Matrix-Geometric Analysis and Its Applications

John C.S. Lui

Department of Computer Science & Engineeering
The Chinese University of Hong Kong
Email: cslui@cse.cuhk.edu.hk

Summer Course at Tsinghua, 2005.

Outline

Motivation

- Closed-form solution is hard to obtain.
- Need to seek efficient, numerical stable solutions.
- Can be viewed as a generalization of conventional queueing analysis.
- A Special way to solve a Markov Chain.

Motivation

- Closed-form solution is hard to obtain.
- Need to seek efficient, numerical stable solutions.
- Can be viewed as a generalization of conventional queueing analysis.
- A Special way to solve a Markov Chain.

Motivation

- Closed-form solution is hard to obtain.
- Need to seek efficient, numerical stable solutions.
- Can be viewed as a generalization of conventional queueing analysis.
- A Special way to solve a Markov Chain.

Motivation

- Closed-form solution is hard to obtain.
- Need to seek efficient, numerical stable solutions.
- Can be viewed as a generalization of conventional queueing analysis.
- A Special way to solve a Markov Chain.

Outline

Key Ideas

- It is a technique to solve stationary state probability for vector state Markov processes. Two parts:
  1. Boundary set
  2. Repetitive set
- Example: a modified $M/M/1$, λ^* if the system is empty, else λ. Customers require two exponential stages of service, μ_1, and μ_2

$$S : \{(i,s) | i \geq 0 \quad \text{and it is the no. of customer in the queue,} \\ s \text{ is the current stage of service, } s \in (1,2)\}$$

- $s = 0$ if no customer in the system
- Well, let us proceed to specify the state transition diagram, then the $\mathbf{Q}$ matrix.

Key Ideas

- It is a technique to solve stationary state probability for vector state Markov processes. Two parts:
  1. Boundary set
  2. Repetitive set
- Example: a modified $M/M/1$, λ^* if the system is empty, else λ. Customers require two exponential stages of service, μ_1, and μ_2

$$S : \{(i, s) | i \geq 0 \quad \text{and it is the no. of customer in the queue,} \\ s \text{ is the current stage of service, } s \in (1, 2)\}$$

- $s = 0$ if no customer in the system
- Well, let us proceed to specify the state transition diagram, then the $\mathbf{Q}$ matrix.

Key Ideas

- It is a technique to solve stationary state probability for vector state Markov processes. Two parts:
  1. Boundary set
  2. Repetitive set
- Example: a modified $M/M/1$, λ^* if the system is empty, else λ. Customers require two exponential stages of service, μ_1, and μ_2

$$S : \{(i, s) | i \geq 0 \quad \text{and it is the no. of customer in the queue,} \\ s \text{ is the current stage of service, } s \in (1, 2)\}$$

- $s = 0$ if no customer in the system
- Well, let us proceed to specify the state transition diagram, then the $\mathbf{Q}$ matrix.

Key Ideas

- It is a technique to solve stationary state probability for vector state Markov processes. Two parts:
  1. Boundary set
  2. Repetitive set
- Example: a modified $M/M/1, \lambda^*$ if the system is empty, else λ. Customers require two exponential stages of service, μ_1, and μ_2

$$S : \{(i, s) | i \geq 0 \text{ and it is the no. of customer in the queue,} \} \\ s \text{ is the current stage of service, } s \in (1, 2)\}$$

- $s = 0$ if no customer in the system
- Well, let us proceed to specify the state transition diagram, then the $\mathbf{Q}$ matrix.

Key Ideas

- It is a technique to solve stationary state probability for vector state Markov processes. Two parts:
  1. Boundary set
  2. Repetitive set
- Example: a modified $M/M/1, \lambda^*$ if the system is empty, else λ. Customers require two exponential stages of service, μ_1, and μ_2

$$S : \{(i,s)|i \geq 0 \text{ and it is the no. of customer in the queue,} \\ s \text{ is the current stage of service, } s \in (1,2)\}$$

- $s = 0$ if no customer in the system
- Well, let us proceed to specify the state transition diagram, then the Q matrix.

