$$

then, by the *Wedderburn Rank-one Reduction Formula*, the matrix

$$T^{(2)} = T - \omega^{-1} T e_{i_1} e_{j_1}^T T \tag{5.1}$$

has rank exactly one less than the rank of T. From the fixed-point equation

$$x = Tx, \tag{5.2}$$

we have

$$x = (T^{(2)} + \omega^{-1} T e_{i_1} e_{j_1}^T T)x, \tag{5.3}$$

or

$$(I - \omega^{-1} T e_{i_1} e_{j_1}^T T)x = T^{(2)} x. \tag{5.4}$$

That is, if $e_{j_1}^T T^2 e_{i_1} \neq \omega$, then

$$x = (I - \omega^{-1} T e_{i_1} e_{j_1}^T T)^{-1} T^{(2)} x = (I - \omega^{-1} T e_{i_1} e_{j_1}^T T)^{-1} (T - \omega^{-1} T e_{i_1} e_{j_1}^T T)x = Hx \tag{5.5}$$

where $rank(H) = rank(T^{(2)}) = rank(T) - 1$ since pre-multiplying the matrix $T^{(2)}$ by a nonsingular matrix does not change its rank. From the *Sherman-Morrison Formula*, we calculate the matrix H as

$$\begin{aligned} H &= (I + \frac{\omega^{-1} T e_{i_1} e_{j_1}^T T}{1 - \omega^{-1} e_{j_1}^T T^2 e_{i_1}})(T - \omega^{-1} T e_{i_1} e_{j_1}^T T) \\ &= T - \omega^{-1} T e_{i_1} e_{j_1}^T T + \frac{\omega^{-1} T e_{i_1} e_{j_1}^T T^2}{1 - \omega^{-1} e_{j_1}^T T^2 e_{i_1}} - \frac{\omega^{-1} T e_{i_1} e_{j_1}^T T \omega^{-1} T e_{i_1} e_{j_1}^T T}{1 - \omega^{-1} e_{j_1}^T T^2 e_{i_1}}. \end{aligned}$$

Multiplying both sides by the scalar $\omega(\omega - e_{j_1}^T T^2 e_{i_1}) \neq 0$, yields

$$\begin{aligned} & \omega(\omega - e_{j_1}^T T^2 e_{i_1}) H \\ = \; & \omega(\omega - e_{j_1}^T T^2 e_{i_1}) T - \omega(\omega - e_{j_1}^T T^2 e_{i_1}) \omega^{-1} T e_{i_1} e_{j_1}^T T + \; \omega(\omega - e_{j_1}^T T^2 e_{i_1}) \frac{\omega^{-1} T e_{i_1} e_{j_1}^T T^2}{1 - \omega^{-1} e_{j_1}^T T^2 e_{i_1}} \\ & - \; \omega(\omega - e_{j_1}^T T^2 e_{i_1}) \frac{\omega^{-1} T e_{i_1} e_{j_1}^T T \omega^{-1} T e_{i_1} e_{j_1}^T T}{1 - \omega^{-1} e_{j_1}^T T^2 e_{i_1}} \\ = \; & \omega(\omega - e_{j_1}^T T^2 e_{i_1}) T - (\omega - e_{j_1}^T T^2 e_{i_1}) T e_{i_1} e_{j_1}^T T + \; \omega T e_{i_1} e_{j_1}^T T^2 - T e_{i_1} (e_{j_1}^T T T e_{i_1}) e_{j_1}^T T \\ = \; & \omega(\omega - e_{j_1}^T T^2 e_{i_1}) T - (\omega - e_{j_1}^T T^2 e_{i_1}) T e_{i_1} e_{j_1}^T T + \; \omega T e_{i_1} e_{j_1}^T T^2 - (e_{j_1}^T T^2 e_{i_1}) T e_{i_1} e_{j_1}^T T \\ = \; & \omega(\omega - e_{j_1}^T T^2 e_{i_1}) T - \omega T e_{i_1} e_{j_1}^T T + \omega T e_{i_1} e_{j_1}^T T^2 \\ = \; & \omega(\omega - e_{j_1}^T T^2 e_{i_1}) T + \omega T e_{i_1} e_{j_1}^T T (T - I). \end{aligned}$$

Dividing both sides by the scalar $\omega(\omega - e_{j_1}^T T^2 e_{i_1}) \neq 0$ thus yields

$$H = T + \frac{T e_{i_1} e_{j_1}^T T(T - I)}{\omega - e_{j_1}^T T^2 e_{i_1}}. \tag{5.6}$$

Notice that the matrix H in (5.6) has rank exactly one less than the matrix T in (5.2). This rank reduction process may be repeated until finally, a rank-1 matrix H is obtained. In order to simplify the calculation of the matrix H in formula (5.6) and to save memory, assume that the indices of the nonzero columns in N form the set $S = \{k_1,\ k_2,\ \cdots,\ k_r\}$. Assume further that $rank(N) = r$ which means that all nonzero columns of N are linearly independent. Let e_j be the j^{th} column of the identity matrix $I^{n\times n}$ and

$$\begin{aligned} \beta &= e_{j_1}^T T^2 e_{i_1} = T[j_1, *]T[*, i_1] \neq \omega \\ \beta_j &= e_{j_1}^T T^2 e_j = T[j_1, *]T[*, j] \\ \omega_j &= e_{j_1}^T T e_j = T[j_1, j]. \end{aligned}$$

Then for the matrix H in (5.6), we have

$$\begin{aligned} (\omega - \beta)He_j &= (\omega - \beta)Te_j + Te_{i_1}e_{j_1}^T T^2 e_j - Te_{i_1}e_{j_1}^T Te_j \\ &= (\omega - \beta)Te_j + \beta_j Te_{i_1} - \omega_j Te_{i_1} \\ &= (\omega - \beta)Te_j + (\beta_j - \omega_j)Te_{i_1}. \end{aligned}$$

It follows that

$$He_j = Te_j + \frac{\beta_j - \omega_j}{\omega - \beta} Te_{i_1}, \quad \text{for } \forall j \in S,$$

or

$$H[*, j] = T[*, j] + \frac{\beta_j - \omega_j}{\omega - \beta} T[*, i_1], \quad \text{for } \forall j \in S. \tag{5.7}$$

Thus the matrix H is generated column by column from the previous iteration matrix T.

Note that the matrix H formed by use of equation (5.7) is the same as that formed by equation (5.6) and has the following properties.

Proposition 5.2.1 *If matrix T (5.2) has zero column j, then the column j of matrix H formed by (5.7) is also zero.*

Proof Assume the j^{th} column of matrix T is zero, i.e, $T[*, j] = 0$. Then

$$\begin{aligned}
H[*, j] &= T[*, j] + \frac{\beta_j - \omega_j}{\omega - \beta} T[*, i_1] \\
&= T[*, j] + \frac{T[j_1, *]T[*, j] - T[j_1, j]}{\omega - \beta} T[*, i_1] \\
&= T[*, j] + \frac{0 - 0}{\omega - \beta} T[*, i_1] \\
&= T[*, j] + 0 = 0.
\end{aligned}$$

□

This property indicates that no additional nonzero columns are generated in successive rank reduction steps.

Proposition 5.2.2 *The matrix H in (5.7) satisfies that $H[*, i_1] = 0$.*

Proof It is straightforward that

$$\begin{aligned}
H[*, i_1] &= T[*, i_1] + \frac{\beta_{i_1} - \omega_{i_1}}{\omega - \beta} T[*, i_1] \\
&= T[*, i_1] + \frac{T[j_1, *]T[*, i_1] - T[j_1, i_1]}{\omega - \beta} T[*, i_1] \\
&= T[*, i_1] + \frac{\beta - \omega}{\omega - \beta} T[*, i_1] \\
&= T[*, i_1] - T[*, i_1] = 0.
\end{aligned}$$

□

This property indicates that one zero column arises in successive iteration matrices during the rank reduction process. Combining these two properties, it is apparent that each successive iteration matrix H has one more zero column than the iteration matrix that preceded it.

We note that in the rank reduction process, it is possible that the iteration matrix has a subset of columns linear dependent. That is, the rank of the iteration matrix is less than the number of nonzero columns. If this is the case, then any nonzero columns j which are linear dependent with column i_1 also satisfies $H[*, j] = 0$. This can be seen in the following proposition.

Proposition 5.2.3 *If the j^{th} column of matrix T in (5.2) is linear dependent with column i_1, then matrix H in (5.7) satisfies $H[*, j] = 0$.*

Proof From the condition stated above, assume that there is a constant c such that $T[*, j] = c \times T[*, i_1]$. Then

$$\begin{aligned}
H[*, j] &= T[*, j] + \frac{\beta_j - \omega_j}{\omega - \beta} T[*, j] \\
&= T[*, j] + \frac{T[j_1, *]T[*, j] - T[j_1, j]}{\omega - \beta} T[*, j] \\
&= cT[*, i_1] + \frac{cT[j_1, *]T[*, i_1] - cT[j_1, i_1]}{\omega - c\beta} T[*, i_1] \\
&= cT[*, i_1] + c\frac{\beta - \omega}{\omega - \beta} T[*, i_1] \\
&= cT[*, i_1] - cT[*, i_1] = 0.
\end{aligned}$$

□

Before introducing another property for matrix H, let us fist state a lemma.

Lemma 5.2.1 *For a Markov-Type system $Ax = 0$ with splitting $A = M - N$, where M is nonsingular and $N \neq 0$, the fixed-point equation $x = Tx$, $T = M^{-1}N$, has a unique normalized positive solution vector π if the Markov chain A is irreducible.*

Proof Assume there are two different solution vectors $\pi^{(1)}$ and $\pi^{(2)}$ such that $\pi_j^{(i)} > 0$, $j =$

$1, 2, \cdots, n, i = 1, 2$; $\sum_{j=1}^{n} \pi_j^{(i)} = 1, i = 1, 2$ and

$$\pi^{(1)} = T\pi^{(1)} \ \ and \ \ \pi^{(2)} = T\pi^{(2)},$$

or

$$\pi^{(1)} = M^{-1}N\pi^{(1)} \ \ and \ \ \pi^{(2)} = M^{-1}N\pi^{(2)}.$$

Multiply both sides by the nonsingular matrix M,

$$M\pi^{(1)} = N\pi^{(1)} \ \ and \ \ M\pi^{(2)} = N\pi^{(2)},$$

or

$$(M - N)\pi^{(1)} = A\pi^{(1)} = 0 \ \ and \ \ (M - N)\pi^{(2)} = A\pi^{(2)} = 0$$

which indicates that the original Markov system $Ax = 0$ has two different solution vectors. This contradicts the irreducibility of the Markov chain A. □

Proposition 5.2.4 *The equation $x = Hx$ with matrix H formed by (5.7) preserves the normalized positive solution to the original equation (5.2).*

Proof Based on the above lemma, we assume that the unique positive solution to the original equation $x = Tx$ (5.2) is the vector $\pi^{(1)}$. We then prove first that $\pi^{(1)}$ is also a solution to the equation $x = Hx$ with matrix H formed by (5.7). This is obvious in the rank reduction procedure. See equations (5.2)(5.3)(5.4) (5.5)(5.6)(5.7). Secondly we prove that there is only one normalized positive solution vector to satisfy the equation $x = Hx$. Assume there are two different normalized positive solution vectors $\pi^{(1)}$ and $\pi^{(2)}$ such that

$$\pi^{(1)} = H\pi^{(1)} \ \ and \ \ \pi^{(2)} = H\pi^{(2)}$$

where H is formed by (5.7). Note that matrix H in equation (5.5)(5.6) is the same as that in equation (5.7), so equation (5.5) has two different normalized positive solution vectors

$\pi^{(1)}$ and $\pi^{(2)}$ as well. Multiplying both sides in equation (5.5) by a nonsingular matrix $(I - \omega^{-1} T e_{i_1} e_{j_1}^T T)$ does not change its solution space. This means equation (5.4) has two different solutions $\pi^{(1)}$ and $\pi^{(2)}$ also. So does equation (5.3). Substituting the matrix $T^{(2)}$ in equation (5.3) by that in the equation (5.1) results in the equation (5.2). Thus equation (5.2) has two different normalized positive solution vectors $\pi^{(1)}$ and $\pi^{(2)}$. This contradicts the above lemma. Therefore equation $x = Hx$ with matrix H formed by (5.7) has a unique normalized positive solution vector. □

When the matrix M is a banded matrix, its inverse can be efficiently computed by means of an LU decomposition $M = LU$.

5.2.1 Algorithm

The overall pseudo code for the rank reduction algorithm is presented in Figure 5.1. It is appropriate for Markov matrices having low rank iteration matrices.

5.2.2 Uniqueness, Irreducibility and Complexity Issues

We have already shown the rank reduction algorithm. Now we ask ourselves questions concerning the uniqueness, irreducibility and other related issues.

Uniqueness of Solution

The rank reduction method starts with the fixed-point equation $x = Tx$ with $rank(T) = r > 1$. With $r - 1$ rank reduction steps resulting in a rank-1 iteration matrix H, it still preserves the unique normalized positive solution to the original equation $x = Tx$. This is because, by property 5.2.4, the equation $x = Hx$ with $rank(H) = r - 1$ formed by rank reduction procedure (5.7) preserves the unique normalized positive solution to the original equation $x = Tx$. Similarly, all successive matrices H $(1 < rank(H) < r)$ generated by the rank reduction procedure (5.2)(5.5)(5.6)(5.7) preserve the unique solution also. Therefore, when the matrix H is reduced to rank 1, it preserves the unique solution π to the original equation $x = Tx$ as well.

Input:

1. $A = M - N$: Markov-Type matrix A of size $n \times n$ with splitting $M - N$;
2. r : The rank of the matrix N;
3. S : The set of all nonzero column indices in matrix N.

Output: The stationary distribution π.

```
1.Initialization: Set $H = 0^{r \times r}$; Form $T = M^{-1}N$     // by LU decomposition
2.If $(r == 1)$    go to step 14
3.Do
4.   Choose vectors $e_{i_1}$ and $e_{j_1}$ such that
5.        $\omega = T[j_1, i_1] \neq 0$ and $\beta = e_{j_1}^T T^2 e_{i_1} \neq \omega$
6.   Calculate: $\omega_j = T[j_1, j]$   and   $\beta_j = T[j_1, *]T[*, j]$,  $j \in S - \{i_1\}$
7.   For   $\forall\ j \in S - \{i_1\}$
8.        $H[*, j] = T[*, j] + \frac{(\beta_j - \omega_j)}{(\omega - \beta)} * T[*, i_1]$          // rank-1 reduction
9.   End
10.  Set  $S := S - \{i_1\}$
11.  Set  $r := r - 1$
12.  Set  $T := H$
13.While ( $r > 1$ )
14.Normalization: Select any $j \in S$ and form $\pi = \frac{H[*,j]}{||H[*,j]||_1}$
15.End.
```

Figure 5.1: Rank Reduction Method for Calculating Stationary Distributions

Concerned with the normalization step (line 12) in the pseudo code of the rank reduction algorithm, we only need to prove that any nonzero column in the final matrix H of rank-1 has all elements positive. Otherwise assume that nonzero column j has zero element in row i, i.e, $T[i, j] = 0$, then all elements in row i of matrix H are zero because the matrix H has rank one. Thus the solution π to the original equation $x = Tx$ does not satisfy the equation $x = Hx$ since $\pi_i \neq 0$ but the row i of the product $H\pi$ is equal to zero. Therefore, the normalization of any nonzero column results in the solution to the original equation. This implies that no initial approximate solution vector in the rank reduction algorithm is needed.

Irreducibility

One might wonder whether the iteration matrix H has to be irreducible? Our answer is *NO*. The only requirements for the rank-1 reduction method to function correctly are as follows.

1. The *Markov chain* must be irreducible: this guarantees that all the elements in the solution vector are strictly positive, i.e.,

$$\pi_i > 0, \quad i = 1, 2, \cdots, n.$$

   and this positive solution vector is unique except for some constant multipliers.

2. The initial matrix splitting $A = M - N$ must not result in an initial iteration matrix $T = 0$ for otherwise $rank(T) = 0$ and we can go no further.

3. At each reduction step, we must have $\omega \neq 0$ and $\beta \neq \omega$.

Complexity

To determine the computational complexity of the algorithm, we proceed as follows.

- Assume the cost of initializing $T = M^{-1}N$ is C_1. This is done once at the beginning.

- The reduction from rank r to rank $r-1$ requires $(r-1)(r+1+n)$ multiplications and $(r-1)(r+1+n)$ additions/subtractions to compute H. So $r-1$ reduction steps requires

$$\sum_{i=1}^{r-1} i(i+2+n)$$

  multiplications and

$$\sum_{i=1}^{r-1} i(i+2+n)$$

additions/subtractions. Evaluating these summations gives

$$\frac{1}{2}(r^2 - r)n + (\frac{1}{3}r^3 + \frac{1}{2}r^2 - \frac{5}{6}r)$$

multiplications and the same number for additions/subtractions. Let their sum be C_2

- The cost of normalizing a vector in step 10 is n multiplications and $n-1$ additions. Let $C_3 = n + n - 1$.

Therefore the overall cost is $C_1 + C_2 + C_3$. For the calculation of the inverse of a general banded matrix (with upper band p and lower band q), an LU factorization takes about $2npq$ flops. See Golub and Van Loan [29]. Since we assume $r \ll n$, the overall computational complexity is approximately $O(r^2 n)$.

In order to make the algorithm as efficient as possible, it is wise to choose M so that it contains the major part of Q and the remaining part N has comparatively low rank. However, there is the obvious trade-off in that it is vital to ensure that the inverse of M is easy to obtain.

5.2.3 A Small Example

The following example illustrates the detailed steps of the algorithm. Consider a CTMC whose infinitesimal generator Q is given below.

$$Q = \begin{pmatrix} -5.00 & 2.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 3.00 \\ 2.30 & -5.30 & 3.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 1.50 & -3.30 & 1.80 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 2.40 & -3.60 & 1.20 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.50 & -3.50 & 2.00 & 0.00 & 0.00 \\ 1.10 & 1.30 & 0.00 & 0.00 & 3.50 & -8.60 & 2.70 & 0.00 \\ 1.40 & 2.80 & 3.30 & 0.00 & 0.00 & 2.10 & -13.00 & 3.40 \\ 2.10 & 1.60 & 1.50 & 1.10 & 0.00 & 0.00 & 1.90 & -8.20 \end{pmatrix}$$

The stationary distribution of this Markov chain is given by π where

$$\pi = [0.0989\ 0.1576\ 0.3363\ 0.2239\ 0.1037\ 0.0270\ 0.0116\ 0.0410].$$

Based on the splitting $Q^T = M - N$ where M and N are respectively

$$M = \begin{pmatrix} -5.00 & 2.30 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 2.00 & -5.30 & 1.50 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 3.00 & -3.30 & 2.40 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 1.80 & -3.60 & 1.50 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.20 & -3.50 & 3.50 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 2.00 & -8.60 & 2.10 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 2.70 & -13.00 & 1.90 \\ 3.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 3.40 & -8.20 \end{pmatrix}$$

and

$$N = \begin{pmatrix} 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & -1.10 & -1.40 & -2.10 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & -1.30 & -2.80 & -1.60 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & -3.30 & -1.50 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & -1.10 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \end{pmatrix},$$

the iteration matrix T is

$$T = M^{-1}N = \begin{pmatrix} 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.6428 & 1.7552 & 1.4939 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.9192 & 3.2069 & 2.3347 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.5239 & 7.1241 & 5.1906 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.9465 & 4.4120 & 3.5937 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.4428 & 2.0400 & 1.6629 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.1183 & 0.5273 & 0.4307 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.0627 & 0.2165 & 0.1803 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.2612 & 0.7319 & 0.6213 \end{pmatrix}$$

and has rank equal to three. Now, using formula (5.7), we reduce the rank of T by one. Setting

$$e_{i_1} = e_6 = [0\ 0\ 0\ 0\ 0\ 1\ 0\ 0]^T \quad \text{and} \quad e_{j_1} = e_1 = [1\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^T$$

we obtain

$$\omega = e_1^T * T * e_6 = 0.6428 \neq 0 \quad \text{and} \quad \beta = e_1^T T^2 e_6 = T[1,*]T[*,6] = 0.0664 \neq \omega.$$

The rank-2 matrix H is then

$$H = \begin{pmatrix} 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 2.3081 & 1.7603 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 3.9975 & 2.7156 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 8.4349 & 5.8221 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 5.2261 & 3.9859 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 2.4208 & 1.8464 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.6290 & 0.4798 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.2704 & 0.2063 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.9565 & 0.7296 \end{pmatrix}.$$

Repeating the process with new vectors

$$e_{i_1} = e_7 = [0\ 0\ 0\ 0\ 0\ 0\ 1\ 0]^T \quad \text{and} \quad e_{j_1} = e_2 = [0\ 1\ 0\ 0\ 0\ 0\ 0\ 0]^T$$

we obtain

$$\omega = e_2^T * H * e_7 = 3.9975 \neq 0 \quad \text{and} \quad \beta = e_2^T H^2 e_7 = H[2,*]H[*,7] = 3.6787 \neq \omega.$$

The final rank-1 matrix H is then

$$H = \begin{pmatrix} 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 2.4129 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 3.8458 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 8.2069 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 5.4635 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 2.5308 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.6576 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.2827 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.0000 \end{pmatrix}.$$

After normalization, column 8 forms the exact solution π.