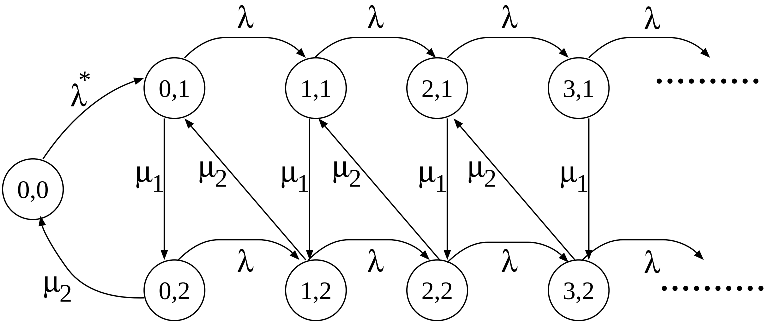
Key Ideas

- It is a technique to solve stationary state probability for vector state Markov processes. Two parts:
  1. Boundary set
  2. Repetitive set
- Example: a modified $M/M/1, \lambda^*$ if the system is empty, else λ. Customers require two exponential stages of service, μ_1, and μ_2

$$S : \{(i,s) | i \geq 0 \text{ and it is the no. of customer in the queue,} \\ s \text{ is the current stage of service, } s \in (1,2)\}$$

- $s = 0$ if no customer in the system
- Well, let us proceed to specify the state transition diagram, then the $\boldsymbol{Q}$ matrix.

Key Idea



Let $a_i = \lambda + \mu_i$, $i = 1, 2$. **Arrange states lexicographically**, $(0,0), (0,1), (0,2), (1,1), (1,2), \ldots$.

The transition rate matrix $\boldsymbol{Q}$ is:

$$\boldsymbol{Q} = \left[\begin{array}{ccc|cc|cc|cc|c}
-\lambda^* & \lambda^* & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & -a_1 & \mu_1 & \lambda & 0 & 0 & 0 & 0 & 0 & \cdots \\
\mu_2 & 0 & -a_2 & 0 & \lambda & 0 & 0 & 0 & 0 & \cdots \\
\hline
0 & 0 & 0 & -a_1 & \mu_1 & \lambda & 0 & 0 & 0 & \cdots \\
0 & \mu_2 & 0 & 0 & -a_2 & 0 & \lambda & 0 & 0 & \cdots \\
\hline
0 & 0 & 0 & 0 & 0 & -a_1 & \mu_1 & \lambda & 0 & \cdots \\
0 & 0 & 0 & \mu_2 & 0 & 0 & -a_2 & 0 & \lambda & \cdots \\
\hline
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots
\end{array}\right.$$

Let us re-write the $\boldsymbol{Q}$ in matrix form:

$$\boldsymbol{A}_0 = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}; \boldsymbol{A}_1 = \begin{bmatrix} -a_1 & \mu_1 \\ 0 & -a_2 \end{bmatrix}; \boldsymbol{A}_2 = \begin{bmatrix} 0 & 0 \\ \mu_2 & 0 \end{bmatrix}$$

$$\boldsymbol{B}_{00} = \begin{bmatrix} -\lambda^* & \lambda^* & 0 \\ 0 & -a_1 & \mu_1 \\ \mu_2 & 0 & -a_2 \end{bmatrix}; \boldsymbol{B}_{01} = \begin{bmatrix} 0 & 0 \\ \lambda & 0 \\ 0 & \lambda \end{bmatrix}; \boldsymbol{B}_{10} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \mu_2 & 0 \end{bmatrix}$$

$$\boldsymbol{Q} = \left[\begin{array}{cc|cccc} \boldsymbol{B}_{00} & \boldsymbol{B}_{01} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \boldsymbol{B}_{10} & \boldsymbol{A}_1 & \boldsymbol{A}_0 & \mathbf{0} & \mathbf{0} & \cdots \\ \hline \mathbf{0} & \boldsymbol{A}_2 & \boldsymbol{A}_1 & \boldsymbol{A}_0 & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \boldsymbol{A}_2 & \boldsymbol{A}_1 & \boldsymbol{A}_0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ldots \end{array}\right]$$

Let us solve it. For the repetitive portion

$$\boldsymbol{\pi}_{j-1}\boldsymbol{A}_0 + \boldsymbol{\pi}_j\boldsymbol{A}_1 + \boldsymbol{\pi}_{j+1}\boldsymbol{A}_2 = \mathbf{0} \quad j = 2, 3, \ldots \tag{1}$$

This is *similar* to the solution of $M/M/1$. Therefore, $\boldsymbol{\pi}_j$ is a function only of the transition rates between states with $j-1$ queued customers and states with j queued customers.