5.3 Markov Chains with Low Rank

It is interesting to examine some examples of Markov chains whose structure leads directly to rank-1 iteration matrices. Some examples with *point* rank-1 Gauss-Seidel iteration matrices are illustrated in Figure 5.2. Extensions to these are easy to imagine.

In the first instance, A, all forward transitions are to the same destination state n. There are no other forward transitions. Backward transitions are unrestricted and may be to any state with a lower index. The transition matrix has one nonzero column in the upper

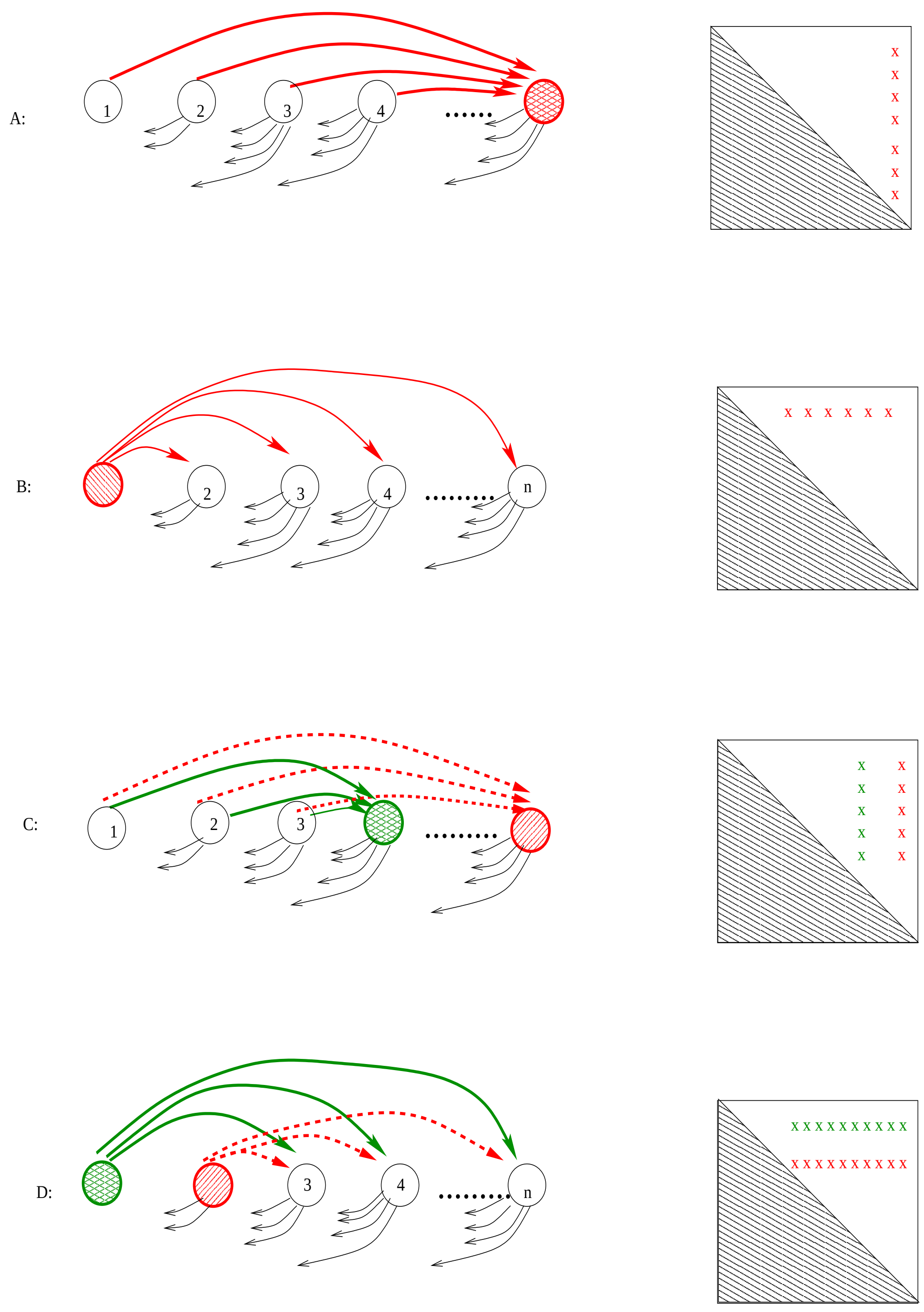


Figure 5.2: Markov Chains With Point Rank-1 Gauss Seidel Iteration Matrices

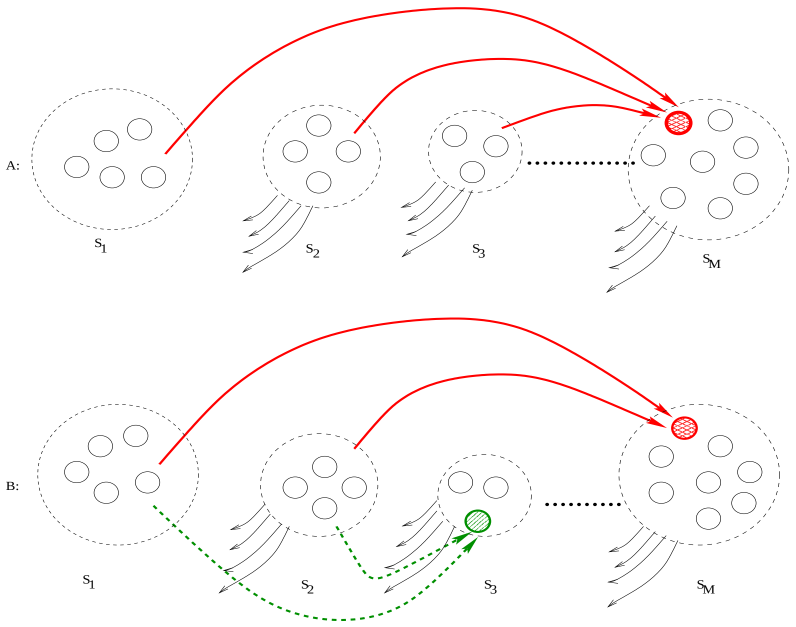
triangular region.

In the second case, B, all forward transitions originate from the same source state 1. Again, there are no restrictions on backward transitions. The transition matrix has one nonzero row in the upper triangular region.

The situation in C is such that forward transitions from a given set of states (1 through 3) are to either state 4 or state n. There are no restrictions on the backward transitions. Now the transition matrix has two nonzero columns in the upper triangular region. When the forward transitions to state 4 are proportional to forward transitions to state n, the point GS iteration matrix has rank-1.

The final situation, D, shows forward transitions originating from states 1 and 2 to the same set of destination states. There are no restrictions on the backward transitions. The transition matrix has two nonzero rows in the upper triangular region. When the forward transitions originating from state 1 are proportional to those originating from state 2, the point GS iteration matrix has rank-1.

Moving on to the *block* Gauss-Seidel case, Figure 5.3 illustrates two instances when the corresponding iteration matrix has rank 1. They are the block versions of cases A and C above.



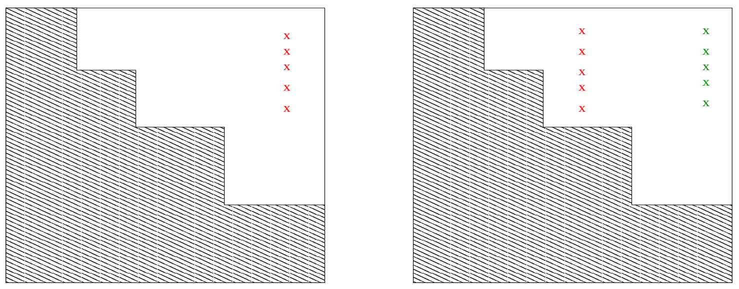


Figure 5.3: Markov Chains With Rank-1 Block Gauss Seidel Iteration Matrices

The first block case illustrates the situation in which forward *intergroup* transitions originating in any group are destined to end up in the last group, S_M. No other forward intergroup transitions are permitted. There are no restrictions on backward (intergroup) transitions nor among the internal transitions in any group. In this case, the transition matrix has one nonzero column in the upper stair-like region. The final situation generalizes the third scenario of the point GS case, to the block case, in a manner analogous to the block case just described.

It is not necessary for a Markov chain to have a structure that is suitable only for Gauss-Seidel type methods. All required is that the splitting $M - N$ is regular and in which M is easily inverted and N has rank one. In the previous examples, M was chosen to be triangular. However, another possibility is the case when M is a narrowly banded matrix. Also, it is possible to extend this to the case in which most of the matrix M is contained in a narrow band around the diagonal, but having some small number of states indexed at the end, that may have many transitions to and from them. The LU decomposition continues within the narrow band until these last few states, but since we only allow a small number of them, a complete reduction of the last two or three rows should not be considered excessive. In anticipation of later results, we show two examples of Markov chains having such a structure for M, but with a matrix N that is not of rank one, but instead is of *low* rank.

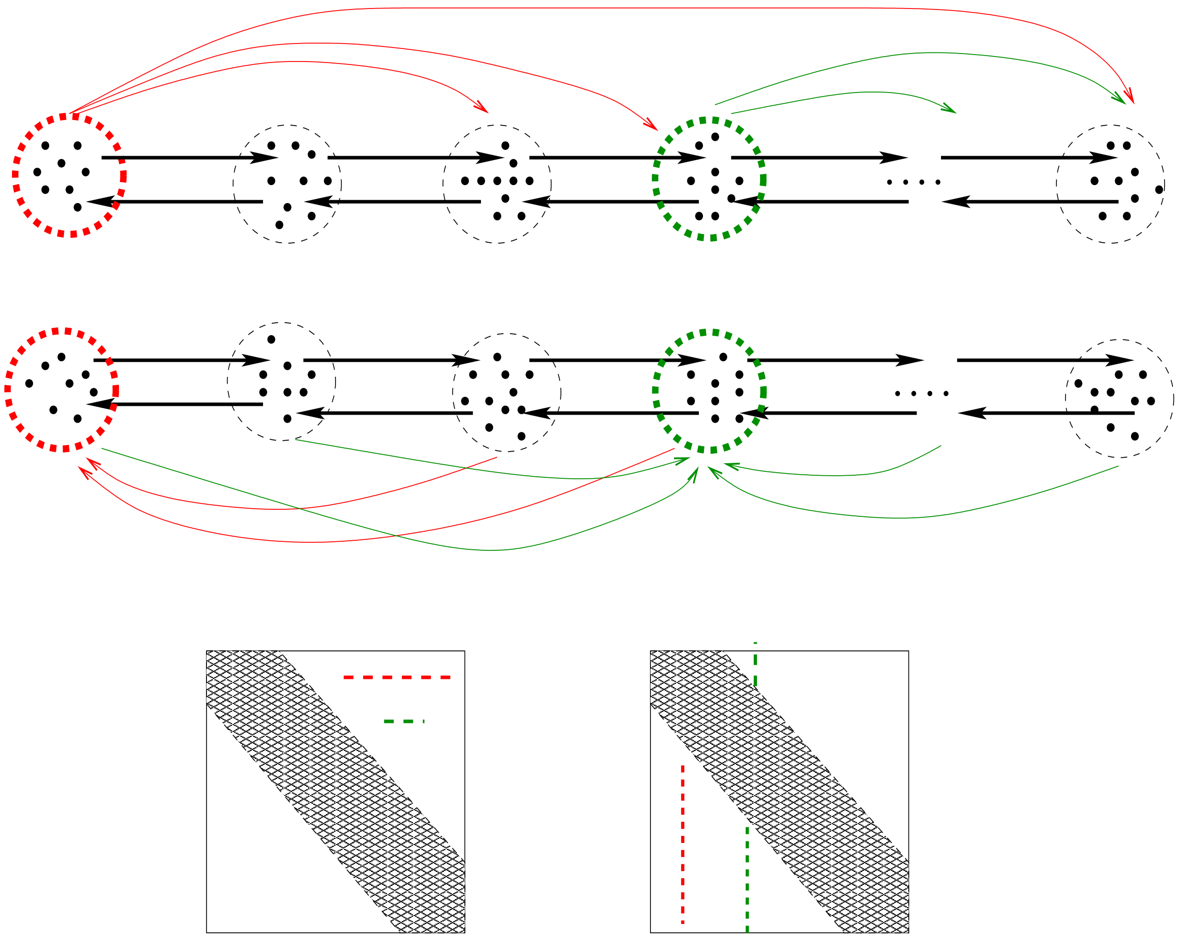


Figure 5.4: Transition Diagrams For Markov Chains With Low-Rank Iterations Matrices: Case I

Figure 5.4 illustrates the first case. Here the matrix M is banded and the elements that constitute N, of rank two, are in two rows or two columns. The top transition diagram is associated with the matrix structure drawn on the left, the bottom one with the matrix on the right. Both are similar to quasi-birth-death process but with some annoying transitions with block-stride larger than one.

Figure 5.5 illustrates the second case, where transition to and from the last two blocks of states may be general.

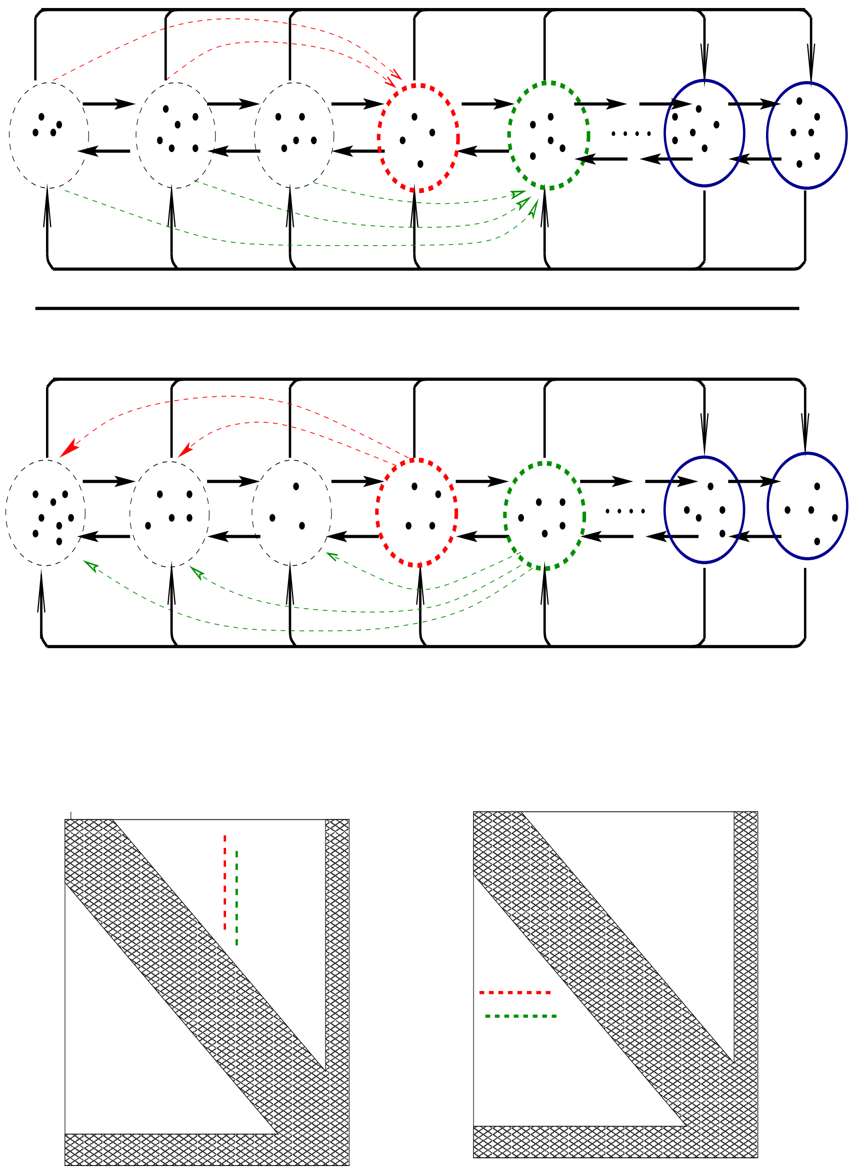


Figure 5.5: Transition Diagrams For Markov Chains With Low-rank Iterations Matrices: Case II

5.4 Testing Rank Reduction Method

5.4.1 Markov Model for a Multiprocessor Failure/Repair System

Consider a dependability model of a multiprocessor system with n processors. Each processor is subject to failures with rate λ. A processor failure is covered with probability q, or not covered with probability p ($p + q = 1$). A covered failure is followed by a brief reconfiguration period which results in a system with one fewer processor with average reconfiguration rate being δ. An uncovered failure is followed by a reboot which results in a fully functioning system with reboot rate being γ. Processors failed are repaired one a time with rate μ. A second failure during reconfiguration, or all processors having failed, results in crash. A crash is repaired with rate β and results in a fully functioning system. Neither reboot nor reconfiguration is performed when the last processor fails. Assume no other event can take place during reboot. All times are assumed to be independently and exponentially distributed. In practice, the reconfiguration time and reboot time are extremely small compared to the time between failures and crash repairs. This model is therefore modeled by a CTMC with the transition diagram shown in Figure 5.6.

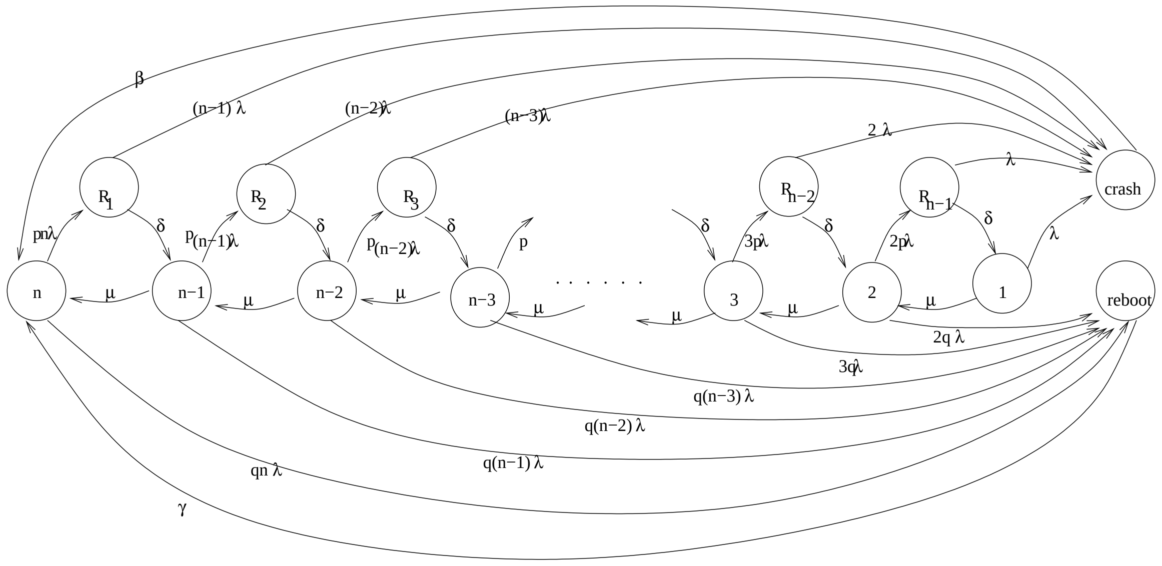


Figure 5.6: A Multiprocessor Failure/Repair Model

Its infinitesimal generator is as follows.

$$Q = \left(\begin{array}{c|ccccccccccccc}
n & * & np\lambda & & & & & & & & & & & nq\lambda \\
r_1 & & * & \delta & & & & & & & & & (n-1)\lambda & \\
n-1 & \mu & & * & (n-1)p\lambda & & & & & & & & & (n-1)q\lambda \\
r_2 & & & & * & \delta & & & & & & & (n-2)\lambda & \\
n-2 & & & \mu & & * & (n-2)p\lambda & & & & & & & (n-2)q\lambda \\
r_3 & & & & & & * & \delta & & & & & (n-3)\lambda & \\
\ldots & & & & & & & & & & & & & \\
3 & & & & & \mu & & * & 3p\lambda & & & & & 3q\lambda \\
r_{n-2} & & & & & & & & * & \delta & & & 2\lambda & \\
2 & & & & & & & \mu & & * & 2p\lambda & & & 2q\lambda \\
r_{n-1} & & & & & & & & & & * & \delta & \lambda & \\
1 & & & & & & & & & & \mu & * & \lambda & \\
crash & \beta & & & & & & & & & & & * & \\
reboot & \gamma & & & & & & & & & & & & *
\end{array}\right).$$

Note that the matrix Q can be split into two parts, one with just the tridiagonal elements and the other containing what it left (nonzeros are in the first column and the last two columns). Thus the initial iteration matrix has rank equal to 3. This CTMC model is taken from Sanders's notes [58]. We can find the similar models from Trivedi's publications [70] as well. One particular set of values for all parameters are: $\lambda = \frac{1.0}{6000.0}$, $\mu = 1.0$, $\delta = \frac{1.0}{0.01}$, $\gamma = \frac{1}{0.5}$, $\beta = \frac{1.0}{20.0}$, $p = 0.95$, $\quad q = 0.05$.