$$\begin{aligned} \boldsymbol{\pi}_j &= \boldsymbol{\pi}_{j-1}\boldsymbol{R} \quad j = 2, 3, \ldots \\ \text{or} \quad \boldsymbol{\pi}_j &= \boldsymbol{\pi}_1\boldsymbol{R}^{j-1} \quad j = 2, 3, \ldots \end{aligned} \tag{2}$$

Putting (2) into (1), we have:

$$\boldsymbol{\pi}_1 \boldsymbol{R}^{j-2}\boldsymbol{A}_0 + \boldsymbol{\pi}_1 \boldsymbol{R}^{j-1}\boldsymbol{A}_1 + \boldsymbol{\pi}_1 \boldsymbol{R}^{j}\boldsymbol{A}_2 = \mathbf{0} \quad j = 2, 3, \ldots$$

Since it is true for $j = 2, 3, \ldots$, substitute $j = 2$, we have:

$$\boldsymbol{A}_0 + \boldsymbol{R}\boldsymbol{A}_1 + \boldsymbol{R}^2\boldsymbol{A}_2 = \mathbf{0}$$

For the initial portion:

$$\begin{aligned} \boldsymbol{\pi}_0 \boldsymbol{B}_{00} + \boldsymbol{\pi}_1 \boldsymbol{B}_{10} &= \mathbf{0} \\ \boldsymbol{\pi}_0 \boldsymbol{B}_{01} + \boldsymbol{\pi}_1 \boldsymbol{A}_1 + \boldsymbol{\pi}_2 \boldsymbol{A}_2 &= \mathbf{0} \end{aligned}$$

or

$$[\boldsymbol{\pi}_0, \boldsymbol{\pi}_1] \begin{bmatrix} \boldsymbol{B}_{00} & \boldsymbol{B}_{01} \\ \boldsymbol{B}_{10} & \boldsymbol{A}_1 + \boldsymbol{R}\boldsymbol{A}_2 \end{bmatrix} = \mathbf{0}$$

We also need:

$$1 = \boldsymbol{\pi}_0 \boldsymbol{e} + \boldsymbol{\pi}_1 \sum_{j=1}^{\infty} \boldsymbol{R}^{j-1} \boldsymbol{e} = \boldsymbol{\pi}_0 \boldsymbol{e} + \boldsymbol{\pi}_1 (\boldsymbol{I} - \boldsymbol{R})^{-1} \boldsymbol{e}$$

$$[\boldsymbol{\pi}_0, \boldsymbol{\pi}_1] \begin{bmatrix} \boldsymbol{e} & \boldsymbol{B}_{00}^* & \boldsymbol{B}_{01} \\ (\boldsymbol{I} - \boldsymbol{R})^{-1} \boldsymbol{e} & \boldsymbol{B}_{10}^* & \boldsymbol{A}_1 + \boldsymbol{R}\boldsymbol{A}_2 \end{bmatrix} = [1, \mathbf{0}]$$

where $\boldsymbol{M}^*$ is $\boldsymbol{M}$ with first column being eliminated.

$$
\begin{aligned}
\bar{N}_q &= \text{E[queued customers]} \\
&= \sum_{j=1}^{\infty} j\, \pi_j \boldsymbol{e} = \sum_{j=1}^{\infty} (j)\, \pi_1 \boldsymbol{R}^{j-1} \boldsymbol{e} = \pi_1 (\boldsymbol{I} - \boldsymbol{R})^{-2} \boldsymbol{e}
\end{aligned}
$$

$$
\begin{aligned}
\text{Note:} \qquad \boldsymbol{S} &= \sum_{j=1}^{\infty} \boldsymbol{R}^{j-1} = \boldsymbol{I} + \boldsymbol{R} + \boldsymbol{R}^2 + \cdots \\
\boldsymbol{SR} &= \boldsymbol{R} + \boldsymbol{R}^2 + \boldsymbol{R}^3 + \cdots \\
\boldsymbol{S}(\boldsymbol{I} - \boldsymbol{R}) &= \boldsymbol{I} \\
\boldsymbol{S} &= \boldsymbol{I}\,(\boldsymbol{I} - \boldsymbol{R})^{-1} = (\boldsymbol{I} - \boldsymbol{R})^{-1}
\end{aligned}
$$

This is true only when the spectral radius of $\boldsymbol{R}$ is less than unity.

Outline

$$\boldsymbol{Q} = \left[\begin{array}{cc|c|c|c|c}
\boldsymbol{B}_{00} & \boldsymbol{B}_{01} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots \\
\boldsymbol{B}_{10} & \boldsymbol{B}_{11} & \boldsymbol{A}_{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots \\
\hline
\boldsymbol{B}_{20} & \boldsymbol{B}_{21} & \boldsymbol{A}_{1} & \boldsymbol{A}_{0} & \boldsymbol{0} & \cdots \\
\hline
\boldsymbol{B}_{30} & \boldsymbol{B}_{31} & \boldsymbol{A}_{2} & \boldsymbol{A}_{1} & \boldsymbol{A}_{0} & \cdots \\
\hline
\boldsymbol{B}_{40} & \boldsymbol{B}_{41} & \boldsymbol{A}_{3} & \boldsymbol{A}_{2} & \boldsymbol{A}_{1} & \cdots \\
\hline
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots
\end{array}\right.$$

We index the state by (i, j) where i is the level, $i \geq 0$ and j is the state within the level, $0 \leq j \leq m - 1$ for $i \geq 1$.

For the repetitive portion,

$$\sum_{k=0}^{\infty} \pi_{j-1+k} \boldsymbol{A}_k = \mathbf{0} \quad j = 2, 3, \ldots \tag{3}$$

$$\pi_j = \pi_1 \boldsymbol{R}^{j-1} \qquad j = 2, 3, \ldots \tag{4}$$

putting (4) to (3), we have:

$$\sum_{k=0}^{\infty} \boldsymbol{R}^k \boldsymbol{A}_k = \mathbf{0}$$

For the boundary states, we have:

$$[\pi_0, \pi_1] \begin{bmatrix} \boldsymbol{B}_{00} & \boldsymbol{B}_{01} \\ \sum_{k=1}^{\infty} \boldsymbol{R}^{k-1} \boldsymbol{B}_{k0} & \sum_{k=1}^{\infty} \boldsymbol{R}^{k-1} \boldsymbol{B}_{k1} \end{bmatrix}$$

Procedure (continue:)

Using the same normalization, we have

$$[\boldsymbol{\pi}_0, \boldsymbol{\pi}_1] \begin{bmatrix} \mathbf{e} & \mathbf{B}_{00}^* & \mathbf{B}_{01} \\ (\mathbf{I}-\mathbf{R})^{-1}\mathbf{e} & \left[\sum_{k=1}^{\infty} \mathbf{R}^{k-1}\mathbf{B}_{k0}\right]^* & \sum_{k=1}^{\infty} \mathbf{R}^{k-1}\mathbf{B}_{k1} \end{bmatrix} = [1, \mathbf{0}]$$

Therefore, it boils down to

1. Solving $\mathbf{R}$.
2. Solving the initial portion of the Markov process.

Procedure (continue:)

Using the same normalization, we have

$$[\boldsymbol{\pi}_0, \boldsymbol{\pi}_1] \begin{bmatrix} \boldsymbol{e} & \boldsymbol{B}_{00}^* & \boldsymbol{B}_{01} \\ (\boldsymbol{I} - \boldsymbol{R})^{-1}\boldsymbol{e} & \left[\sum_{k=1}^{\infty} \boldsymbol{R}^{k-1}\boldsymbol{B}_{k0}\right]^* & \sum_{k=1}^{\infty} \boldsymbol{R}^{k-1}\boldsymbol{B}_{k1} \end{bmatrix} = [1, \boldsymbol{0}]$$

Therefore, it boils down to

1. Solving $\boldsymbol{R}$.
2. Solving the initial portion of the Markov process.

Outline

Multiprocessing System

We have a multiprocessing system in which

- K homogeneous processors.
- Each processor is subjected to failure with rate γ.
- A single repair facility with repair rate α.
- Jobs arrive at a Poisson rate λ.
- Whenever there is no processor available, all jobs are lost.

Multiprocessing System

We have a multiprocessing system in which

- K homogeneous processors.
- Each processor is subjected to failure with rate γ.
- A single repair facility with repair rate α.
- Jobs arrive at a Poisson rate λ.
- Whenever there is no processor available, all jobs are lost.

Multiprocessing System

We have a multiprocessing system in which

- K homogeneous processors.
- Each processor is subjected to failure with rate γ.
- A single repair facility with repair rate α.
- Jobs arrive at a Poisson rate λ.
- Whenever there is no processor available, all jobs are lost.