We set the accuracy requirement to be 1.0×10^{-16}. In order to compare the rank reduction method and the Gauss Seidel method, we change the values of n, p and q to see what difference this makes.

From table 5.1, we observe that

- The rank reduction method is much faster than Gauss Seidel.

- The convergence of Gauss Seide depends on the eigenvalue distribution of the iteration matrix and thus depends on the elements of the iteration matrix. It can be observed that, with different values of p, q and λ, Gauss Seidel has different execution

# processors	*# states*	λ	p	q	*time (GS)*	*time (RR)*
20	41	$\frac{1}{6000}$	0.01	0.99	0.052	0.048
20	41	$\frac{1}{100}$	0.01	0.99	0.046	0.050
20	41	$\frac{1}{6000}$	0.99	0.01	0.111	0.051
20	41	$\frac{1}{100}$	0.01	0.99	0.116	0.049
50	101	$\frac{1}{6000}$	0.01	0.99	1.810	0.425
50	101	$\frac{1}{6000}$	0.99	0.01	2.014	0.424
50	101	$\frac{1}{100}$	0.01	0.99	1.879	0.442
50	101	$\frac{1}{100}$	0.99	0.01	6.553	0.410
100	201	$\frac{1}{100}$	0.01	0.99	24.324	1.168
100	201	$\frac{1}{100}$	0.99	0.01	26.712	1.310
100	201	$\frac{1}{6000}$	0.01	0.99	—	1.437
100	201	$\frac{1}{6000}$	0.99	0.01	—	1.395

Table 5.1: Testing On Rank Reduction

times to converge. On the other hand, the rank reduction method is independent of the elements of the iteration matrix. Once the number of states of the matrix is determined, rank reduction takes almost the same amount of time to terminate no matter what these parameters are.

- Gauss Seidel takes long time for convergence with the high accuracy requirement, i.e., with the tolerance being 1.0×10^{-16}. Due to the high reliability of the multiprocessor system, several states have stationary probabilities less than 10^{-250} when there are 100 multiprocessors. That is why we would like to set such a small tolerance.

- The rank reduction method is independent of the tolerance. It always terminates in a finite number of steps.

Chapter 6

Lumpability, Weak Lumpability, Quasi-lumpability and NCD

We firstly introduce several concepts such as lumpability, weak lumpability, quasi-lumpability and near-complete decomposability. Then we discuss the relationship among them.

Assume that we are given a finite-state homogeneous discrete time Markov chain $\mathcal{X}_n, n = 0, 1, \cdots$, with transition probability matrix (t.p.m) P. Let $S = \{S_1, S_2, \cdots, S_M\}$ be a partition of the state space of this Markov chain and let matrix P be partitioned accordingly

$$P = \begin{pmatrix} P_{11} & P_{12} & \cdots & P_{1M} \\ P_{21} & P_{22} & \cdots & P_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ P_{M1} & P_{M2} & \cdots & P_{MM} \end{pmatrix}. \tag{6.1}$$

Each subset $S_I, I = 1, \cdots, M$, can be considered a state of a new process. If we use $\mathcal{Y}_t$ to denote the state occupied at time t by the new process, then the probability of a transition occurring at time t from state S_I to state S_J is denoted by $p_{S_I S_J}(t)$ and is given by

$$p_{S_I S_J}(t) = Pr\{\mathcal{Y}_t = S_J | \mathcal{Y}_{t-1} = S_I \wedge \mathcal{Y}_{t-2} = S_K \wedge \cdots \wedge \mathcal{Y}_0 = S_L\}. \tag{6.2}$$

This new process is called a *lumped* process and S_I a *lumped* (or *macro* [8]) state. Note that, $\mathcal{Y}_t$ is not necessarily Markovian nor even homogeneous. The lumped process is again a first-order homogeneous Markov chain only if

$$p_{S_I S_J}(t) = p_{S_I S_J} = Pr\{\mathcal{Y}_t = S_J | \mathcal{Y}_{t-1} = S_I\}, \quad \forall t \geq 0. \tag{6.3}$$

Let the lumped process have transition probability matrix K with element k_{ij} equal to $p_{S_I S_J}$, i.e.,

$$K = \begin{pmatrix} k_{11} & k_{12} & \cdots & k_{1M} \\ k_{21} & k_{22} & \cdots & k_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ k_{M1} & k_{M2} & \cdots & k_{MM} \end{pmatrix}. \tag{6.4}$$

6.1 Definitions

Definition 6.1.1 (Lumpable) *A discrete time homogeneous Markov chain $\mathcal{X}_n$ is said to be* **lumpable** *with respect to a given state space partition $S = \bigcup_{I=1}^{I=M} S_I$ with $S_I \bigcap S_J = \emptyset, \forall I \neq J$ if for every initial probability distribution vector $\pi^{(0)}$, the lumped process $\mathcal{Y}_t$ defined by (6.2) is a first-order homogeneous Markov chain and does not depend on the choice of $\pi^{(0)}$.*

In [36] , Kemeny and Snell prove that a homogeneous Markov chain $\mathcal{X}_n$ with transition probability matrix $P = (p_{ij})$, is *lumpable* with respect to a partition $S = \{S_1, S_2, \cdots, S_M\}$ if and only if for all $I, J, 1 \leq I, J \leq M$, there exists real-valued k_{IJ}, $0 \leq k_{IJ} \leq 1$ such that equation (6.5) holds

$$\sum_{j \in S_J} p_{ij} = k_{IJ}, \forall i \in S_I. \tag{6.5}$$

Note that in some papers [22] [8] this necessary and sufficient condition for a Markov chain to be lumpable is used as an alternative definition for *lumpability.* The following example demonstrates the concept of *lumpability.*

Example 1. Let

$$P = \left(\begin{array}{cc|cc} 0.25 & 0.25 & 0.30 & 0.20 \\ 0.30 & 0.20 & 0.20 & 0.30 \\ \hline 0.25 & 0.30 & 0.20 & 0.25 \\ 0.35 & 0.20 & 0.25 & 0.20 \end{array}\right)$$

with the partition $S_1 = \{1, 2\}$, $S_2 = \{3, 4\}$ indicated in the matrix above, the condition (6.5) is satisfied and the lumped chain is indeed a Markov chain with transition probability matrix:

$$K = \begin{pmatrix} 0.50 & 0.50 \\ 0.55 & 0.45 \end{pmatrix}$$

Unfortunately, since many Markov chains are not necessarily *lumpable*, another important concept, *weak lumpability*, is introduced. Let $\mathbf{\Omega}$ be the set of all probability vectors and $\mathbf{\Omega}_\alpha$ be any proper subset of $\mathbf{\Omega}$.

Definition 6.1.2 (Weak Lumpable) *A discrete time homogeneous Markov chain $\mathcal{X}_n$ is said to be weakly lumpable with respect to a given state space partition $S = \bigcup_{I=1}^{I=M} S_I$ with $S_I \bigcap S_J = \emptyset$, if there exists a proper subset $\mathbf{\Omega}_\alpha$ of $\mathbf{\Omega}$ such that $\forall \pi^{(0)} \in \mathbf{\Omega}_\alpha$, the lumped process $\mathcal{Y}_t$ is Markov homogeneous while for any $\pi^{(0)} \notin \mathbf{\Omega}_\alpha$, the lumped process is not Markov.*

Example 2. Given a transition probability matrix [56]

$$P = \left(\begin{array}{cccc|cc} \frac{3}{14} & \frac{3}{14} & \frac{3}{14} & \frac{3}{14} & \frac{1}{14} & \frac{1}{14} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{3} & \frac{1}{12} & \frac{1}{12} \\ \frac{1}{8} & \frac{3}{8} & \frac{1}{4} & \frac{1}{6} & \frac{1}{24} & \frac{1}{24} \\ \frac{3}{8} & \frac{1}{8} & \frac{1}{4} & \frac{1}{6} & \frac{1}{24} & \frac{1}{24} \\ \hline \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{10} & \frac{1}{10} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{10} & \frac{1}{10} \end{array}\right)$$

and state space partition indicated in matrix P, this Markov chain is *weakly lumpable* with any initial distribution belonging to the set $\mathbf{\Omega}_\alpha$ below

$$\begin{aligned}\mathbf{\Omega}_\alpha &= \{\lambda_1(0,0,0,0,0,1)+\lambda_2(0,0,0,0,1,0)\\ &+ \lambda_3(\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4},0,0)|\lambda_1+\lambda_2+\lambda_3=1,\lambda_i\in[0,1],i=1,2,3\}.\end{aligned}$$

Under what conditions is an Markov chain weakly lumpable with respect to a given state space partition? If it is proved to be weakly lumpable, how do we find the set of all initial distributions for it to be weakly lumpable? There is an active research effort to try to answer these questions. The first investigation of these problems was carried out by Kemeny and Snell [36]. They provided a local necessary and sufficient condition and a useful sufficient condition for weak lumpability. Moneim and Leysieffer [48] gave another but incorrect necessary and sufficient condition as was shown with a counterexample by Rubino and Sericola [55]. Rubino and Sericola also obtained in 1991 a finite characterization of weak lumpability by means of an algorithm which computes the set $\mathbf{\Omega}_\alpha$. In 1996, Peng [49] improved Rubino and Sericola's method [55]. Under a mild condition on its transition probability matrix, a necessary and sufficient condition for a Markov chain to be weakly lumpable and the characterization of the set of initial starting probability vectors which make it weakly lumpable were obtained. Interested readers can resort to their publications [36][48][55] [56][49].

With respect to the requirement on the initial probability distribution for weak lumpability, we would like to ask ourselves the question: Why do we need initial distributions for *weak lumpability* of Markov chains?

First we recall that the newly lumped process (6.2) from an Markov chain is not necessarily Markovian. In order to make it Markovian, formula (6.3) must be satisfied. That is, the probability of a transition to S_J on the next step from any state belonging to the set S_I is independent of the different histories of the process prior to the current step. So when can we expect to ignore the past? There are two cases.

1. *The information gained from the past would not help.*

   This happens, for example, when the probability of moving to the set S_J from a state in set S_I is the same for all states in S_I. Therefore the probabilities of being in each state of S_I would not affect our prediction for the next outcome in the lumped process. This is exactly the condition for *lumpability* (equation (6.5)). That is why we did not see any requirement on the initial probability distribution when *lumpability* is defined in the literature.

2. If the probabilities of ending up in each of the states in S_I are the same no matter what the past information, then again the past has no influence on the next-step transition and can be ignored.

These two scenarios can be clearly illustrated in the following graph.

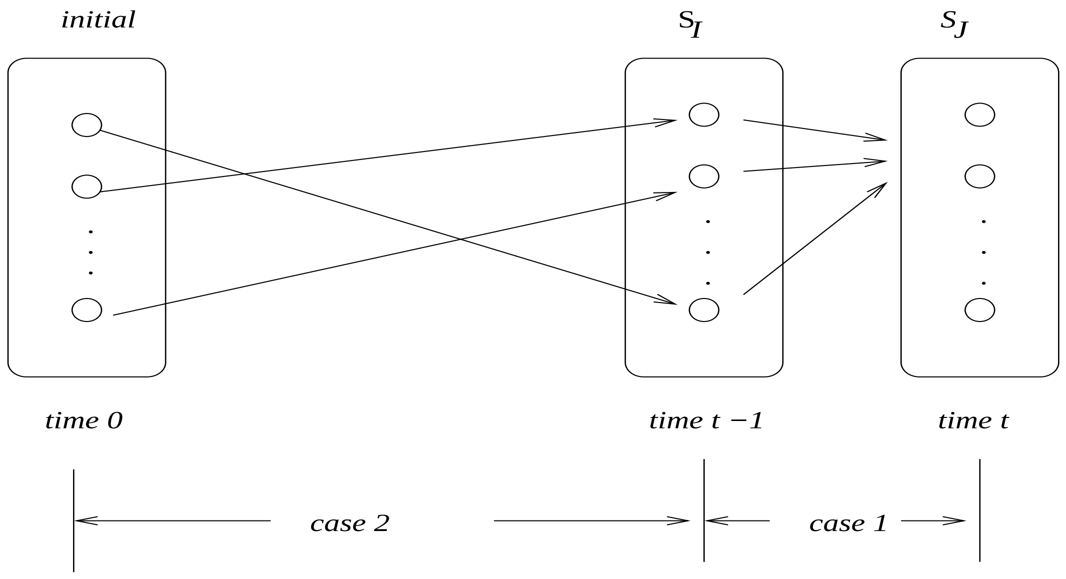


Figure 6.1: Initial Probability Distributions For Weak Lumpability

In case 2, the initial probability distribution is required to be specified. No initial probability distribution however, is required in case 1. In fact in the literature, many papers discuss how to find all initial probability distributions for a Markov chain to be *weakly lumpable*.

Definition 6.1.3 (ϵ-Quasi-Lumpable) *A discrete time homogeneous Markov chain $\mathcal{X}_n$ is said to be ϵ-Quasi-Lumpable [22][27] with respect to a given state space partition $S = \bigcup_{i=1}^{i=M} S_i$ with $S_i \bigcap S_j = \emptyset, \forall i \neq j$ if its transition probability matrix P can be written as $P = P^- + P^\epsilon$ where the P^- is a component-wise lower bound for P that satisfies the lumpability condition (6.6)*

$$\forall S_I, S_J \subset S, \forall s \in S_I : \sum_{s' \in S_J} p^-_{ss'} = k_{IJ}, \forall I \neq J \tag{6.6}$$

and no element in P^ϵ is greater than ϵ in value.

The intention is that P^ϵ is a matrix with more zero elements than P^- and with relatively small non-zero elements. Consider the following example again. Assume that $\lambda_2 = \lambda_3 = \lambda, \lambda_1 = \lambda + \epsilon$ and $\mu_1 = 2\mu + \epsilon, \mu_2 = \mu$.

Example 3. Let [1]

$$P = \left(\begin{array}{cc|ccc|cc}
* & 0 & \lambda_1 & \lambda_2 & 0 & 0 & 0 \\
0 & * & 0 & \lambda_2 & \lambda_2 & 0 & 0 \\
\hline
\mu_1 & 0 & * & 0 & 0 & \lambda_1 & 0 \\
\mu_2 & \mu_2 & 0 & * & 0 & 0 & \lambda_2 \\
0 & 2\mu_2 & 0 & 0 & * & \lambda_3 & 0 \\
\hline
0 & 0 & \mu_1 & \mu_2 & 0 & * & 0 \\
0 & 0 & 0 & \mu_1 & \mu_2 & 0 & *
\end{array}\right),$$

[1]The diagonal elements of P are such that the matrix is stochastic.

$$P^{-} = \left(\begin{array}{cc|ccc|cc} * & 0 & \lambda & \lambda & 0 & 0 & 0 \\ 0 & * & 0 & \lambda & \lambda & 0 & 0 \\ \hline 2\mu & 0 & * & 0 & 0 & \lambda & 0 \\ \mu & \mu & 0 & * & 0 & 0 & \lambda \\ 0 & 2\mu & 0 & 0 & * & \lambda & 0 \\ \hline 0 & 0 & 2\mu & \mu & 0 & * & 0 \\ 0 & 0 & 0 & 2\mu & \mu & 0 & * \end{array}\right) \quad and \quad P^{\epsilon} = \left(\begin{array}{cc|ccc|cc} 0 & 0 & \epsilon & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline \epsilon & 0 & 0 & 0 & 0 & \epsilon & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right).$$

With the partition indicated in the matrix, this Markov chain is not lumpable but is ϵ-quasi-lumpable. Remark that all definitions can be applied to Continuous Time Markov Chains (CTMC) as well. There could be several reasons why we want to investigate the lumpability property of Markov chains.

- In many cases, performance measures of interest depend on the probability of being in certain groups of states, i.e., the probability needs to be computed at a coarser level. In such a case, solving a much smaller lumped chain will give us the information we need.
- Systems in reality are becoming more and more complex. The usual way to solve them is computationally inefficient. But analyzing smaller subsystems individually and then combining them together could save much computation time.

Now we introduce another concept, nearly completely decomposable Markov chains (NCD) that are irreducible chains with their states ordered such that their transition matrices have block structure in which the nonzero elements on the off-diagonal blocks are much small compared with those in diagonal blocks. These Markov chains arise frequently in queueing networks, computer systems and economic models and they can be expressed in the form (6.1) where blocks $P_{i,i}$ are square, of order n_i with $n = \sum_{i=1}^{M} n_i$.

Definition 6.1.4 (Nearly Completely Decomposable) *Given an Markov chain P with* *partition shown in (6.1), then P is nearly completely decomposable when*

$$||P_{i,i}|| = O(1), \quad i = 1, 2, , \cdots, M$$

and

$$||P_{i,j}|| = O(\epsilon), \quad i \neq j$$

where ϵ is a sufficiently small positive number.

Let $P = diag(P_{1,1}, P_{2,2}, \cdots, P_{M,M}) + E$. Partition the stationary distribution vector similarly, $\pi = (\pi_1, \pi_1, \cdots, \pi_M)$. The quantity $||E||_\infty$ is referred to as the degree of coupling and is a measure of the decomposability of the matrix. If it is zero, then the Markov chain is reducible.

6.2 Examples of Lumpable Markov Chains

We remark that lumpable Markov chains are not uncommon in realistic models. Several lumpable Markovian Models are introduced in this section.

Markov Model for Central Servers with Breakdown/Repair

In this Markov model a maximum of m customers can be served by any one of n identical servers in the central server station. Customers arrive at the station with arrival rate λ. Service time is exponentially distributed with mean $\frac{1}{\mu}$. Furthermore, all servers are subject to failure and repair, both times to failure and repair being exponentially distributed with parameters β and γ. Let (i, j) be the state of this queuing system where i denotes the number of customers in the queue and j the number of operational servers. So we have the state transition diagram depicted in Figure 6.2. Let us assume that $m = 4, n = 2$.

If we reorder and partition all states to be

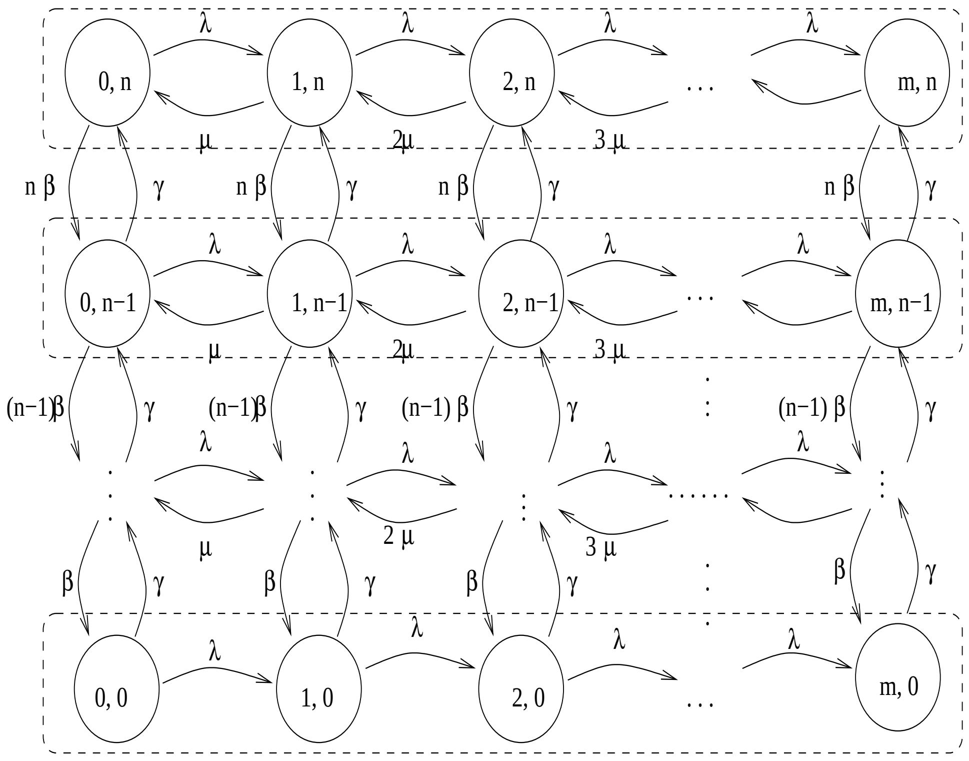


Figure 6.2: A Queueing System With Breakdown And Repair

$S = S_1 \cup S_2 \cup S_3$ with

$$S_1 = \{(0,2),\ (1,2),\ (2,2),\ (3,2),\ (4,2)\},$$

$$S_2 = \{(0,1),\ (1,1),\ (2,1),\ (3,1),\ (4,1)\}$$

and

$$S_3 = \{(0,0),\ (1,0),\ (2,0),\ (3,0),\ (4,0)\},$$

then the transition rate matrix is as follows.

$$Q = \left(\begin{array}{ccccc|ccccc|ccccc}
* & \lambda & 0 & 0 & 0 & 2\beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mu & * & \lambda & 0 & 0 & 0 & 2\beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2\mu & * & \lambda & 0 & 0 & 0 & 2\beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2\mu & * & \lambda & 0 & 0 & 0 & 2\beta & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 2\mu & * & 0 & 0 & 0 & 0 & 2\beta & 0 & 0 & 0 & 0 & 0 \\
\hline
\gamma & 0 & 0 & 0 & 0 & * & \lambda & 0 & 0 & 0 & \beta & 0 & 0 & 0 & 0 \\
0 & \gamma & 0 & 0 & 0 & \mu & * & \lambda & 0 & 0 & 0 & \beta & 0 & 0 & 0 \\
0 & 0 & \gamma & 0 & 0 & 0 & 2\mu & * & \lambda & 0 & 0 & 0 & \beta & 0 & 0 \\
0 & 0 & 0 & \gamma & 0 & 0 & 0 & 2\mu & * & \lambda & 0 & 0 & 0 & \beta & 0 \\
0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 2\mu & * & 0 & 0 & 0 & 0 & \beta \\
\hline
0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & * & \lambda & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & * & \lambda & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & * & \lambda & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & * & \lambda \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & *
\end{array}\right).$$

Obviously this is lumpable and its lumped Markov chain has the following infinitesimal generator

$$K = \begin{pmatrix} * & 2\beta & 0 \\ \gamma & * & \beta \\ 0 & \gamma & * \end{pmatrix}.$$

Based on this smaller Markov chain K, we may calculate the distribution of the average number of operational severs.