Multiprocessing System

We have a multiprocessing system in which

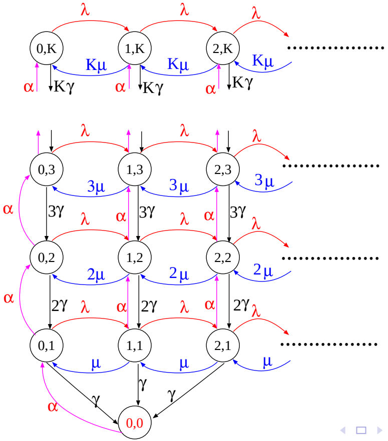
- K homogeneous processors.
- Each processor is subjected to failure with rate γ.
- A single repair facility with repair rate α.
- Jobs arrive at a Poisson rate λ.
- Whenever there is no processor available, all jobs are lost.

Multiprocessing System

We have a multiprocessing system in which

- K homogeneous processors.
- Each processor is subjected to failure with rate γ.
- A single repair facility with repair rate α.
- Jobs arrive at a Poisson rate λ.
- Whenever there is no processor available, all jobs are lost.

Markov Model



Define $b_i = \lambda + i\gamma + \alpha$ for $i = 1, 2, \ldots, K$. We have:

$$\boldsymbol{B}_{00} = [-\alpha]\,; \boldsymbol{B}_{01} = [\alpha, 0, \cdots, 0]\,; \boldsymbol{B}_{j0} = \begin{bmatrix} \gamma \\ 0 \\ \vdots \\ 0 \end{bmatrix}; \boldsymbol{B}_{0,j} = \boldsymbol{0} \quad j = 2, 3, \ldots,$$

$$\boldsymbol{B}_{1,1} = \begin{bmatrix} -b_1 & \alpha & 0 & 0 & \cdots & 0 & 0 & 0 \\ 2\gamma & -b_2 & \alpha & 0 & \cdots & 0 & 0 & 0 \\ 0 & 3\gamma & -b_3 & \alpha & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & (K-1)\gamma & -b_{K+1} & \alpha \\ 0 & 0 & 0 & 0 & \cdots & 0 & K\gamma & -b_K \end{bmatrix}$$

- The matrices of the repeating portion of the process are:

$$\boldsymbol{A}_0 = \lambda \boldsymbol{I}; \boldsymbol{A}_1 = \boldsymbol{B}_{1,1}; \boldsymbol{A}_2 = \begin{bmatrix} \mu & 0 & \cdots & 0 & 0 \\ 0 & 2\mu & \cdots & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & (K-1)\mu & 0 \\ 0 & 0 & \cdots & 0 & K\mu \end{bmatrix}$$

- This can be solved numerically rather than using the transform method.

- The matrices of the repeating portion of the process are:

$$\boldsymbol{A}_0 = \lambda \boldsymbol{I}; \boldsymbol{A}_1 = \boldsymbol{B}_{1,1}; \boldsymbol{A}_2 = \begin{bmatrix} \mu & 0 & \cdots & 0 & 0 \\ 0 & 2\mu & \cdots & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & (K-1)\mu & 0 \\ 0 & 0 & \cdots & 0 & K\mu \end{bmatrix}$$

- This can be solved numerically rather than using the transform method.

Outline

- Informally, the stability of a process depends on the *drift* of the process for states in the repetitive portion.
- For example, $M/M/1$, the expected drift toward higher states is $\lambda 1$. The expected drift toward lower states is $\mu(-1) = -\mu$. The drift of the process is $\lambda - \mu$. Process is stable if the total expected drift is *NEGATIVE*, or $\lambda < \mu$ in our case.
- Now suppose the process can go up by 1 and go down by at most K steps. Let the rate for l steps be $r(l)$, $l = -K, -K-1, \ldots, 0, 1$.

$$r(1) + \sum_{l=1}^{K} (-l)\, r(-l) \rightarrow r(1) < \sum_{l=1}^{K} l\, r(l)$$

- Informally, the stability of a process depends on the *drift* of the process for states in the repetitive portion.
- For example, $M/M/1$, the expected drift toward higher states is $\lambda 1$. The expected drift toward lower states is $\mu(-1) = -\mu$. The drift of the process is $\lambda - \mu$. Process is stable if the total expected drift is *NEGATIVE*, or $\lambda < \mu$ in our case.
- Now suppose the process can go up by 1 and go down by at most K steps. Let the rate for l steps be $r(l)$, $l = -K, -K-1, \ldots, 0, 1$.

$$r(1) + \sum_{l=1}^{K} (-l) r(-l) \to r(1) < \sum_{l=1}^{K} l r(l)$$

- Informally, the stability of a process depends on the *drift* of the process for states in the repetitive portion.
- For example, $M/M/1$, the expected drift toward higher states is $\lambda 1$. The expected drift toward lower states is $\mu(-1) = -\mu$. The drift of the process is $\lambda - \mu$. Process is stable if the total expected drift is *NEGATIVE*, or $\lambda < \mu$ in our case.
- Now suppose the process can go up by 1 and go down by at most K steps. Let the rate for l steps be $r(l)$, $l = -K, -K-1, \ldots, 0, 1$.

$$r(1) + \sum_{l=1}^{K} (-l) r(-l) \rightarrow r(1) < \sum_{l=1}^{K} l r(l)$$

- Analogous to the scalar case, we can think of the drift of the process in terms of *levels*. Assume that for the repetitive portion, we have m states, a transition from level i, $i >> 0$, to level $i-k$, $1 \leq k \leq K$

$$-k \sum_{l=1}^{m} \boldsymbol{A}_{k+1}(j,l)$$

  where $A_{k+1}(j,l)$ is the transition from state j in level i to state l in level $i-k$.

- Let f_j, $0 \leq j \leq m-1$ be the probability that the process is in inter-level j of the repeating portion of the process of level $i >> 0$. The average drift from level i to level $i-k$ is

$$-k \sum^{m-1} f_j \sum^{m-1} \boldsymbol{A}_{k+1}(j,l)$$

- Analogous to the scalar case, we can think of the drift of the process in terms of *levels*. Assume that for the repetitive portion, we have m states, a transition from level i, $i >> 0$, to level $i-k$, $1 \leq k \leq K$

$$-k \sum_{l=1}^{m} \mathbf{A}_{k+1}(j,l)$$

  where $A_{k+1}(j,l)$ is the transition from state j in level i to state l in level $i-k$.
- Let f_j, $0 \leq j \leq m-1$ be the probability that the process is in inter-level j of the repeating portion of the process of level $i >> 0$. The average drift from level i to level $i-k$ is

$$-k \sum^{m-1} f_j \sum^{m-1} \mathbf{A}_{k+1}(j,l)$$

- To get the total drift, we sum the previous equation for all k, $0 \leq k \leq K+1$.
- But what is f_j? Let us define $\boldsymbol{A} = \sum_{l=0}^{K+1} \boldsymbol{A}_l$, we have $\boldsymbol{f} = (f_0, f_1, \ldots, f_{m-1})$. Therefore:

$$\boldsymbol{f}\boldsymbol{A} = \boldsymbol{0} \quad \& \quad \boldsymbol{f}\boldsymbol{e} = 1$$

- The stability condition is:

$$\boldsymbol{f}\boldsymbol{A}_0\boldsymbol{e} < \sum_{k=2}^{K+1}(k-1)\boldsymbol{f}\boldsymbol{A}_k\boldsymbol{e}$$

- To get the total drift, we sum the previous equation for all k, $0 \leq k \leq K+1$.
- But what is f_j? Let us define $\boldsymbol{A} = \sum_{l=0}^{K+1} \boldsymbol{A}_l$, we have $\boldsymbol{f} = (f_0, f_1, \ldots, f_{m-1})$. Therefore:

$$\boldsymbol{f}\boldsymbol{A} = \boldsymbol{0} \quad \& \quad \boldsymbol{f}\boldsymbol{e} = 1$$

- The stability condition is:

$$\boldsymbol{f}\boldsymbol{A}_0\boldsymbol{e} < \sum_{k=2}^{K+1} (k-1)\boldsymbol{f}\boldsymbol{A}_k\boldsymbol{e}$$

- To get the total drift, we sum the previous equation for all k, $0 \leq k \leq K+1$.
- But what is f_j? Let us define $\boldsymbol{A} = \sum_{l=0}^{K+1} \boldsymbol{A}_l$, we have $\boldsymbol{f} = (f_0, f_1, \ldots, f_{m-1})$. Therefore:

$$\boldsymbol{fA} = \boldsymbol{0} \quad \& \quad \boldsymbol{fe} = 1$$

- The stability condition is:

$$\boldsymbol{fA}_0\boldsymbol{e} < \sum_{k=2}^{K+1} (k-1)\boldsymbol{fA}_k\boldsymbol{e}$$