Markov Model For An ATM Network

Here we introduce another Markov model [47]. It is a re-routing model in ATM networks graphically illustrated in Figure 6.3.

It is a network with 4 nodes. The arrival process in Queue 1 and Queue 2 consists of a Poisson arrival stream with respective rates λ_1 and λ_2. The service rate is exponentially

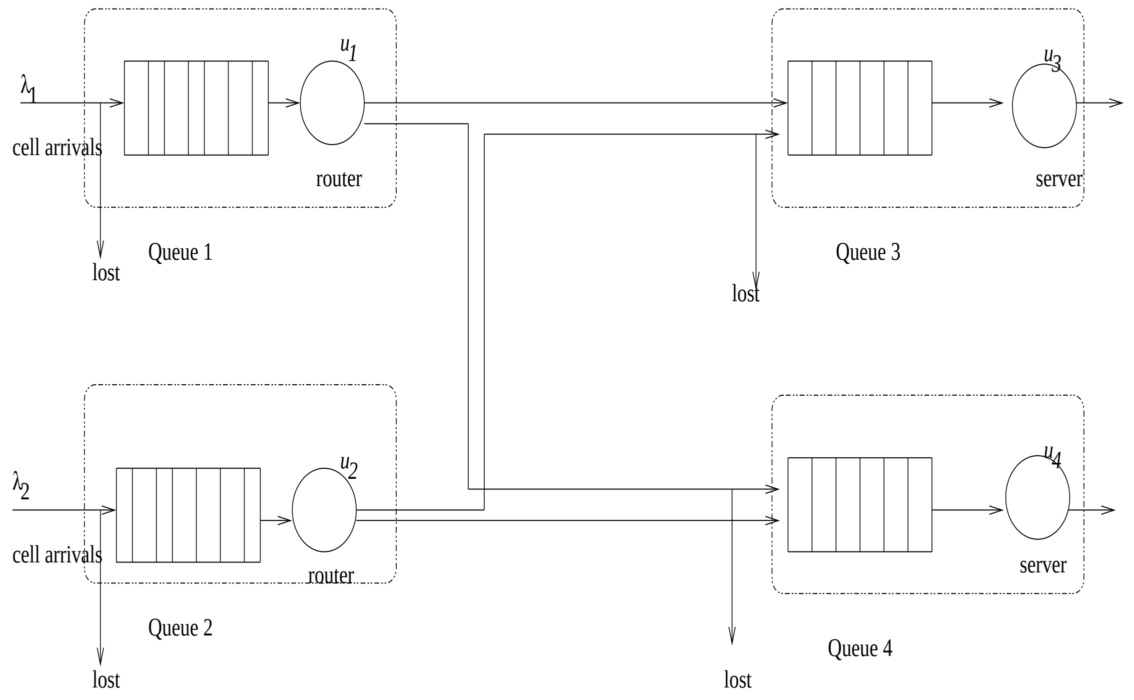


Figure 6.3: Re-routing Model For ATM Networks

distributed in each node with respective rates μ_1, μ_2, μ_3 and μ_4. If the arrivals in Queue 1 find the node buffer is not full, then they wait in the buffer queue for service. Otherwise they are simply dropped by the node. After their service in node 1, they will be routed into node 3 if the buffer in node 3 is not full. Otherwise if the node 4 is not yet filled, they are routed to that node instead. Otherwise, they are lost. The same mechanism is applied to arrival stream in Queue 2 except that they will be first routed to node 4 if it is not full otherwise they are then routed to node 3. The performance measure of interest is the cell loss probability. But our focus is on the steady state probabilities.

When all buffers have capacity of 1, the underlying Markov chain of the model has 16 states which is defined as (i_1, i_2, i_3, i_4) where i_j, $j = 1, 2, 3, 4$, indicates the the number of cells in the buffer j. Let us order and partition all states like

$$\{(0,0,0,0),\ (0,0,0,1),\ (0,0,1,0),\ (0,0,1,1)\}\cup\{(0,1,0,0),\ (0,1,0,1),\ (0,1,1,0),\ (0,1,1,1)\}$$

$$\cup\{(1,0,0,0),\ (1,0,0,1),\ (1,0,1,0),\ (1,0,1,1)\}\cup\{(1,1,0,0),\ (1,1,0,1),\ (1,1,1,0),\ (1,1,1,1)\},$$

then the transition rate matrix is as follows

$$Q = \left(\begin{array}{cccc|cccc|cccc|cccc}
* & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mu_4 & * & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mu_3 & 0 & * & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 \\
0 & \mu_3 & \mu_4 & * & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 \\
\hline
0 & \mu_2 & 0 & 0 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 \\
0 & 0 & 0 & \mu_2 & \mu_4 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 \\
0 & 0 & 0 & \mu_2 & \mu_3 & 0 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 \\
0 & 0 & 0 & \mu_2 & 0 & \mu_3 & \mu_4 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 \\
\hline
0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & 0 & * & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 \\
0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & \mu_4 & * & 0 & 0 & 0 & \lambda_2 & 0 & 0 \\
0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & \mu_3 & 0 & * & 0 & 0 & 0 & \lambda_2 & 0 \\
0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & 0 & \mu_3 & \mu_4 & * & 0 & 0 & 0 & \lambda_2 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & \mu_2 & 0 & 0 & * & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & \mu_2 & \mu_4 & * & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & \mu_2 & \mu_3 & 0 & * & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & \mu_2 & 0 & \mu_3 & \mu_4 & *
\end{array}\right).$$

Obviously it is lumpable with respect to the partition above and its lumped chain has the following infinitesimal generator

$$K = \left(\begin{array}{cc|cc}
* & \lambda_2 & \lambda_1 & 0 \\
\mu_2 & * & 0 & \lambda_1 \\
\hline
\mu_1 & 0 & * & \lambda_2 \\
0 & \mu_1 & \mu_2 & *
\end{array}\right).$$

This is again lumpable with its lumped matrix

$$K^{'} = \begin{pmatrix} * & \lambda_1 \\ \mu_1 & * \end{pmatrix}.$$

When the source side and the destination side are not symmetric or buffers have different capacity, for instance in the Figure 6.4, there are 3 nodes in source side but has only 2 nodes on the destination side, the underlying Markov chain still keeps lumpable. With the lexicographic ordering of all states, its infinitesimal generator is

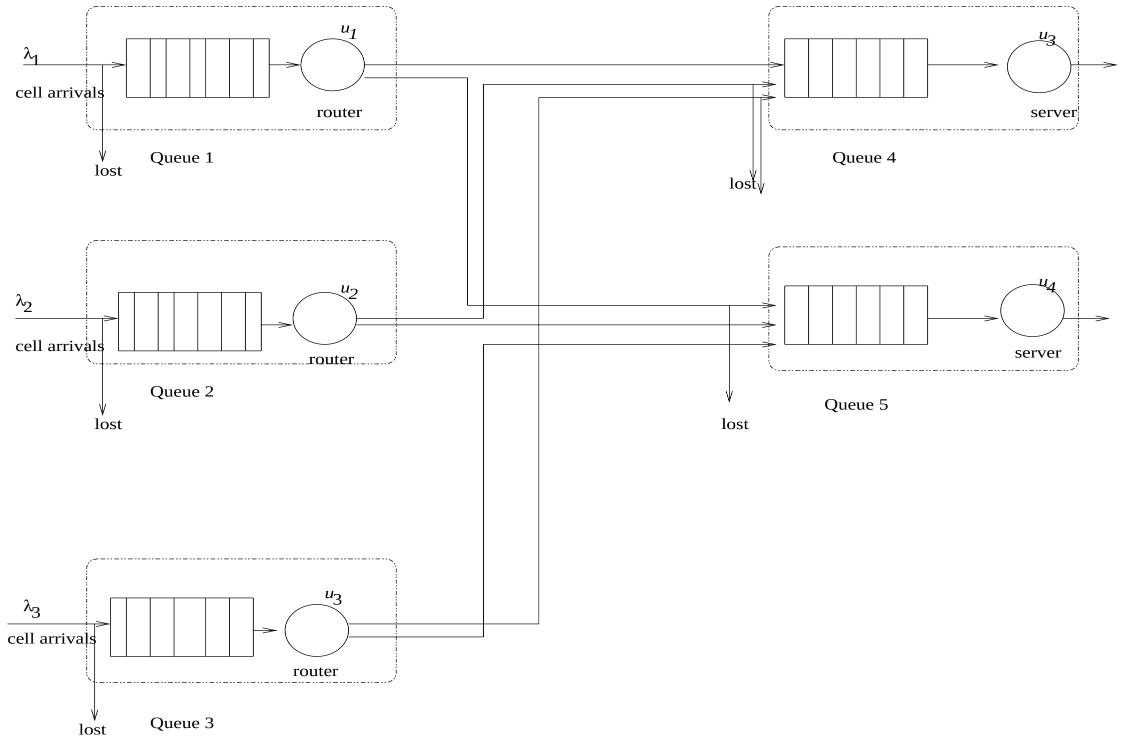


Figure 6.4: Non-symmetric Re-routing Model For ATM Networks

$$Q = \left(\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right)$$

Where

$$A = \left(\begin{array}{cccc|cccc|cccc|cccc}
* & 0 & 0 & 0 & \lambda_3 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mu_5 & * & 0 & 0 & 0 & \lambda_3 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mu_4 & 0 & * & 0 & 0 & 0 & \lambda_3 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 \\
0 & \mu_4 & \mu_5 & * & 0 & 0 & 0 & \lambda_3 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 \\
\hline
0 & \mu_3 & 0 & 0 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 \\
0 & 0 & 0 & \mu_3 & \mu_5 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 & 0 \\
0 & 0 & 0 & \mu_3 & \mu_4 & 0 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 \\
0 & 0 & 0 & \mu_3 & 0 & \mu_4 & \mu_5 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 \\
\hline
0 & 0 & \mu_2 & 0 & 0 & 0 & 0 & 0 & * & 0 & 0 & 0 & \lambda_3 & 0 & 0 & 0 \\
0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & 0 & \mu_5 & * & 0 & 0 & 0 & \lambda_3 & 0 & 0 \\
0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & 0 & \mu_4 & 0 & * & 0 & 0 & 0 & \lambda_3 & 0 \\
0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & 0 & 0 & \mu_4 & \mu_5 & * & 0 & 0 & 0 & \lambda_3 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & \mu_2 & 0 & 0 & \mu_3 & 0 & 0 & * & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & \mu_3 & \mu_5 & * & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & \mu_3 & \mu_4 & 0 & * & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & \mu_3 & 0 & \mu_4 & \mu_5 & *
\end{array}\right),$$

$$B = \left(\begin{array}{cccc|cccc|cccc|cccc}
\lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1
\end{array}\right),$$

$$C = \left(\begin{array}{cccc|cccc|cccc|cccc}
0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1
\end{array}\right)$$

and

$$D = \left(\begin{array}{cccc|cccc|cccc|cccc}
* & 0 & 0 & 0 & \lambda_3 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mu_5 & * & 0 & 0 & 0 & \lambda_3 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mu_4 & 0 & * & 0 & 0 & 0 & \lambda_3 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 \\
0 & \mu_4 & \mu_5 & * & 0 & 0 & 0 & \lambda_3 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 \\
\hline
0 & \mu_3 & 0 & 0 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 \\
0 & 0 & 0 & \mu_3 & \mu_5 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 & 0 \\
0 & 0 & 0 & \mu_3 & \mu_4 & 0 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 \\
0 & 0 & 0 & \mu_3 & 0 & \mu_4 & \mu_5 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 \\
\hline
0 & 0 & \mu_2 & 0 & 0 & 0 & 0 & 0 & * & 0 & 0 & 0 & \lambda_3 & 0 & 0 & 0 \\
0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & 0 & \mu_5 & * & 0 & 0 & 0 & \lambda_3 & 0 & 0 \\
0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & 0 & \mu_4 & 0 & * & 0 & 0 & 0 & \lambda_3 & 0 \\
0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & 0 & 0 & \mu_4 & \mu_5 & * & 0 & 0 & 0 & \lambda_3 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & \mu_2 & 0 & 0 & \mu_3 & 0 & 0 & * & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & \mu_3 & \mu_5 & * & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & \mu_3 & \mu_4 & 0 & * & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_2 & 0 & 0 & 0 & \mu_3 & 0 & \mu_4 & \mu_5 & *
\end{array}\right).$$

This Markov chain is lumpable with respect to partitions which divide the original matrix into 8×8 or 4×4 or 2×2 sub-matrices.

There are plenty of lumpable Markov chains from Queuing networks as well. Figure 6.5 is a more general closed queuing model with parallel queues. If the $p_i,\ i = 1, 2, 3, 4$ are equal and $\mu_i,\ i = 1, 2, 3, 4$ are equal also, then the partitioning indicated in the Figure is lumpable. This may be a hint on how to determine the lumpable partitioning by the use of symmetric structure (parallel structure here, for instance).

6.3 Relationship Among Lumpability, Weak Lumpability, Quasi-lumpability and NCD

The relationship between lumpability and NCD can be observed by several examples below.

- There may be cases where the Markov chain with a given partition is neither lumpable nor weakly lumpable for any initial distribution. The Markov chain with its transition probability matrix [56], for example,

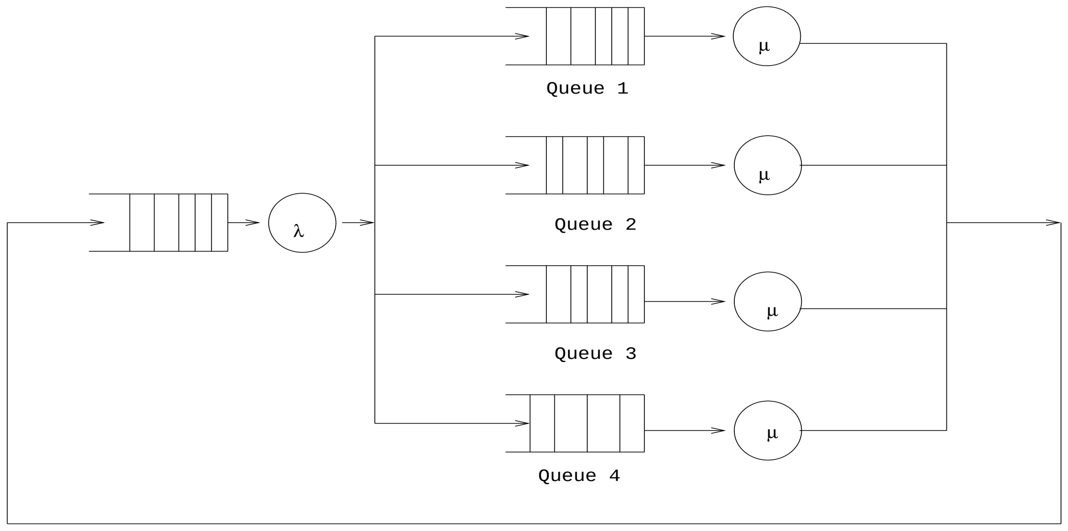
$$P = \left(\begin{array}{c|cc} 1-p & p & 0 \\ \hline 0 & 0 & 1 \\ 1 & 0 & 0 \end{array}\right)$$

  where $0 \leq p \leq 1$, is neither lumpable nor weakly lumpable because for any initial distribution, we have

$$Pr\{\mathcal{X}_{i+1} \in S_1 | \mathcal{X}_i \in S_2, \mathcal{X}_{i-1} \in S_1\} = 0,$$

  and

$$Pr\{\mathcal{X}_{i+1} \in S_1 | \mathcal{X}_i \in S_2, \mathcal{X}_{i-1} \in S_2\} = 1.$$



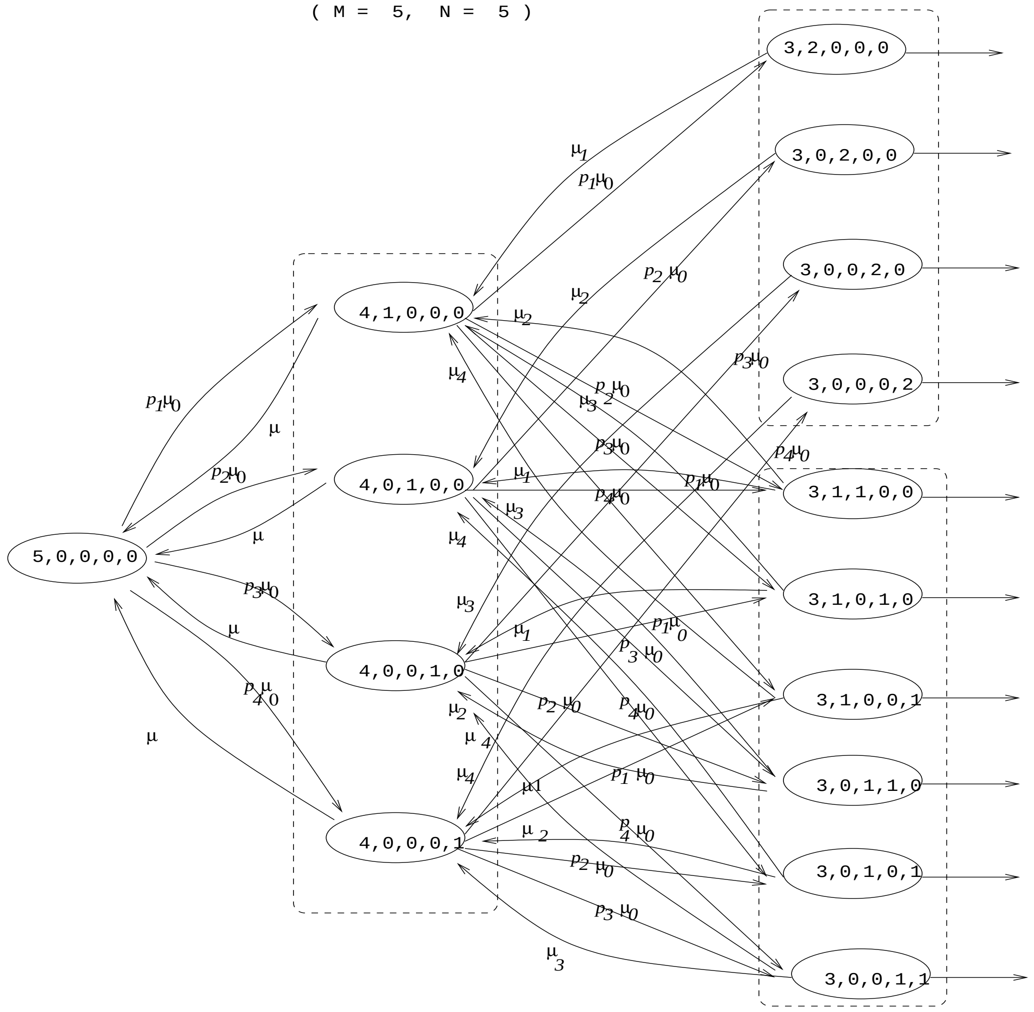


Figure 6.5: Closed Queueing Model With Parallel Queues(1)

- A Markov chain which is lumpable with respect to a given state space partition can be completely decomposable. For example, let

$$P = \left(\begin{array}{cc|cc} 0.75 & 0.25 & 0.00 & 0.00 \\ 0.50 & 0.50 & 0.00 & 0.00 \\ \hline 0.00 & 0.00 & 0.60 & 0.40 \\ 0.00 & 0.00 & 0.45 & 0.55 \end{array}\right).$$

  Obviously this Markov chain is lumpable with respect to the partition indicated in the matrix and is completely decomposable. Actually this is a special case of lumpability.

- A Markov chain which is lumpable with respect to a given state space partition is *not* completely decomposable. For example, let

$$P = \left(\begin{array}{cc|cc} 0.35 & 0.25 & 0.20 & 0.20 \\ 0.30 & 0.50 & 0.10 & 0.10 \\ \hline 0.05 & 0.25 & 0.30 & 0.40 \\ 0.10 & 0.05 & 0.45 & 0.40 \end{array}\right).$$

- A Markov chain which is lumpable with respect to a given state space partition is nearly completely decomposable. For example, let

$$P = \left(\begin{array}{cc|cc} 0.4500 & 0.5497 & 0.0001 & 0.0002 \\ 0.6498 & 0.3500 & 0.0001 & 0.0001 \\ \hline 0.0005 & 0.0005 & 0.4550 & 0.5440 \\ 0.0020 & 0.0030 & 0.3750 & 0.6200 \end{array}\right).$$

  This Markov chain is nearly completely decomposable with $\epsilon = 0.01$.

- A Markov chain which is lumpable with respect to a given state space partition is

NOT nearly completely decomposable. For example, let

$$P = \left(\begin{array}{cc|cc} 0.25 & 0.25 & 0.25 & 0.25 \\ 0.20 & 0.30 & 0.20 & 0.30 \\ \hline 0.25 & 0.30 & 0.20 & 0.25 \\ 0.35 & 0.20 & 0.25 & 0.20 \end{array}\right).$$

Dayar and Stewart proved that NCD Markov chains are *ϵ-quasi lumpable* [22].

The relationship among *lumpability*, *weak lumpability* and ϵ quasi lumpability of Markov chains is illustrated below .

- *lumpability* implies *weak lumpability.*

  This is obviously the case by their definitions.

- *weak lumpability* implies *lumpability* when the Markov chain is reversible and regular.

  This is proved by Snell.

- *lumpability* implies ϵ-quasi lumpability.

  This is the case by specifying $\epsilon = 0$.

- ϵ-quasi lumpability can not infer *lumpability.*

- *weak lumpability* implies ϵ-quasi lumpability when the Markov chain is reversible and regular.

  This is due to the statement 2 and 3 above.

- ϵ-quasi lumpability can not infer *weak lumpability.*

6.4 Identify All Lumpable Partitions

In the previous section, we introduced the concept of lumpability. But we have not yet answered the question on how to identify or construct the lumpable partitions if the Markov chain is indeed lumpable. In this section, we first introduce the procedure presented by Langford B. White, etc [73] and then we suggest improvements and present several empirical results.

6.4.1 White's Algorithm

Let us start with a new defintion. A *n-state* Markov chain X_t is said to be *M-lumpable* if chain X_t is lumpable with respect to the state partition $S = \cup_{I=1}^{I=M} S_I$ with $S_I \bigcap S_J = \emptyset, \forall I \neq J$. Note that it is trivial when $M = n$. Suppose $2 \leq M \leq n$. We now need to test whether an *(M-1)-lumpable* partitioning exists. This test is easily made determining if the following equations hold:

$$k_{MJ} = k_{(M-1)J}, \; J = 1, 2, \cdots, M-2. \tag{6.7}$$

If this is the case, then S_M and $S_{(M-1)}$ are combined together (without loss of generality, by relabeling if necessary) to form a *(M-1)-lumpable*

$$T = \bigcup_{I=1}^{I=M-1} T_I \tag{6.8}$$

where $T_I = S_I, \; I = 1, 2, \cdots, M-2$ and $T_{M-1} = S_{M-1} \bigcup S_M$. The procedure for obtaining all lumpings is shown in Figure 6.6. It is obvious that this procedure in the worst case is exponential. From the following queueing model examples, we observe that most tests executed in the above procedure can be avoided.

6.4.2 Improvements

We first introduce another definition.

Definition 6.4.1 *Let $S = \{S_1, S_2, \cdots, S_M\}$ be a partition of the set of states of a Markov chain, then S is called a block agreement if the state indices of subset I is less than those of subset J, for any $I < J$.*

Input:

1. L: the set of all partitionings ;
2. S, T: the working partitionings;
3. M: the # of subsets for any one partitioning;
4. n: the # of states for the Markov chains;

Output: All lumpable partitionings L.

```
1.  Set $M = n$; Define $S_I = \{I\},\ I = 1, 2, \cdots, M$; Set $L = L \bigcup S$
2.  For each *M-lumpable* partitioning $S \in L$
3.      Check (6.7) on each distinct pair $S_I,\ S_J,\ I \neq J$
4.      If failure
5.          Stop with the process *M-lumpable* but not $(M-1) - lumpable$
5.      Else
6.          Union $S_{M-1}$ and $S_M$; Define *(M-1)-lumpable* partitioning $T$ (6.8)
7.  Set M:= M - 1 and go to line 2 if necessary;
8.  End the procedure.
```

Figure 6.6: Procedure To Obtain Lumpable Partitionings

Several state orderings are considered.

1. lexicographical ordering(including inverse lexicographical ordering)
2. antilexicographical ordering
3. MARCA ordering
4. the ordering used in the Example 4

With the definition of *block agreement* and the orderings listed, considerable saving can be made in determining all possible lumpings. The *block agreement* prohibits the union of subsets like $S_1 = \{1, 2\}$ and $S_3 = \{5, 6\}$. Any subset S_I can only be combined by either subset S_{I-1} or S_{I+1}. For each state ordering above, partitioning is made according to *block agreement*. Let us look at several cases.

Case 1: Our experience shows that very few open queueing models have underlying Markov chains that are lumpable. The only one we found is the loss system. In Fig 6.7, if a customer leaving queue 1 and finding queue 2 full is lost. In this case, the Markov chain is lumpable as indicated in the figure, no matter what the service rates are. The lumped Markov chain can be treated as the chain caused by the first queue/server in isolation. Note the transitions indicated by dashed arrows which result from the loss situation. Without this situation, it is not lumpable. Next, we consider the closed queueing networks.

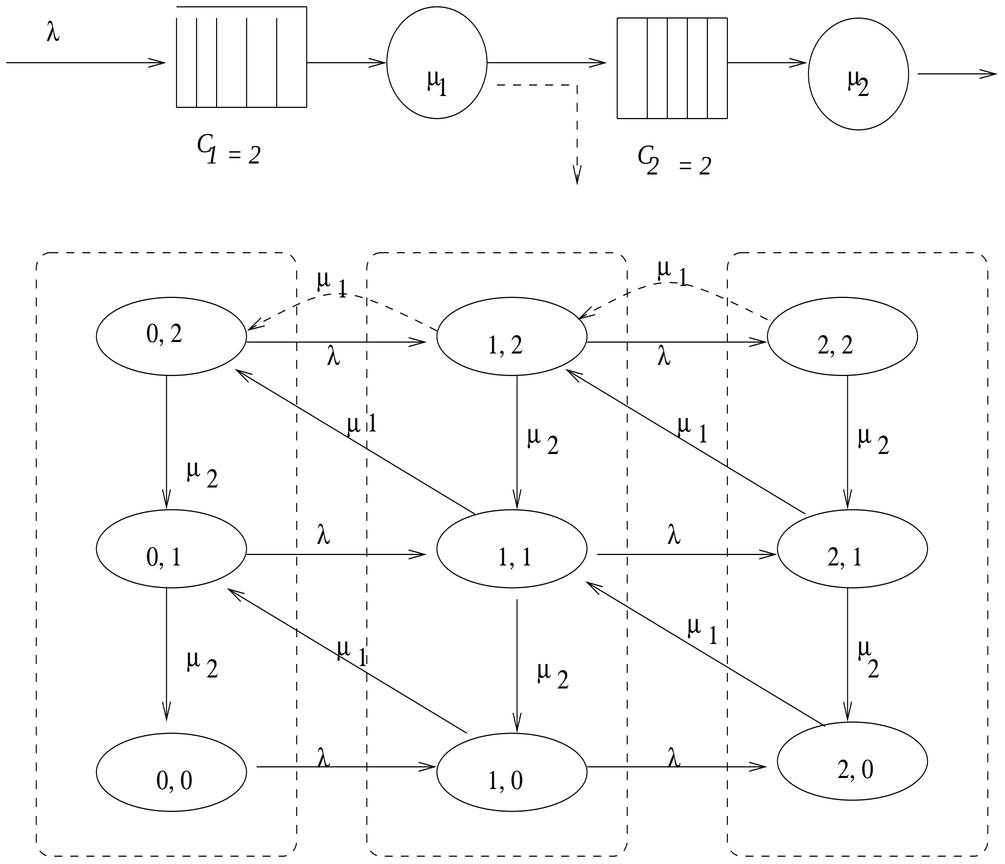


Figure 6.7: Open Queueing Network And Its Transition Diagram

Case 2: This model has 3 server nodes with rates $\mu_i,\ i = 1, 2, 3$ and 4 customers circulate among them in the clockwise direction. Its transition rate diagram is illustrated in the middle of Figure 6.8 Obviously there does not exist any lumpable partitioning even if all μ_i are equal. If however, the capacity of queue in sever $i,\ i = 1, 2, 3$ is 2, 2 and 4 respectively and if any customer leaving queue i finds queue $(i+1)$ *mod* 3 full, strides over it goes directly to the next non-fill queue. In this situation, the underlying Markov chain is lumpable with the partitioning shown in the Figure 6.8.

Case 3**:** This a closed queueing model with parallel queues (See figure 6.11). It may be readily observed that the partition

$$\{(3,0,0)\}\cup\{(2,1,0),(2,0,1)\}\cup\{(1,2,0),(1,0,2)\}\cup\{(1,1,1)\}\cup\{(0,3,0),(0,0,3)\}\cup\{(0,2,1),(0,1,2)\}$$

is a lumpable partitioning if

$$p_{12} = p_{13},\ and\ \mu_2 = \mu_3.$$

It is well known that two single server centers with their own queues have different performance than a single server center with the same 2 servers but only one single queue. So it is impossible to treat the two parallel queues as one bigger imaginary queue without loss of accuracy. The lumpable partitioning above says however that it is possible to group states according to the number of queues with the same length. That is, all states with queue 1 having 2 customers are grouped together; all states with queue 1 having 1 customers are grouped together; and all states with queue 1 having 0 customers are grouped together. See Figure 6.9.

Case 4**:** Figure 6.5 is a more general closed queueing model with parallel queues. If $p_i,\ i = 1,2,3,4$ are equal and $\mu_i,\ i = 1,2,3,4$ are also equal, then the partitioning indicated in the Figure is lumpable. This suggests how lumpable partitioning may be found from the symmetric structure (parallel structure here, for instance).

Heuristics: For most queueing models that are not lumpable for any of the above state ordering, then it is quite possible that the Markov chain is not lumpable. It would appear that all queueing models follow this rule but a rigorous proof is still lacking. when this rule is applicable, it will be more efficient to incorporate it with White's algorithm.

The underlying Markov chains of most queueing models are not lumpable. The loss systems for open queueing networks are the only ones we find to be lumpable. All closed queueing models with parallel queues are however, lumpable. And for the serial (tandem) closed queueing model, if the *skip jump* is allowed, then they are lumpable as well.

6.5 Iterative Aggregation/Disaggregation

When the Markov chain under investigation is lumpable with respect to the partition (6.1), then the aggegation step of the iterative aggregation and disaggregation method is exact and needs only to be formed once. This can be seen in the following theorem.

Theorem 6.5.1 *If the irreducible Markov chain P with partition as in (6.1) can be lumped to form the Markov chain K as in (6.4), then the coupling matrix $C = K$ where C is formed by exact aggregation.*

Proof Recall that the coupling matrix is formed by (see also equation (2.13))

$$c_{IJ} = \phi_i P_{IJ} e$$

where ϕ_i is the unique stationary distribution vector for stochastic complement S_{ii}, i.e.,

$$\phi_i = \phi_i S_{ii}.$$

So, when the Markov chain is lumpable, i.e.,

$$P_{IJ} e = k_{IJ} e, \ I, \ J = 1, \ 2, \ \cdots, M,$$

then $c_{IJ} = \phi_i k_{IJ} e = k_{IJ}$ since $\phi_i e = 1$.

□

Therefore, the iterative aggregation disaggregation method can be simplified for the aggregation step. The IAD algorithm for lumpable Markov chains is illustrated as follows. Note that when the first execution of disaggregation is finished and more iterations are needed, the execution process goes to line 5 instead of line 2. That is, the aggregation step is executed once and for all.

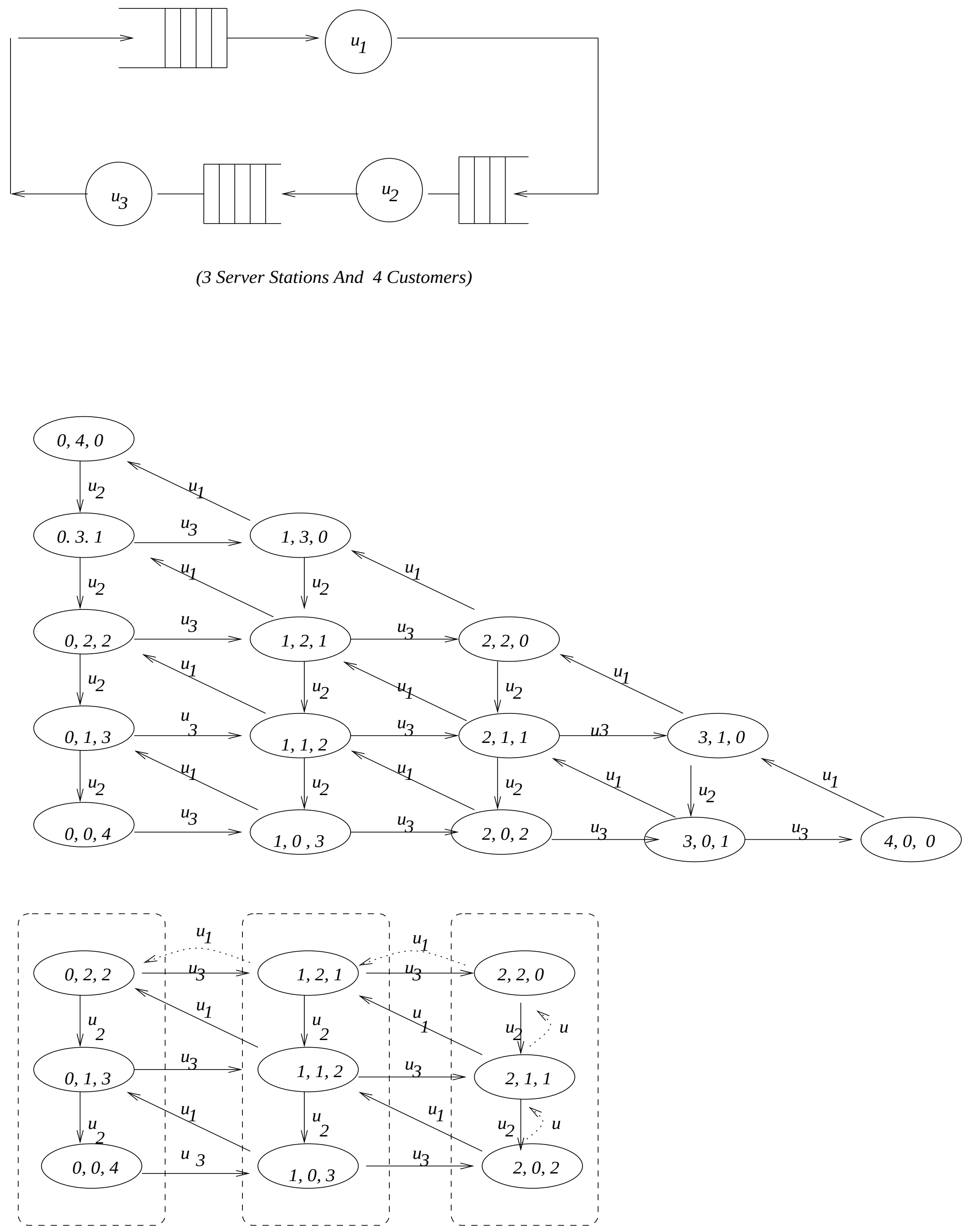


Figure 6.8: Closed Queueing Network And Its Transition Diagram

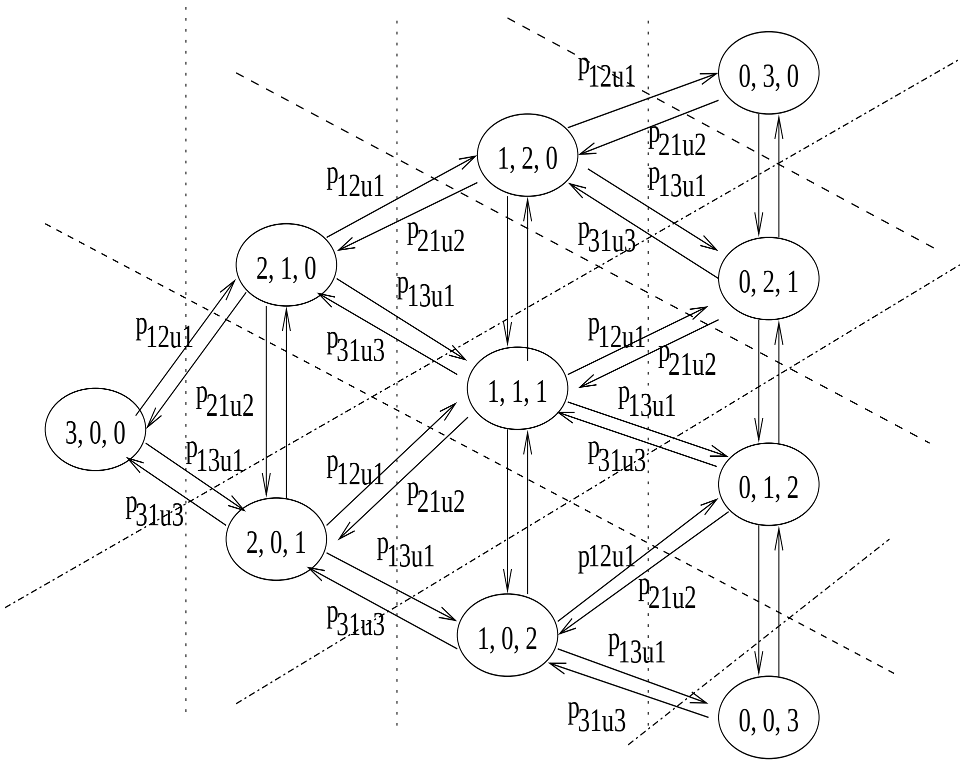


Figure 6.9: State Transition Diagram For Model In Figure 6.11

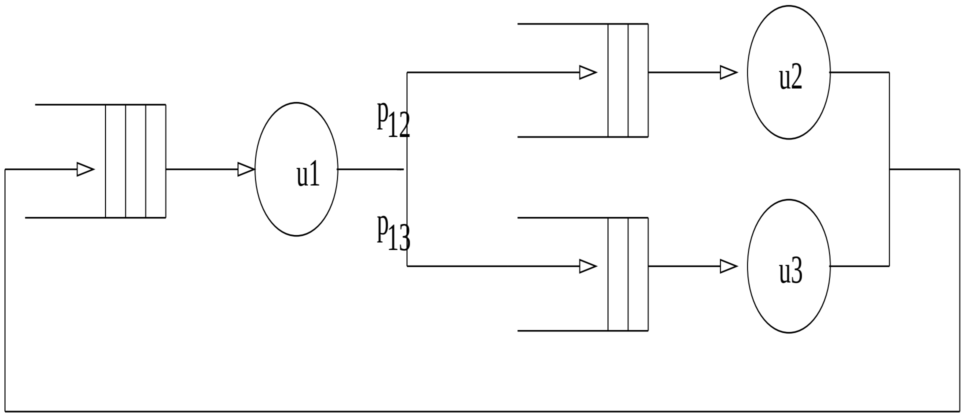


Figure 6.10: Closed Queueing Network With Parallel Queues(2)