Outline

- It is an iterative method. Let

$$\begin{aligned} \boldsymbol{R}(0) &= \mathbf{0} \\ \boldsymbol{R}(n+1) &= -\sum_{l=0, l\neq 1}^{\infty} \boldsymbol{R}^{l}(n)\boldsymbol{A}_l\boldsymbol{A}_1^{-1} \qquad n = 0, 1, 2, \ldots \end{aligned}$$

- The iterative process halts whenever entries in $\boldsymbol{R}(n+1)$ and $\boldsymbol{R}(n)$ differ in absolute value by less than a given constant.
- The sequence $\{\boldsymbol{R}(n)\}$ are entry-wise non-decreasing and converge monotonically to a non-negative matrix $\boldsymbol{R}$.
- the number of iteration needed for convergence increases as the spectral radius of $\boldsymbol{R}$ increases. This is similar to the scalar case where $\rho \rightarrow 1$. As the system utilization increases, it becomes computationally more difficult to get

- It is an iterative method. Let

$$
\begin{aligned}
\boldsymbol{R}(0) &= \boldsymbol{0} \\
\boldsymbol{R}(n+1) &= -\sum_{l=0, l\neq 1}^{\infty} \boldsymbol{R}^{l}(n)\boldsymbol{A}_l\boldsymbol{A}_1^{-1} \qquad n = 0, 1, 2, \ldots
\end{aligned}
$$

- The iterative process halts whenever entries in $\boldsymbol{R}(n+1)$ and $\boldsymbol{R}(n)$ differ in absolute value by less than a given constant.
- The sequence $\{\boldsymbol{R}(n)\}$ are entry-wise non-decreasing and converge monotonically to a non-negative matrix $\boldsymbol{R}$.
- the number of iteration needed for convergence increases as the spectral radius of $\boldsymbol{R}$ increases. This is similar to the scalar case where $\rho \rightarrow 1$. As the system utilization increases, it becomes computationally more difficult to get

- It is an iterative method. Let

$$\begin{aligned} \boldsymbol{R}(0) &= \boldsymbol{0} \\ \boldsymbol{R}(n+1) &= -\sum_{l=0, l\neq 1}^{\infty} \boldsymbol{R}^l(n)\boldsymbol{A}_l\boldsymbol{A}_1^{-1} \quad n = 0, 1, 2, \ldots \end{aligned}$$

- The iterative process halts whenever entries in $\boldsymbol{R}(n+1)$ and $\boldsymbol{R}(n)$ differ in absolute value by less than a given constant.
- The sequence $\{\boldsymbol{R}(n)\}$ are entry-wise non-decreasing and converge monotonically to a non-negative matrix $\boldsymbol{R}$.
- the number of iteration needed for convergence increases as the spectral radius of $\boldsymbol{R}$ increases. This is similar to the scalar case where $\rho \rightarrow 1$. As the system utilization increases, it becomes computationally more difficult to get

- It is an iterative method. Let

$$\begin{aligned} \boldsymbol{R}(0) &= \mathbf{0} \\ \boldsymbol{R}(n+1) &= -\sum_{l=0, l\neq 1}^{\infty} \boldsymbol{R}^{l}(n)\boldsymbol{A}_l\boldsymbol{A}_1^{-1} \qquad n = 0, 1, 2, \ldots \end{aligned}$$

- The iterative process halts whenever entries in $\boldsymbol{R}(n+1)$ and $\boldsymbol{R}(n)$ differ in absolute value by less than a given constant.
- The sequence $\{\boldsymbol{R}(n)\}$ are entry-wise non-decreasing and converge monotonically to a non-negative matrix $\boldsymbol{R}$.
- the number of iteration needed for convergence increases as the spectral radius of $\boldsymbol{R}$ increases. This is similar to the scalar case where $\rho \to 1$. As the system utilization increases, it becomes computationally more difficult to get

Replicated Database

- Poisson arrival with rate λ.
- Probability it is a read request: r.
- A read request can be served by any server.
- A write request has to be served by **BOTH** servers.
- What is the proper state space?