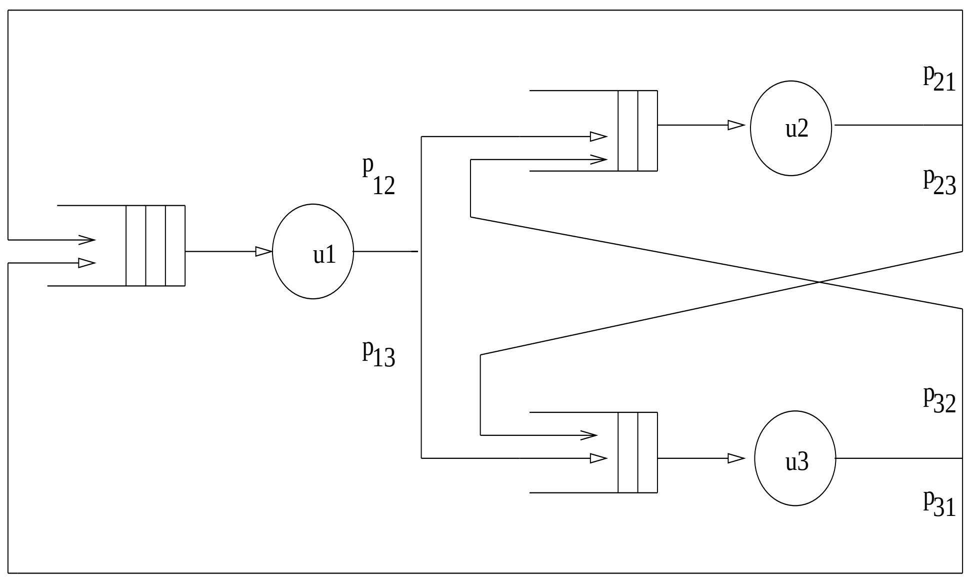


Figure 6.11: Closed Queueing Network With Parallel Queues(3)

Input:

$$P = \begin{pmatrix} P_{11} & P_{12} & \cdots & P_{1M} \\ P_{21} & P_{22} & \cdots & P_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ P_{M1} & P_{M2} & \cdots & P_{MM} \end{pmatrix},$$

Output: The steady state distribution $\pi = (\pi_1, \cdots, \pi_M)$.

```
1.  Initialization: $\pi^{(0)} = (\pi_1^{(0)}, \cdots, \pi_M^{(0)})$, set $m = 1$.
2.  Aggregation:
3.      Form the coupling matrix: $C[I,J] = K[I,J],\ I,J = 1,2,\cdots,M$.
4.      Solve the censored M.C.: $\xi = \xi C\ \xi e = 1,\ \xi > 0$.
5.  Disaggregation:
6.      Form the row vector: $z^{(m)} = (\xi_1 \frac{\pi_1^{(m-1)}}{||\pi_1^{(m-1)}||_1},\ \xi_2 \frac{\pi_2^{(m-1)}}{||\pi_2^{(m-1)}||_1},\ \cdots, \xi_M \frac{\pi_M^{(m-1)}}{||\pi_M^{(m-1)}||_1})$.
7.  **For** each $K = 1,2,\cdots,M$
8.      Compute $\pi_K^{(m)}$: $\pi_K^{(m)} = \pi_K^{(m)} P_{KK} + \sum_{J<K} \pi_J^{(m)} P_{JK} + \sum_{J>K} z_J^{(m)} P_{JK}$
9.  **End for**
10. Test for convergence:
11.     **If** not convergence, set $m = m + 1$, go to step 5.
12.     **Else** Stop.
```

Figure 6.12: IAD Algorithm For Lumpable Markov Chains

Chapter 7

Bounding Technique Based on Stochastic Comparison

7.1 Introduction

In some application areas for Markov Chains, it is sufficient to bound the performance measures of interest instead of having to calculate exact solutions. This can be seen in the performance analysis of computer networks which requires only a guaranteed Quality of Service (QoS): exact performance measures are not necessary. One bounding technique of current interest uses stochastic comparisons [67, 71]. There are different stochastic orderings on which a stochastic comparison method may be based. The best known and most widely applied, is that of *strong stochastic ordering*. Intuitively, a random variable X is more likely to assume larger values than another random variable Y if $Y \leq_{st} X$ when both random variables take values on a totally ordered space [20, 67, 26]. Observe that we shall use the notation $\leq_{st}$ to denote a strong stochastic ordering.

In order to obtain componentwise probability bounds of interest, the state space of the Markov chain must be ordered in such a way in which the state of interest, i.e., the state from which the actual measure of the quality of service is obtained, is the *last* state (details to be discussed in the following sections). In this paper, we observe that the bounding procedure works symmetrically when the state are ordered in such a way that the state of interest is placed in the *first* position instead of the *last*. In this thesis, a lower bounding stochastic matrix is constructed instead of an upper bounding matrix.

Several constructive algorithms based on strong stochastic comparison have been presented in the literature [1, 6, 25, 26]. These algorithms first construct a bounding matrix which is then modified in order to to render the numerical computation of a bounding stationary distribution vector more facile. Since it is the size of the Markov chain transition matrix that usually causes numerical difficulties, it is generally not beneficial if the computation of the stationary distribution vector for the upper bounding stochastic matrix is to be computed from a matrix that is the same order of magnitude as the original matrix. So, in order to simplify the numerical computation once the upper bounding stochastic matrix has been constructed, one possibility is to reduce the state space by lumping states into different groups to form a new *lumped* Markov chain having much fewer states [36]. States in the lumped chains are called *macro states* and each *macro state* state contains one or more (and generally, many) of the original states. The details will be discussed in the following section. In this dissertation, we consider the effect of the number of partitions on the tightness of the bounds obtained. We observe that the more partitions (lumped macro states) there are, the tighter the upper bounds tend to be.

Dayar and his co-authors [20] observe that different state orderings (yet nevertheless subject to the fact that the state of interest is the *last* state) result in different upper bounds on the probability of the last state and they present several heuristics to try to obtain the tightest possible upper bounds. One of these heuristics is to minimize the 1-norm of the last column of the perturbation matrix. In [21], they introduce a second approach to obtaining tight upper bounds by first transforming the transition probability matrix and then constructing its upper bounding stochastic monotone matrix.

Based on the permutation of the states and the transformation of the transition probability matrix, we pose the following questions: Is it beneficial to combine both the state permutation method and the probability matrix transformation method (i.e., permuting states first and then transforming the probability matrix)? Is the optimal state ordering still the best after the transformation of the transition probability matrix? If so, then why is it still consistent? With regard to the lumping method, is it advantageous to go one step further by partitioning the permuted and transformed probability matrix instead of simply partitioning the original probability matrix? In summary, we wish to investigate the effects of combining permutation, transformation and partition in order to obtain tight upper bounds for the states of interest.

7.2 Background Review

We briefly review stochastic comparisons based on strong stochastic orderings. The algorithmic approach to constructing upper bounding stochastic and monotone matrices is also reviewed. For the sake of simplicity, our focus is restricted to discrete time Markov chains (DTMC) on a finite state space, $S = \{1,\ 2,\ \cdots,\ n\}$ with transition probability matrices P of order $n \times n$. The approach is equally applicable to continuous time Markov chains (CTMC) once the chains have been uniformized. The goal is to obtain an upper bound for the stationary distribution vector π which satisfies the relation

$$\pi P = \pi \quad \text{and} \quad \sum_{i=1}^{n} \pi_i = 1.0.$$

We use the abbreviation "*st*" for *strong stochastic ordering*. We shall denote row i of the matrix P by $P_{i,*}$.

7.2.1 Strong Stochastic Ordering And Constructive Algorithms

Definition 7.2.1 *Let X and Y be random variables taking values on a totally ordered space. Then X is said to be less than Y in the strong stochastic sense, i.e., $X <_{st} Y$ iff $E[f(X)] \leq E[f(Y)]$ for all nondecreasing functions f whenever the expectations exist.*

Definition 7.2.2 *Let X and Y be random variables taking values on a finite state space*

$\{1, 2, \cdots, n\}$. Let p and q be probability distribution vectors such that

$$p_j = Prob(X = j) \text{ and } q_j = Prob(Y = j) \quad \text{for } j = 1, 2, \cdots, n.$$

Then X is said to be less than Y in the strong stochastic sense, i.e., $X <_{st} Y$ iff

$$\sum_{j=i}^{n} p_j \leq \sum_{j=i}^{n} q_j \text{ for } i = 1, 2, \cdots, n.$$

Definition 7.2.3 *Let P and Q be two stochastic matrices. Then $P \leq_{st} Q$ iff for all $i = 1, 2, \ldots, n$, we have $P_{i,*} <_{st} Q_{i,*}$.*

Definition 7.2.4 *Let P be a stochastic matrix. Then P is st-monotone iff for all $i, j > i$, we have $P_{i,*} <_{st} P_{j,*}$.*

Theorem 7.2.1 *Let $X(t)$ and $Y(t)$ be two DTMC and let P and Q be their respective stochastic matrices. Then $X(t) <_{st} Y(t)$ if*

- *$X(0) <_{st} Y(0)$,*
- *st-monotone of at least one of the matrices holds,*
- *st-comparability of the matrices holds, i.e., $P_{i,*} <_{st} Q_{i,*}$ for all i.*

Algorithm 1 below constructs an st-monotone upper bounding DTMC Q for a given DTMC P. This algorithm is optimal in the sense that, if there is another st-monotone upper bounding DTMC Q' for P, then $Q <_{st} Q'$. Note that bounds on the stationary probability distributions may still be improved. i.e., they may be not optimal.

Example 1 [20]: Consider the 4-state DTMC P below

$$P = \begin{pmatrix} 0.3 & 0.2 & 0.3 & 0.2 \\ 0.3 & 0.1 & 0.4 & 0.2 \\ 0.4 & 0.1 & 0.2 & 0.3 \\ 0.3 & 0.2 & 0.1 & 0.4 \end{pmatrix}$$

```
Algorithm 1    Constructs an optimal st-monotone upper bounding DTMC $Q$:
1. $q_{1,n} = p_{1,n}$
2. for $i = 2, 3, \cdots, n$ do
3.     $q_{i,n} = \max\{q_{i-1,n},\ p_{i,n}\}$
4. for $j = n-1, n-2, \cdots, 1$ do
5.     $q_{1,j} = p_{1,j}$
6.     for $i = 2, 3, \cdots, n$ do
7.         $q_{i,j} = \max\{\sum_{k=j}^{n} q_{i-1,k},\ \sum_{k=j}^{n} p_{i,k}\} - \sum_{k=j+1}^{n} q_{i,k}$
```

Figure 7.1: Algorithm to Construct st-monotone Upper Bounding Markov Chains

whose stationary distribution vector is (with 4 decimal digits of accuracy)

$$\pi_P = [0.3236\ 0.1603\ 0.2365\ 0.2796].$$

Applying algorithm 1, the st-monotone upper bounding DTMC Q is

$$Q = \begin{pmatrix} 0.3 & 0.2 & 0.3 & 0.2 \\ 0.3 & 0.1 & 0.4 & 0.2 \\ 0.3 & 0.1 & 0.3 & 0.3 \\ 0.3 & 0.1 & 0.2 & 0.4 \end{pmatrix}$$

whose stationary distribution vector is (with 4 decimal digits of accuracy)

$$\pi_Q = [0.3000\ 0.1300\ 0.2844\ 0.2856].$$

Other than in the case of matrices having a very special structure, such as bandedness, symmetry and so on, it serves little purpose if the new algorithm requires the computation of the stationary distribution vector of an upper bounding st-monotone matrix Q that is the same size as the original matrix P. In order to reduce the time needed to compute the stationary distribution vector, Fourneau and Pekergin [25] developed algorithms to construct upper-Hessenberg matrices, while Feinberg & Chiu employed a reduced matrix based on a *Single Input Macro State* approach. However, the most widely used approach is to reduce the size of the state space by lumping states into groups to form a new Markov chain having much fewer number of *macro* states (by lumping states into groups). We now present an algorithm to construct a lumpable st-monotone upper bounding matrix.

Theorem 7.2.2 *Let Q be an st-monotone matrix which is lumpable with respect to the partition $S = \{S_1, S_2, \cdots, S_M\}$ and which is an upper bound for P. Let Q_{IJ} and P_{IJ} be block transitions from set S_I to set S_J for Q and P respectively. Then for all $I, J,\ 1 \leq I, J \leq M$, block Q_{IJ} is st-monotone.*

The proof of this theorem may be found in [25]. The algorithm to construct a lumpable upper bounding st-monotone matrix is illustrated in Figure 7.2 where $b(I)$ and $e(I)$, $I = 1, 2, \ldots, M$ represent the first state and the last state respectively.

```
Algorithm 2    Constructs a lumpable st-monotone upper bounding DTMC $Q$:
1. $q_{1,n} = p_{1,n}$
2. for $J = M, M-1, \cdots, 1$ do
3.     for $l = e(J), e(J)-1, \cdots, b(J)$ do
4.         $q_{1l} = \sum_{j=l}^{n} p_{1j} - \sum_{j=l+1}^{n} q_{1j}$
5.         for $i = 2, 3, \cdots, n$ do
6.             $q_{il} = \max\{\sum_{j=l}^{n} q_{i-1,j}, \sum_{j=l}^{n} p_{i,j}\} - \sum_{j=l+1}^{n} q_{i,j}$
7.     for $I = 1, 2, \cdots, M$ do
8.         $c = \sum_{j=b(I)}^{e(I)} q_{e(I),j}$
9.         for $i = b(I), b(I)+1, \cdots, e(I)-1$ do
10.            $q_{i,b(I)} = c - \sum_{j=b(I)+1}^{e(I)} q_{i,j}$
```

Figure 7.2: Algorithm to Construct Lumpable st-monotone Upper Bounding Markov Chains

This algorithm constructs the matrix column by column in two steps. The first step is based on Algorithm 1 while the second makes modifications on the first column of the block to satisfy the lumpability constraint (6.2). This constraint requires that all blocks have equal row sums. Due to st-monotonicity, the maximal row sum is reached on the last row of the block. Therefore, after we construct the current block by use of Algorithm 1, we modify the first column of the current block with the now available maximum block row sum. That is, we simply increase each element in the first column of the block by the difference between its original row sum and the maximal row sum of the block (which is the last).

7.2.2 Permutation And Transformation

Let us consider the state permutation method and the transformation method proposed by Dayar and examine how we might combine these approaches to obtain tight upper bounds.

State Permutations

In [20], Dayar presented the approach of re-ordering the states of the Markov chain so as to find a tight probability bound for the last state, state n, by using a st-monotone upper bounding matrix Q on the original stochastic probability matrix P . A natural choice is to consider orderings that minimize the 1-norm of the last column of the perturbation matrix E, i.e., to minimize $|Ee_n|_1$ where $E = Q - P$ and e_n is the n^{th} column of the identity matrix. It is possible to view the 1-norm of the last column of E as the minimum total perturbation necessary to ensure that the last state is st-monotone.

For illustrative purposes, consider the 4-state Markov chain of example 1 and the results presented in the first five columns of Table 7.1. In this table, for each of the four possible states placed in the last position, we permute the other three and obtain $3! = 6$ different state orderings for each group. The best upper bound for each group of orderings is in bold fonts. Dayar noted that minimizing the 1-norm of the the last column of E may not always give the best upper bound but that a large 1-norm always results in a poor upper bound.

Matrix Transformation

In [21], Dayar et al. considered a linear transformation for stochastic matrices of the type

$$P \rightarrow \mathcal{T}(P, \delta) = (1 - \delta)P + \delta I, \ for \ \delta \in (0, \ 1). \tag{7.1}$$

This transformation does not affect the steady-state distribution [21, 64] but it does have a large influence on the effect of Algorithm 1. It was shown that if the given stochastic matrix is *not* row diagonally dominant (RDD), then the steady-state probability distribution of the optimal st-monotone upper bounding matrix corresponding to the row diagonally dominant transformed matrix is better in the strong stochastic sense than the one corresponding to the original matrix. The transformation $\frac{1}{2}P+\frac{1}{2}I$ provides the best bound for the family of linear transformation considered. In what follows, we shall let $\nu(P)$ denote the st-monotone upper bounding matrix for the original stochastic matrix P. Recall that the stochastic matrix P is said to be row diagonally dominant (RDD) if all of its diagonal elements are greater than or equal to 0.5. Furthermore, observe that it suffices to choose $\delta = 1/2$ in (7.1) to transform an arbitrary stochastic matrix into one that is RDD.

Theorem 7.2.3 *Let P be a DTMC of order n that is* not *RDD and let $\delta_1, \delta_2 \in (0,\ 1)$ be two parameters such that $\delta_1 < \delta_2$. Then*

$$\pi_P <_{st} \pi_{\nu(\mathcal{T}(P,\delta_1))} <_{st} \pi_{\nu(\mathcal{T}(P,\delta_2))} <_{st} \pi_{\nu(P)}.$$

Dayar et al. show that when the DTMC is already RDD, no benefit arises from choosing values of δ that are less than one half. For a RDD DTMC, smaller values of δ do not result in tighter upper bounds. Since $\delta = 1/2$ is sufficient to transform an arbitrary stochastic matrix into one that is RDD, the optimal linear transformation (of degree 1) is $\mathcal{T}(P, \frac{1}{2}) = \frac{1}{2}P + \frac{1}{2}I$. This approach may be generalized to transformations based on a set of higher-degree polynomials which may give better upper bounds.

Definition 7.2.5 *Let $\mathcal{D}$ be the set of polynomials $\phi(\cdot)$ such that $\phi(1) = 1$, different from the identity and with non-negative coefficients.*

Theorem 7.2.4 *Let ϕ be an arbitrary polynomial in $\mathcal{D}$. Then Algorithm 1 applied to $\phi(P)$ provides a more accurate bound than that of Q, i.e.,*

$$\pi_P <_{st} \pi_{\nu(\phi(P))} <_{st} \pi_{\nu(P)}.$$

The proof may be found in [21]. Furthermore, in [9] it is claimed that the larger the degree of ϕ, the more accurate the bound $\nu(\phi(P))$. However, the optimal transformation when the degree is greater than 1, is still an open problem. In other words, it is still not known how to determine the best coefficients for the polynomials of degree greater than 1.

Example 2 Let us re-examine the matrix P of Example 1, this time applying the transformations $\phi_1(X) = \frac{1}{2}X + \frac{1}{2}$ and $\phi_2(X) = \frac{1}{2}X^2 + \frac{1}{2}$. We have

$$\phi_1(P) = \begin{pmatrix} 0.6500 & 0.1000 & 0.1500 & 0.1000 \\ 0.1500 & 0.5500 & 0.2000 & 0.1000 \\ 0.1500 & 0.1000 & 0.6000 & 0.1500 \\ 0.1500 & 0.1000 & 0.0500 & 0.7000 \end{pmatrix}.$$

The corresponding st-monotone upper bounding matrix obtained by use of Algorithm 1 is

$$\nu(\phi_1(P)) = \begin{pmatrix} 0.6500 & 0.1000 & 0.1500 & 0.1000 \\ 0.1500 & 0.5500 & 0.2000 & 0.1000 \\ 0.1500 & 0.1000 & 0.6000 & 0.1500 \\ 0.1500 & 0.1000 & 0.0500 & 0.7000 \end{pmatrix}.$$

Also,

$$\phi_2(P) = \begin{pmatrix} 0.6650 & 0.0750 & 0.1250 & 0.1350 \\ 0.1700 & 0.5750 & 0.1150 & 0.1400 \\ 0.1600 & 0.0850 & 0.6150 & 0.1400 \\ 0.1550 & 0.0850 & 0.1150 & 0.6450 \end{pmatrix}$$

and the corresponding st-monotone upper bounding matrix obtained from Algorithm 1 is

$$\nu(\phi_2(P)) = \begin{pmatrix} 0.6650 & 0.0750 & 0.1250 & 0.1350 \\ 0.1700 & 0.5700 & 0.1200 & 0.1400 \\ 0.1600 & 0.0850 & 0.6150 & 0.1400 \\ 0.1550 & 0.0850 & 0.1150 & 0.6450 \end{pmatrix}.$$

Calculating all steady-state distributions and arranging them side-by-side for ready comparison, we have

$$\begin{aligned}
\pi_P &= [0.3236\ 0.1603\ 0.2365\ 0.2796],\\
\pi_{\nu(P)} &= [0.3000\ 0.1300\ 0.2844\ 0.2856],\\
\pi_{\nu(\phi_1(P))} &= [0.3000\ 0.1818\ 0.2384\ 0.2798],\\
\pi_{\nu(\phi_2(P))} &= [0.3236\ 0.1588\ 0.2381\ 0.2796].
\end{aligned}$$

Index	*Ordering*	π_P	$\pi_{\nu(P)}$	$\lVert Ee_n \rVert_1$	$\pi_{\nu(\phi_1(P))}$	$\lVert Ee_n \rVert_1$	$\pi_{\nu(\phi_2(P))}$	$\lVert Ee_n \rVert_1$
1	{1, 2, 3, 4}	0.2796	**0.2856**	0.0	**0.2798**	0.00	**0.2796**	0.000
2	{1, 3, 2, 4}	0.2796	0.3000	0.1	0.2974	0.05	0.22796	0.000
3	{2, 1, 3, 4}	0.2796	0.2889	0.0	0.2828	0.00	0.2828	0.005
4	{2, 3, 1, 4}	0.2796	0.3222	0.1	0.3222	0.05	0.2828	0.005
5	{3, 1, 2, 4}	0.2796	0.3333	0.2	0.3333	0.10	0.2828	0.005
6	{3, 2, 1, 4}	0.2796	0.3333	0.1	0.3333	0.10	0.2828	0.005
7	{1, 2, 4, 3}	0.2365	0.3700	0.5	0.3083	0.15	0.2451	0.020
8	{1, 4, 2, 3}	0.2365	0.3500	0.4	0.2917	0.10	0.2451	0.020
9	{2, 1, 4, 3}	0.2365	0.4000	0.6	0.3333	0.20	0.2422	0.010
10	{2, 4, 1, 3}	0.2365	0.4000	0.6	0.3333	0.20	0.2368	0.000
11	{4, 1, 2, 3}	0.2365	**0.2950**	0.2	**0.2458**	0.00	0.2397	0.010
12	{4, 2, 1, 3}	0.2365	0.3250	0.3	0.2708	0.05	**0.2366**	0.000
13	{1, 3, 4, 2}	0.1603	0.2000	0.2	0.1818	0.05	**0.1604**	0.000
14	{1, 4, 3, 2}	0.1603	0.2000	0.2	0.1818	0.05	0.1605	0.000
15	{3, 1, 4, 2}	0.1603	**0.1850**	0.1	**0.1653**	0.00	0.1667	0.010
16	{3, 4, 1, 2}	0.1603	0.1889	0.1	0.1717	0.00	0.1667	0.010
17	{4, 1, 3, 2}	0.1603	0.2000	0.2	0.1818	0.05	0.1667	0.010
18	{4, 3, 1, 2}	0.1603	0.2000	0.2	0.1818	0.05	0.1667	0.010
19	{2, 3, 4, 1}	0.3236	0.3900	0.2	0.3545	0.05	0.3366	0.025
20	{2, 4, 3, 1}	0.3236	0.3700	0.1	0.3306	0.00	0.3366	0.025
21	{3, 2, 4, 1}	0.3236	0.4000	0.3	0.3636	0.10	0.3321	0.015
22	{3, 4, 2, 1}	0.3236	0.4000	0.3	0.3636	0.10	0.3265	0.005
23	{4, 2, 3, 1}	0.3236	**0.3625**	0.1	**0.3244**	0.00	0.3283	0.010
24	{4, 3, 2, 1}	0.3236	0.3722	0.2	0.3382	0.05	**0.3237**	0.000

Table 7.1: Bounding Effects on a 4−state Discrete Time Markov Chain(1)

7.3 Combine Permutation, Transformation and Partitioning

In this section, we provide some new observations and results. First of all, we observe that the lower bounding stochastic matrix can give us the same upper bounding for the *first* state as the upper bounding stochastic matrix does for the *last* state. Secondly, we observed that the more lumpable partitions we make, the more accurate the upper bound will become. Thereafter, we investigated the effects on upper bounds by combining state permutation and matrix transformations. Finally, we combine all together, i.e., put state permutation, matrix transformation and lumpable partitionings together to improve the quality of the upper bounds for the state of interest.

7.3.1 Upper Bounds For The First State

Based on definition 7.2.5 of the previous section, we have

$$p_1 \geq q_1$$

since

$$\sum_{j=2}^{n} p_j \leq \sum_{j=2}^{n} q_j$$

and

$$p_1 + \sum_{j=2}^{n} p_j = q_1 + \sum_{j=2}^{n} q_j = 1.0.$$

So, if we construct a st-monotone lower bounding matrix Q, then the *first* state of interest is bounded from above by $\pi_Q(1)$. The algorithm to construct a st-monotone *lower* bounding matrix Q for a given stochastic matrix P is as follows.

The motivation for constructing a st-monotone lower bounding matrix to obtain an upper bound on the probability distribution for the first state is that the errors for state $2, 3, \ldots,$ n may cancel each other out. Perhaps this could lead to a tighter upper bound on state 1. We carried out some experiments by applying Algorithm 3 on the matrix P of Example

```
Algorithm 3    Construct an optimal st-monotone lower bounding DTMC $Q$:
1. for $j = 1, 2, \cdots, n$ do
2      $q_{n,j} = p_{n,j}$
3. for $i = n-1, n-2, \cdots, 1$ do
4.       $q_{i,n} = \min\{q_{i+1,n},\ p_{i,n}\}$
5. for $i = n-1, n-2, \cdots, 1$ do
6.       for $j = n-1, n-2, \cdots, 1$ do
7.            $q_{i,j} = \min\{\sum_{k=j}^{n} q_{i+1,k},\ \sum_{k=j}^{n} p_{i,k}\} - \sum_{k=j+1}^{n} q_{i,k}$
```

Figure 7.3: Algorithm to Construct st-monotone Lower Bounding Markov Chains

1. This matrix P and its lower bounding matrix Q, based on the ordering $\{4, 1, 2, 3\}$, are shown below.

$$P = \begin{pmatrix} 0.4000 & 0.3000 & 0.2000 & 0.1000 \\ 0.2000 & 0.3000 & 0.2000 & 0.3000 \\ 0.2000 & 0.3000 & 0.1000 & 0.4000 \\ 0.3000 & 0.4000 & 0.1000 & 0.2000 \end{pmatrix} \ and\ Q = \begin{pmatrix} 0.4000 & 0.3000 & 0.2000 & 0.1000 \\ 0.3000 & 0.4000 & 0.1000 & 0.2000 \\ 0.3000 & 0.4000 & 0.1000 & 0.2000 \\ 0.3000 & 0.4000 & 0.1000 & 0.2000 \end{pmatrix}.$$

In order to examine the effects of the transformations when the first state is of interest, we include the numerical results for them as well in Table 7.4.

Observe that the lower bounding matrix gives exactly the same upper bound for the first state which means that the errors for state 2, 3 and 4 have not cancelled out. The upper bounding matrix and lower bounding matrix have symmetric behavior. For example, the ordering $\{1, 2, 3, 4\}$ in Table 7.4 (lower bounding case) has all upper bounds and 1-norm of the first column of the perturbation matrix the same as the ordering $\{4, 3, 2, 1\}$ in the Table 7.1 (upper bounding case).

7.3.2 Bounding Accuracy For Lumpable Partitionings

It is natural to assume that the greater the number of lumpable partitions, the more accurate the bounds obtained. We observed this phenomena on a number of DTMCs randomly generated by Matlab. We first considered the following suite of three 8×8 randomly generated DTMCs.

$$P_1 = \begin{pmatrix} 0.1974 & 0.1707 & 0.1944 & 0.0289 & 0.0925 & 0.1742 & 0.0633 & 0.0786 \\ 0.0609 & 0.1171 & 0.2415 & 0.0534 & 0.2454 & 0.0052 & 0.0500 & 0.2265 \\ 0.1507 & 0.1529 & 0.1019 & 0.0494 & 0.1158 & 0.1692 & 0.0480 & 0.2121 \\ 0.1002 & 0.1633 & 0.1843 & 0.1245 & 0.0863 & 0.0783 & 0.1407 & 0.1224 \\ 0.1929 & 0.1995 & 0.0125 & 0.0589 & 0.1831 & 0.1800 & 0.0655 & 0.1075 \\ 0.1686 & 0.1633 & 0.0780 & 0.0440 & 0.1161 & 0.1112 & 0.1198 & 0.1990 \\ 0.1364 & 0.0527 & 0.2430 & 0.0046 & 0.0606 & 0.2120 & 0.0451 & 0.2456 \\ 0.0051 & 0.1119 & 0.0027 & 0.2060 & 0.1854 & 0.1183 & 0.1925 & 0.1779 \end{pmatrix},$$

$$P_2 = \begin{pmatrix} 0.0024 & 0.1788 & 0.0820 & 0.2048 & 0.1197 & 0.1626 & 0.0144 & 0.2353 \\ 0.1271 & 0.0587 & 0.1985 & 0.1125 & 0.0563 & 0.1106 & 0.1091 & 0.2272 \\ 0.1637 & 0.1965 & 0.0851 & 0.0923 & 0.1522 & 0.0304 & 0.1191 & 0.1607 \\ 0.1644 & 0.1179 & 0.1299 & 0.1232 & 0.2035 & 0.1174 & 0.0542 & 0.0894 \\ 0.1620 & 0.1113 & 0.1231 & 0.0996 & 0.1421 & 0.1450 & 0.1052 & 0.1117 \\ 0.1910 & 0.0530 & 0.0898 & 0.1620 & 0.1591 & 0.1524 & 0.0111 & 0.1816 \\ 0.0924 & 0.1379 & 0.1039 & 0.1284 & 0.1218 & 0.2511 & 0.1435 & 0.0211 \\ 0.1088 & 0.1632 & 0.1148 & 0.1361 & 0.0201 & 0.1685 & 0.1228 & 0.1657 \end{pmatrix}$$

and

$$P_3 = \begin{pmatrix} 0.1458 & 0.1575 & 0.1763 & 0.0193 & 0.1326 & 0.2568 & 0.0407 & 0.0709 \\ 0.0822 & 0.2221 & 0.2953 & 0.0238 & 0.0250 & 0.1693 & 0.0411 & 0.1412 \\ 0.2612 & 0.0583 & 0.0041 & 0.0038 & 0.0614 & 0.3012 & 0.2424 & 0.0677 \\ 0.0774 & 0.1871 & 0.0912 & 0.0668 & 0.1116 & 0.2434 & 0.2121 & 0.0105 \\ 0.1818 & 0.0411 & 0.1879 & 0.1245 & 0.0667 & 0.2213 & 0.1572 & 0.0196 \\ 0.1481 & 0.1212 & 0.0690 & 0.1029 & 0.1731 & 0.0254 & 0.1693 & 0.1910 \\ 0.0466 & 0.1356 & 0.2016 & 0.1530 & 0.0683 & 0.1767 & 0.1443 & 0.0740 \\ 0.1106 & 0.1260 & 0.1246 & 0.1070 & 0.1172 & 0.1668 & 0.1622 & 0.0856 \end{pmatrix}.$$

The numerical results obtained are presented in Table 7.2. In these experiments, the order of states used was the natural ordering, $\{1, 2, 3, 4, 5, 6, 7, 8\}$. The partitioning schemes are as follows.

$$\begin{aligned} &\textit{partition } a: && \{\{1,2,3,4,5,6,7\},\{8\}\} \\ &\textit{partition } b: && \{\{1,2,3,4\},\{5,6,7\},\{8\}\} \\ &\textit{partition } c: && \{\{1,2\},\{3,4\},\{5,6\},\{7\},\{8\}\} \\ &\textit{partition } d: && \{\{1\},\{2\},\{3\},\{4\},\{5\},\{6\},\{7\},\{8\}\} \end{aligned}$$

Algorithm 2 was applied to the matrices P_i, $i = 1, 2, 3$ based on different lumpable partitioning schemes defined above. It can be seen that the more partitionings, the tighter the upper bound obtained for state 8.

Partitioning Schemes	*exact*	$\pi_{\nu(P_1)}(8)$	*exact*	$\pi_{\nu(P_2)}(8)$	*exact*	$\pi_{\nu(P_3)}(8)$
partition a	0.1714	0.2456	0.15555	0.2352	0.0941	0.1909
partition b	0.1714	0.2376	0.15555	0.2312	0.0941	0.1746
partition c	0.1714	0.2325	0.15555	0.2297	0.0941	0.1726
partition d	0.1714	0.2239	0.15555	0.2251	0.0941	0.1634

Table 7.2: Bounding Accuracy For Lumpable Partitionings(1)

A second suite of two matrices, P_4 and P_5 of sizes 16×16 and P_6 of 24×24 respectively, were also generated randomly by Matlab, but are omitted in order to save space. For these matrices, the state partitionings chosen are as follows.

$$\begin{array}{ll}
partition1: & \{\{1,2,3,4,5,6,7,8\},\{9,10,11,12,13,14,15\},\{16\}\} \\
partition2: & \{1,2,3,4\},\{5,6,7,8\},\{9,10,11,12\},\{13,14,15\},\{16\}\} \\
partition3: & \{\{1,2\},\{3,4\},\{5,6\},\{7,8\},\{9,10\},\{11,12\},\{13,14\},\{15\},\{16\}\} \\
partition4: & \{\{1\},\{2\},\{3\},\{4\},\{5\},\{6\},\{7\},\{8\},\{9\},\{10\},\{11\},\{12\},\{13\},\{14\},\{15\},\{16\}\}
\end{array}$$

Notice that the partitions for matrix P_6 of size 24×24 are different yet similar to those for matrix P_4 and P_5. The results of Table 7.3 again show that increasing the number of partitionings results in tighter upper bounds for the last state.

Lumpable Partitionings	*exact*	$\pi_{\nu(P_4)}(16)$	*exact*	$\pi_{\nu(P_5)}(16)$	*exact*	$\pi_{\nu(P_6)}(24)$
partition 1	0.0588	0.1250	0.0751	0.1336	0.0388	0.0940
partition 2	0.0588	0.1219	0.0751	0.1335	0.0388	0.0940
partition 3	0.0588	0.1219	0.0751	0.1235	0.0388	0.0908
partition 4	0.0588	0.1201	0.0751	0.1182	0.0388	0.0900
partition 5					0.0388	0.0889

Table 7.3: Bounding Accuracy For Lumpable Partitionings(2)

7.3.3 Combining State Permutation And Matrix Transformation

Let us now consider the effects of first transforming the original matrix P *before* reordering the states. In Table 7.1, we present the results obtained with all possible orderings by use of permutations.

In that table, the matrices E in columns 5, 7 and 9 are given by $E = \nu(P) - P$, $E = \nu(\phi_1(P)) - \phi_1(P)$ and $E = \nu(\phi_2(P)) - \phi_2(P)$ respectively. Observe that both transformed matrices give much better upper bounds. Also, the linear transformation is consistent with the st-monotone upper bounds for the original matrix P, i.e., they reach their best bound with the same ordering in each group of permutations. For example, the state

Index	*Ordering*	π_P	$\pi_{\nu(P)}$	$\|\|Ee_1\|\|_1$	$\pi_{\nu(\phi_1(P))}$	$\|\|Ee_1\|\|_1$	$\pi_{\nu(\phi_2(P))}$	$\|\|Ee_1\|\|_1$
1	{4, 1, 2, 3}	0.2796	0.3333	0.2	0.3333	0.10	0.2828	0.005
2	{4, 1, 3, 2}	0.2796	0.3222	0.1	0.3222	0.05	0.2828	0.005
3	{4, 2, 1, 3}	0.2796	0.3333	0.2	0.3333	0.10	0.2828	0.005
4	{4, 2, 3, 1}	0.2796	0.3000	0.1	0.2974	0.05	0.2796	0.000
5	{4, 3, 1, 2}	0.2796	0.2889	0.0	0.2828	0.00	0.2828	0.005
6	{4, 3, 2, 1}	0.2796	**0.2856**	0.0	**0.2798**	0.00	**0.2796**	0.000
7	{3, 1, 2, 4}	0.2365	0.3250	0.3	0.2708	0.05	**0.2366**	0.000
8	{3, 1, 4, 2}	0.2365	0.4000	0.6	0.3333	0.20	0.2368	0.000
9	{3, 2, 1, 4}	0.2365	**0.2950**	0.2	**0.2458**	0.00	0.2397	0.010
10	{3, 2, 4, 1}	0.2365	0.3500	0.4	0.2917	0.10	0.2451	0.020
11	{3, 4, 1, 2}	0.2365	0.4000	0.6	0.3333	0.20	0.2422	0.010
12	{3, 4, 2, 1}	0.2365	0.3700	0.5	0.3083	0.15	0.2451	0.020
13	{2, 1, 3, 4}	0.1603	0.2000	0.2	0.1818	0.05	0.1667	0.010
14	{2, 1, 4, 3}	0.1603	0.1889	0.1	0.1717	0.00	0.1667	0.010
15	{2, 3, 1, 4}	0.1603	0.2000	0.2	0.1818	0.05	0.1667	0.010
16	{2, 3, 4, 1}	0.1603	0.2000	0.2	0.1818	0.05	0.1605	0.000
17	{2, 4, 1, 3}	0.1603	**0.1850**	0.1	**0.1653**	0.00	0.1667	0.010
18	{2, 4, 3, 1}	0.1603	0.2000	0.2	0.1818	0.05	**0.1604**	0.000
19	{1, 2, 3, 4}	0.3236	0.3722	0.2	0.3382	0.05	**0.3237**	0.000
20	{1, 2, 4, 3}	0.3236	0.4000	0.3	0.3382	0.05	0.3265	0.005
21	{1, 3, 2, 4}	0.3236	**0.3625**	0.1	**0.3244**	0.00	0.3283	0.010
22	{1, 3, 4, 2}	0.3236	0.3700	0.1	0.3306	0.00	0.3366	0.025
23	{1, 4, 2, 3}	0.3236	0.4000	0.3	0.3636	0.10	0.3321	0.015
24	{1, 4, 3, 2}	0.3236	0.3900	0.2	0.3545	0.05	0.3366	0.025

Table 7.4: Bounding Effects on a 4−state Discrete Time Markov Chain(2)

ordering $\{1, 2, 3, 4\}$ gives the best bound for both the original matrix and the linearly transformed matrix among those orderings which has state 4 as the last state. However, this is not the case (at least for this example) for the transformation $\phi_2(\cdot)$. Observe also that both transformations have their best bounds when the 1-norm of the last column of the perturbation matrix is minimized. Finally, the order of applying permutation and transformation makes no difference. This can be seen in the following property.

Proposition 7.3.1 *Let P be a DTMC of order n. Let $\mathcal{M}$ and $\mathcal{T}$ be the operation of permutation and transformation on P, i.e.,*

$$\mathcal{M}(P) = SPS^T \quad \text{and} \quad \mathcal{T}(P) = \delta P + (1 - \delta)I$$

where S is a permutation matrix, δ is a scalar between 0 and 1 and I is an identity matrix. Then

$$\mathcal{T}\mathcal{M} = \mathcal{M}\mathcal{T}.$$

for any DTMC, P.

Proof Since

$$\mathcal{M}\mathcal{T}(P) = S[\delta P + (1-\delta)I]S^T = \delta SPS^T + (1-\delta)SIS^T = \delta SPS^T + (1-\delta)I$$

and

$$\mathcal{T}\mathcal{M}(P) = \mathcal{T}(SPS^T) = \delta(SPS^T) + (1-\delta)I$$

it follows that $\mathcal{M}\mathcal{T}(P) = \mathcal{T}\mathcal{M}(P)$ which completes the proof. $\square$

Note that this proposition continues to hold when the transformation $\mathcal{T}$ is any polynomial of degree greater than 1. Let us take another example.

Example 3 (Courtois) Consider a DTMC whose stochastic transition probability matrix P is as follows.

$$P = \begin{pmatrix} 0.8500 & 0.0000 & 0.1490 & 0.0009 & 0.0000 & 0.0001 & 0.0000 & 0.0001 \\ 0.1000 & 0.6500 & 0.2490 & 0.0000 & 0.0009 & 0.0001 & 0.0000 & 0.0001 \\ 0.1000 & 0.8000 & 0.0996 & 0.0003 & 0.0000 & 0.0000 & 0.0001 & 0.0000 \\ 0.0000 & 0.0004 & 0.0000 & 0.7000 & 0.2995 & 0.0000 & 0.0001 & 0.0000 \\ 0.0005 & 0.0000 & 0.0004 & 0.3990 & 0.6000 & 0.0001 & 0.0000 & 0.0000 \\ 0.0000 & 0.0001 & 0.0000 & 0.0000 & 0.0001 & 0.6000 & 0.2499 & 0.1500 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.1000 & 0.8000 & 0.0999 \\ 0.0000 & 0.0001 & 0.0000 & 0.0000 & 0.0001 & 0.1999 & 0.2500 & 0.5500 \end{pmatrix}$$

The steady-state distribution of this DTMC is

$$\pi_P = [0.0893\ 0.0928\ 0.0405\ 0.1585\ 0.1189\ 0.1204\ 0.2778\ 0.1018]$$

Index	*Ordering*	π_P	$\pi_{\nu(P)}$	$\pi_{\nu(\phi_1(P))}$	$\pi_{\nu(\phi_2(P))}$
1	{6, 8, 7, 4, 5, 2, 3, 1}	0.0893	0.1076	0.1062	0.0966
2	{6, 8, 7, 4, 5, 1, 3, 2}	0.0928	0.1281	0.1114	0.0998
3	{6, 8, 7, 4, 5, 1, 2, 3}	0.0405	0.0471	**0.0405**	**0.0405**
4	{7, 8, 6, 3, 2, 1, 5, 4}	0.1585	0.1626	0.1586	**0.1585**
5	{7, 8, 6, 3, 1, 2, 4, 5}	0.1189	0.1415	0.1401	0.1323

Table 7.5: Bounding Effects on a 8×8 Courtois Matrix

The experimental results for this example are presented in Table 7.5. Note that in Table 7.5, we tested only those orderings which give the best upper bounds obtained through the use of permutations only. Based on Tables 7.1 and 7.5, it appears that combinations of transformation and permutation results in better bounds for the last state, the state of interest.