Replicated Database

- Poisson arrival with rate λ.
- Probability it is a read request: r.
- A read request can be served by any server.
- A write request has to be served by **BOTH** servers.
- What is the proper state space?

Replicated Database

- Poisson arrival with rate λ.
- Probability it is a read request: r.
- A read request can be served by any server.
- A write request has to be served by **BOTH** servers.
- What is the proper state space?

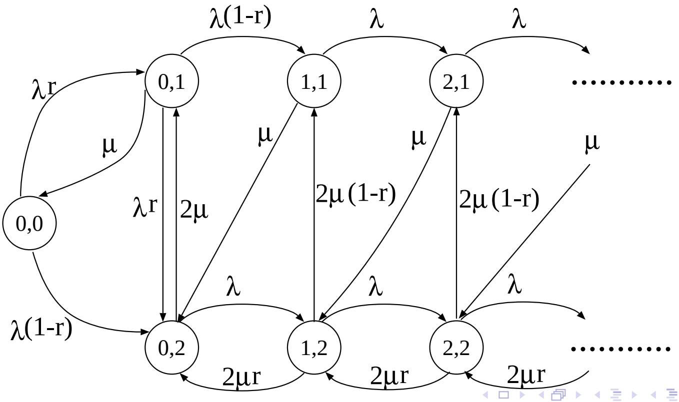
Replicated Database

- Poisson arrival with rate λ.
- Probability it is a read request: r.
- A read request can be served by any server.
- A write request has to be served by **BOTH** servers.
- What is the proper state space?

Replicated Database

- Poisson arrival with rate λ.
- Probability it is a read request: r.
- A read request can be served by any server.
- A write request has to be served by **BOTH** servers.
- What is the proper state space?

The state space $S = (i, j)$ where i is the number of queued customers and j is the number of replications that are involved in service. So $i \geq 0$ and $j = 0, 1, 2$.



$$\boldsymbol{Q} = \left[\begin{array}{ccc|cc|cc}
-\lambda & \lambda r & \lambda(1-r) & 0 & 0 & 0 & \cdots \\
\mu & -(\lambda+\mu) & \lambda r & \lambda(1-r) & 0 & 0 & \cdots \\
0 & 2\mu & -(\lambda+2\mu) & 0 & \lambda & 0 & \cdots \\
\hline
0 & 0 & \mu & -(\lambda+\mu) & 0 & \lambda & \cdots \\
0 & 0 & 2\mu r & 2\mu(1-r) & -(\lambda+2\mu) & 0 & \cdots \\
\hline
0 & 0 & 0 & 0 & \mu & -(\lambda+\mu) & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots
\end{array}\right.$$

$$\boldsymbol{B}_{00} = \begin{bmatrix} -\lambda & \lambda r & \lambda(1-r) \\ \mu & -(\lambda+\mu) & \lambda r \\ 0 & 2\mu & -(\lambda+2\mu) \end{bmatrix}; \boldsymbol{B}_{01} = \begin{bmatrix} 0 & 0 \\ \lambda(1-r) & 0 \\ 0 & \lambda \end{bmatrix}$$

$$\boldsymbol{B}_{10} = \begin{bmatrix} 0 & 0 & \mu \\ 0 & 0 & 2\mu r \end{bmatrix}; \boldsymbol{B}_{11} = \boldsymbol{A}_1 = \begin{bmatrix} -(\lambda+\mu) & 0 \\ 2\mu(1-r) & -(\lambda+2\mu) \end{bmatrix};$$

$$\boldsymbol{A}_0 = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}; \boldsymbol{A}_2 = \begin{bmatrix} 0 & \mu \\ 0 & 2\mu r \end{bmatrix}$$

To determine the stability:

$$\boldsymbol{A} = \begin{bmatrix} -\mu & \mu \\ 2\mu(1-r) & -2\mu(1-r) \end{bmatrix} = \boldsymbol{A}_0 + \boldsymbol{A}_1 + \boldsymbol{A}_2$$

$$f_1 = \frac{2(1-r)}{3-2r}; \qquad f_2 = \frac{1}{3-2r}$$

$$\boldsymbol{f}\boldsymbol{A}_0\boldsymbol{e} < \sum_{k=2}^{K+1}(k-1)\boldsymbol{f}\boldsymbol{A}_k\boldsymbol{e} \rightarrow \lambda < \frac{2\mu}{3-2r}$$