7.3.4 Combining All Together

We now combine all three possibilities. That is, we first permute the states so that the best state ordering is obtained. Then the linear transformation on the permuted matrix is incorporated so that the upper bound becomes tighter. Finally, the matrix obtained after these first two steps is partitioned and lumped and the stationary distribution of this lumped matrix computed. We seek to answer questions as to whether this combined approach gives more accurate upper bounds for the state of interest; does it maintain the optimal state ordering? Are there still no differences when the order of permutation and transformation are interchanged?

With these questions in mind, we experimented on the three 8×8 matrices P_1, P_2 and P_3 by combining permutation, transformation and lumpable partitioning. The results obtained are shown in Table 7.6. In order to save space for the table, we use the same re-named partitioning schems as those in Table 7.2. Then for each particular partitioning scheme of each matrix, we permute states $1, 2, 3, 4, 5, 6, 7$ (a total of $7! = 5040$ orderings) and find the optimal state ordering which is shown in column 4. We then combine permutation and transformation and find the optimal state ordering which is shown in the last column. We included also the 1-norm of the last column of the perturbation matrix.

Partitioning	*No Transformation*			*Permute and Transform*		
P_1	$\pi(8)$	$\|\|Ee_8\|\|_1$	*opt. ordering*	$\pi(8)$	$\|\|Ee_8\|\|_1$	*opt. ordering*
a	0.2456	0.5952	{65173428}	0.2300	0.2638	{23671548}
b	0.2282	0.4088	{15642378}	0.2137	0.1706	{16453728}
c	0.2017	0.2138	{15362478}	0.1888	0.0731	{51364278}
d	0.1917	0.0677	{15463278}	0.1786	0.0000	{15463278}
P_2	$\pi(8)$	$\|\|Ee_8\|\|_1$	*opt. ordering*	$\pi(8)$	$\|\|Ee_8\|\|_1$	*opt. ordering*
a	0.2253	0.6897	{31465728}	0.2199	0.3101	{76321458}
b	0.2033	0.3913	{73542618}	0.1901	0.1609	{47351268}
c	0.1883	0.2325	{74356218}	0.1753	0.0815	{47352618}
d	0.1745	0.0696	{74536218}	0.1607	0.0000	{74536218}
P_3	$\pi(8)$	$\|\|Ee_8\|\|_1$	*opt. ordering*	$\pi(8)$	$\|\|Ee_8\|\|_1$	*opt. ordering*
a	0.1909	0.8675	{25617438}	0.1727	0.3811	{71523648}
b	0.1381	0.3995	{31752468}	0.1249	0.1471	{73516248}
c	0.1280	0.2937	{13754268}	0.1124	0.0909	{35172468}
d	0.1195	0.1239	{54731268}	0.1058	0.0092	{54731268}

Table 7.6: Bounding Effects on Combining Permutation, Transformation And Lumpable Partitioning

Among the many observations that might be made concerning the Table 7.6, we highlight the following.

- Once again, these results illustrate that the more partitionings, the tighter the upper bound for the state of interest, both with and without the linear transformation being performed on the matrix.

- The 1-norm of the last column of the perturbation matrix decreases as the number of partitions increases.

- The combination of permutation and transformation gives a tighter upper bound on the state of interest than that obtained by the use of permutation only.

- Although we did not include it into this table, we found that, given a particular partitioning scheme, the order in which permutation and transformation were carried out, made no difference whatsoever, i.e., applying the permutation first and then transforming a DTMC gives the exactly the same (tight) upper bound as that obtained by first applying the transformation and secondly, the permutation.

- Given a particular partitioning scheme, the optimal state ordering obtained when no transformation is applied, is different from that obtained when a transformation is applied. This may be seen by comparing columns 4 and column 7 of the table..

7.4 Conclusions

In this chapter, we investigated several issues concerning stochastic comparison based bounding methods for Markovian performance analysis. We observed that the lower bounding st-monotone matrix for the *first* state can give the same upper bounds as that obtained by upper bounding st-monotone matrix for the *last* state. We experimented with suites of randomly generated DTMCs and saw that the more lumpable partitions, the tighter the upper bounds for the state of interest. Finally we presented an approach to combining state permutation, matrix transformation and state partitioning in order to obtain a high-quality upper bounds in less computation time. These elementary experimental results indicate that the approach is efficient in terms of the accuracy of the upper bound for the state of interest.

Chapter 8

Summary and Future Research

We have presented a analysis of the convergence behavior of Markovian iteration matrices. That is, for an irreducible and homogeneous Markov chain, a necessary and sufficient condition for convergence in one iteration is that the iteration matrix have rank one. We considered also spectrum condition which is shown to be insufficient for convergence in a single iteration.

Most Markov iteration matrices, however, have rank greater than 1, so we use Wedderburn's rank-1 reduction formula to design a rank reduction algorithm with an initial iteration matrix having rank greater than one and modify it in successive steps, under the constraint that the exact solution be preserved at each step, until a rank-1 iteration matrix is obtained.

Then we investigate the relationship among lumpability, weak lumpability, quasi-lumpability and near complete decomposability. These concepts are important in the analysis of aggregating and disaggregating Markov chains. We improve White's algorithm for identifying all possible lumpable partitions of Markov chains. This is achieved by incorporating tests on special state orderings so that not all state orderings must be examined.

It is common in real situations, that the exact stationary distributions of MCs are not necessary. So instead of computing exact stationary distributions, we design stochastic-ordering-based techniques to bound them. We obtain upper bounds by using newly developed constructive algorithms. We find that the more lumpable partitionings, the more accurate the upper bound for the state of interest would be. This always holds whether matrix transformation is applied or not. In order to improve the quality of the upper bound for the state of interest, we combine the approaches of state permutation, matrix transformation and state partitioning and observe that the quality of the upper bounds is indeed much improved.

We recognize however that much more extensive testing needs to be done to verify the rank reduction's success and bounding technique's efficiency. A possible future work plan is listed as follows.

- Apply the rank reduction method to more realistic MCs and compare it with other numerical methods.

- Investigate the possibility of applying a block rank reduction method. That is, in each step, k (greater than 1) ranks are reduced instead of a single one.

- Some better heuristics/thumb rules are needed to improve White's algorithm. We find it still time-consuming to identify all lumpable partitions and consequently the benefits of doing aggregation once and for all is offset by the time needed in constructing the lumpable partitions.

- Conduct more experiments on bounding techniques. Tests on a variety of large, sparse MC matrices from different disciplines would make the bounding technique more reliable.

References

[1] O. Abu-Amsha, J.M. Vincent. An Algorithm to Bound Functionals of Markov Chains with Large State Space. *The* 4th *INFORMS Conference on Telecommunications, Boca Raton, Florida, 1998.*

[2] Mor Harchol-Balter. Lecture Notes: Performance Analysis and Design of Computer Systems. School of Computer Science, CMU, 2000.

[3] G. P. Barker and R. J. Plemmons. Convergent Iterations for Computing Stationary Distributions of Markov Chains. *SIAM Journal, 1985.*

[4] V. A. Barker. Numerical Solution of Sparse Singular Systems of Equations Arising From Ergodic Markov Chains. *Commun. Statist. – Stochastic Models, 5(3), 335-381 (1991).*

[5] Richard Barrett, Michael Berry, James Demmel, $\cdots$, Henk van der Vorst. Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods. *SIAM, 1994.*

[6] M. Benmammoun, J.M. Fourneay, N. Pekergin, A. Troubnikoff. An Algorithmic and Numerical Approach to Bound the Performance of High Speed Networks. *Proceedings of* 10th *IEEE Int'l Symp. on Modeling, Analysis and Simulation of Computer and Telecommunications Systems.*

[7] Abraham Berman, Robert J. Plemmons. Nonnegative Matrices in the Mathematical Sciences. *Academic press, 1979.*

[8] Gunter Bolch, Stefan Greiner, Hermann de Meer and Kishor S. Trivedi. Queueing Networks and Markov Chains. *John Wiley & Sons, Inc, 1998.*

[9] F. Boujdaine, T. Dayar, J.M. Fourneau, N. Pekergin, S. Saadi, J.M. Vincent. A New Proof of st-comparison for Polynomials of a Stochastic Matrix, submitted

[10] Peter Buchholz. Hierarchical Markovian Models: Symmetries and Reduction. *Performance Evaluation, Vol.22, 93-110, (1995).*

[11] Peter Buchholz. Exact and Ordinary Lumpability in Finite Markov Chains. *Journal of Applied Probability, 1994.*

[12] Peter Buchholz. Lumpability and Nearly-Lumpability in Hierachical Queueing Networks. *IEEE 82-91, 1995.*

[13] Peter Buchholz. An Adaptive Aggregation/Disaggregation Algorithm for Hierachical Markovian Models. *European Journal of Operations Research 116 (1999), 545-564.*

[14] Grace E. Cho, C.D. Meyer. Aggregation/Disaggregation Methods of Nearly Uncoupled Markov Chains. `http://meyer.math.ncsu.edu/Meyer/home.html`.

[15] Moody T. Chu, Robert E. Funderlic, and Gene H. Golub. A Rank-One Redution Formula and its Applications to Matrix Factorizations. In *SIAM Review, Volume 37, No. 4, 512-530, 1995.*

[16] G.Ciardo, J. Muppala, and K.S. Trivedi. SPNP: Stochastic Petri Net Package. In Proceedings of the 3rd international Workshop on Petri Nets and Performance models. pages 142-151 *IEEE Computer Society Press, 1989.*

[17] Gianfranco Ciardo, Evgenia Smirni. ETAQA: An Efficient Technique for the Analysis of QBD-processes by Aggregation. *Preprint submitted to Elsevier Science, 2002.*

[18] P. J. Courtois. Decomposability: Queueing and Computer System Applications. *Acacemic Press, New York, 1977.*

[19] P. J. Courtois, P. Semal. Block Iterative Algorithms for Stochastic Matrices. In *Linear Algebra and its Application 59:70-70 (1986)*

[20] Tugrul Dayar, Nihal Pekergin. Stochastic Comparison, Re-orderings, and Nearly Completely Decomposable Markov Chains. *Numerical Solution of Markov Chains (NSMC'99), pp.228-246 (1999).*

[21] Tugrul Dayar, J.M. Fourneay and Nihal Pekergin. Transforming Stochastic Matrices for Stochastic Comparison with the st-order. *RAIRO Operations Research, Volume 37, pp.85-97 (2003).*

[22] Tugrul Dayar and William J. Stewart. Quasi-Lumpability, Lower Bounding Coupling Matrices and Nearly Completely Decomposable Markov Chains. In *SIAM Journal on Matrix Analysis and Applications, Volume 18, Number 2, 1997.*

[23] A. Demir and P. Feldmann. Modeling and Simulation of Interference Noise in Electronic Integrated Circuits with Markov chain Models. In *B. Plateau, W.J. Stewart, and M. Silva, eds. Numerical Solution of Markov Chains, pp. 149-168. Zaragoza: Prensas Universitarias de Zaragoza, 1999.*

[24] Brion N. Feinberg; Samuel S. Chiu. A Method to Calculate Steady-State Distributions of Large Markov Chains by Aggregating States, *Operations Research, vol 35, Issue 2, 282-290, 1987.*

[25] J.M. Fourneau and N. Pekergin. An Algorithmic Approach to Stochastic Bounds. *Performance 2002, LNCS 2459, pp. 64-88 (2002).*

[26] J.M. Fourneau, B. Plateau, I. Sbeity and W.J. Stewart. Stochastic Automata Networks and Lumpable Stochastic Bounds: Bounding Availability. *submitted for publication, April, 2004.*

[27] G. Franceschinis and Richard r. Muntz. Bounds for Quasi-lumpable Markov Chains. In *Performance Evaluation, 20 (1994), 223-243*

[28] Robert E. Funderlic, R. J. Plemmons. Updating LU Factorizations for Computing Stationary Distributions. *SIAM Journal Alg. Disc. METH. Vol 7, No 1, 30-42, 1986.*

[29] G.H. Golub, C.F. Van Loan. Matrix Computation. *The Johns Hopkins University Press, Baltimore, 1989.*

[30] W.K. Grassmann, M.I. Taskar, and D.P. Heyman. Regenerative Analysis and Steady State Distributions for Markov Chains. *Operations Research, 33(5):1107-1116, 1985.*

[31] M. Grinfeld and P. A. Knight. A Weak Lumpability in the k-SAT problem. *Applied Mathematics Letters, 13 (2000), 49-53.*

[32] Daniel P. Heyman. Comparisons Between Aggregation/Disaggregation and a Direct Algorithm for Computing the Stationary Probabilities of a Markov Chain. *ORSA Journal of Computing, Vol 7, No.1, pp. 101-108, 1995.*

[33] Anston S. HouseHolder. The Theory of Matrices in Numerical Analysis. In *Blaisdell Publishing company, 1964.*

[34] Octavian Iordache and Sergiu Corbu. A Stochastic Model of Lumping. *Chemical Engineering Science, Vol. 42, No. 1, pp. 125-132, 1987.*

[35] Linda Kaufman. Matrix Methods for Queueing Problems. In *SIAM Sci. Stat. Comput, Volume 4, No. 3, 525-552, 1983.*

[36] J. R. Kemeny and R. J. Snell. Finite Markov Chains. *D. Van Nostrand, New York, 1960.*

[37] G. N. Krieg and H. Kuhn. A Decomposition Method for Multi-product Kanban Systems with Setup Times and Lost Sales. *IIE Transactions 34(7), pp. 613-625, July, 2002*

[38] Udo R. Krieger, Bruno Muller-Clostermann, and Michael Sczittnick. Modeling and Analysis of Communication Systems Based on Computational Methods for Markov Chains. *IEEE journal on Selected Areas in Communications, Vol. 8, No. 9, 1990.*

[39] Dogancay Kutluyil and Vikram Krishnamurthy. Quick Aggregation of Markov Chain Functionals via Stochastic Complementatioin. *IEEE, 1997.*

[40] V.G. Kulkarni. Modeling and Analysis of Stochastic Systems. *New York: Chapman and Hall, 1995.*

[41] Genyan Li and Herschel Rabitz. A General Analysis of Exact Lumping in Chemical kinetics. *Chemical Engineering Science, Vol. 44, No. 6, pp. 1413-1430, 1989.*

[42] A. Mahmood, D.J. Lynch and L.d. Phillip. A Fast Banded Matrix Inversion Using Connectivity of Schur's Complements. *IEEE 303-306 (1991).*

[43] C.D. Meyer. Stochastic Complementation, Uncompling Markov Chains and the Theory of Nearly Reducible Systems. *SIAM Review, 31(2):240-272, 1989.*

[44] Carl D. Meyer. Matrix Analysis and Applied Linear Algebra. In *SIAM, 2000* .

[45] Carl D. Meyer, Jr.; R. J. Plemmons. Convergent Powers of a Matrix with Applications to Iterative Methods for Singular Linear Systems. *SIAM Journal on Numerical Analysis, Vol 14, Issue 4, 699-705 (1977).*

[46] Henryk Minc. Nonnegative Matrices. *John Wiley Sons, 1988.*

[47] Lynda Modkad, Jalel Ben-Othman and Abdelhak Gueroui. Quality of Service of a Rerouting Algorithm in Wireless Networks. *Quality of Service over Next-Generation Data Networks, Mohammed Atiquzzaman, Mahbub Hassan, Editors, Proceedings of SPIE Vol.4524, 2001.*

[48] Abdel-Moneim A. M and F.W. Leysieffer. Weak Lumpability in Finite Markov Chains *J. Appl. Prob, 19 (1982), 685-691.*

[49] Nan-Fu Peng. On Weak Lumpability of a Finite Markov Chain. In *Statistics & Probability Letters, 27 (1996), 313-318*

[50] Alma Riska, Evgenia Smirni. M/G/1-Type Markov Processes: A Tutorial. *Performance 2002, LNCX 2459, pp.36-63, Springer-Verlag Berlin Heidelberg 2002.*

[51] Alma Riska, Evgenia Smirni. Exact Aggregat Solution for M/G/1-type Markov Processes. `http://www.cs.wm.edu/ riska/publications.html`.

[52] Thomas G. Robertazzi. Recursive Solution of Non-Product Form. *IEEE, 38-46 (1989).*

[53] Thomas G. Robertazzi. Recursive Comptation of Stead-State Probabilities of Non-product Form Queueing Networks Associated with Computer Network Models. *IEEE Transactions on Commumunications, Vol. 38, No. 1. 115-117 (1990).*

[54] Sheldon M. Ross. Introduction to Probability Models, Seventh Edition. *Academic Press, 2000.*

[55] Gerardo Rubino and Bruno Sericola. On Weak Lumpability in Markov Chains. *J. Appl. Prob, 26 (1989), 233-237.*

[56] Gerardo Rubino and Bruno Sericola. A Finite Characterization of Weak Lumpable Markov Processes. Part I: The Discret Time Case. *Stochastic Processes and their Applications 38 (1991) 195-204.*

[57] Gerardo Rubino and Bruno Sericola. A Finite Characterization of Weakly Lumpable Markov Processes. Part II: The Continuous Time Case. *Stochastic Processes and their Applications 45 (1993) 115-125.*

[58] William H. Sanders. Computer Systems Analysis. *Course notes for ECE 541 (Fall 2003), University of Illinois at Urbana Champion.*

[59] R.R. Sarukkai. Link Prediction and Path Analysis Using Markov Chains. `http://www9.org/w9cdrom/68/68.html`.

[60] Arun K. Somani. Reliability Modeling of Structured Systems: Exploring Symmetry in State-Space Generation. *IEEE 78-84 ,1997.*

[61] G.W. Stewart. Implementing an Algorithm for Solving Block Hessenberg Systems. `http://www.cs.umd.edu/ stewart/` .

[62] G.W. Stewart. On the Solution of Block Hessenberg Systems. `http://www.cs.umd.edu/ stewart/` .

[63] G.W. Stewart, W.J. Stewart and D.F. McAllister. A Two Stage Iteratin for Solving Nearly Uncoupled Markov Chains. *IMA Volumes in Mathematics and Its Applications, Vol.60:1 Recent Advances in Iterative Methods, pp210-216. Springer-Verlag, 1993.*

[64] William J. Stewart. Introduction to the Numerical Solution of Markov Chains. *Princeton University Press, 1994.*

[65] William J. Stewart and Wei Wu. Numerical Experiments with Iteration and Aggregation for Markov Chains. `http://www.csc.ncsu.edu/faculty/WStewart/index.html`.

[66] William J. Stewart, A. Touzene. On Solving Stochastic Coupling Matrices Arising in Iterative Aggregation/Disaggregation Methods. *IEEE, 1994.*

[67] D. Stoyan. Comparison Methods for Queues and Other Stochastic Models. *Wiley, 1983.*

[68] Marlin U. Thomas and Donald R. Barr. An Approximate Test of Markov Chain Lumpability. *Journal of the American Statistical Association, Vol. 72, Issue 357, pp. 175-179, 1977.*

[69] Michel Trehel, Chantal Balayer,and Abdelghani Alloui. Modeling Load Balancing inside Groups using Queueing Theory. *The 10th int. conf. on Parall. and Distr. Comp. (PDCS'97), New Orleans, USA, October 1997.*

[70] K.S. Trivedi. Probability and Statistics with Reliability, Queueing and Computer Science Applications. *Englewood, NJ: Prentice-hall, 2002.*

[71] L. Truffet. Reduction Technique For Discrete Time Markov Chains on Totally Ordered State Space Using Stochastic Comparisons. *Journal of Applied Probability, Volume 37, pp. 795-806, 2000.*

[72] Hendrik Vantilborgh, Exact Aggregation in Exponential Queueing Networks, *Journal of the Association for Computing machinery, Vol 25, No 4, 620-629 (1978).*

[73] Langford B. White, Robert Mahony and Gary D. Brushe. Lumpable Hidden Markov Models– Model Reduction and Reduced Complexity Filtering. *IEEE Transactions on Automatic Control, vol.45, No.12, 2297-2306, (2001).*

[74] Aiguo Xie, Peter A. Beerel. Efficient State Classification of Finite State Markov Chains. *IEEE 605-610 ,1998.*

[75] M. Young. Iterative Solution of Large Linear Systems. *Academic Press (1971).*

[76] Z. Zaman. Coach Markov Pulls Goalie Poisson. *Chance, 14(2):31-35 (2001).*