King Fahd University of Petroleum & Minerals
Computer Engineering Dept

COE 540 – Computer Networks
Term 101
Dr. Ashraf S. Hasan Mahmoud
Rm 22-148-3
Ext. 1724
Email: ashraf@kfupm.edu.sa

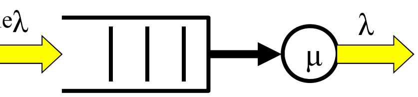
Queueing Networks

- **Feed-forward network (no feedback paths/acyclic)**
  - **Burkes Theorem**
- **Open Network with feedback**
  - **Single node M/M/1 queue**
  - **Two queues Example**
  - **N-node Jackson Network example**
- **Closed Jackson Network**

Burke's Theorem

- **Consider an M/M/1 or M/M/S or M/M/∞ queueing system at steady state with arrival rate λ, then**
    - **The departure process is Poisson with rate λ**
    - **At each time t, the number of customers in the system N(t) is *independent* of the sequence of departure times prior to t.**

M/M/1 example



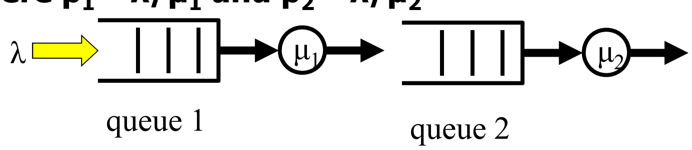
<u>Theorem:</u> The departure process from an M/M/S queue is Poisson and is independent of the content of the queue

Two Queues in Tandem

- **Consider the following system – assuming infinite storage in each queue.**
- **We can show that**

$$P(N_1 = n_1, N_2 = n_2) = (1-\rho_1)\,\rho_1^{n1}(1-\rho_2)\,\rho_2^{n2}$$

where $\rho_1 = \lambda/\mu_1$ and $\rho_2 = \lambda/\mu_2$

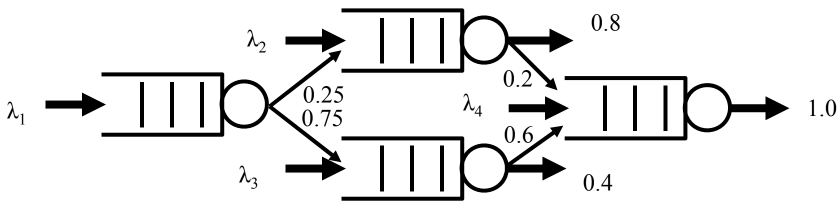


- **The above can be generalized to M queues in Tandem**

$$P(Q_1(t)=k_1, Q_2(t)=k_2, \cdots, Q_M(t)=k_M) = \prod_{i=1}^{M}(1-\rho_i)\rho_i^{k_i}$$

Feedforward Networks – Example 1

- **Example1: feedforward acyclic networks (i.e. no feedback paths)**



- **Since joining and splitting of Poisson streams results in Poisson streams – Burks' theorem still applicable**
- **Solution key: deal with the individual queues <u>after determining the total flow to the queue</u>**

Traffic Equation and Routing Matrix

- **Consider a network of N queues, each having an independent exponential server and an infinite buffer**
- **External arrivals at each node – Poisson with rate λ_i**
- **Messages are routed probabilistically:**
  - **q_{ji}: i,j = 1, 2, 3, …, N is the probability of a message being routed from node j to node i**
  - **q_{jN+1}: is the probability of a message being routed outside the network**
  - **Note:** $\sum_{i=1}^{N+1} q_{ji} = 1$

Traffic Equation and Routing Matrix – cont'd

- **Let Λ_i: total flow into the i^th node**
- **Clearly, one can write**

$$\Lambda_i = \lambda_i + \sum_{j=1}^{N} q_{ji}\Lambda_j$$

- **The matrix version is**

$$[\Lambda_1 \quad \Lambda_2 \quad \cdots \quad \Lambda_N] = [\lambda_1 \quad \lambda_2 \quad \cdots \quad \lambda_N] + [\Lambda_1 \quad \Lambda_2 \quad \cdots \quad \Lambda_N]\begin{bmatrix} 0 & q_{12} & \cdots & q_{1N} \\ q_{21} & 0 & \cdots & q_{2N} \\ \vdots & & 0 & \\ q_{N1} & q_{N2} & \cdots & 0 \end{bmatrix}$$

or

$$\Lambda = \lambda + \Lambda Q$$

Note:
$q_{jj} = 0$ – no routing to same source node

Traffic Equation and Routing Matrix – cont'd

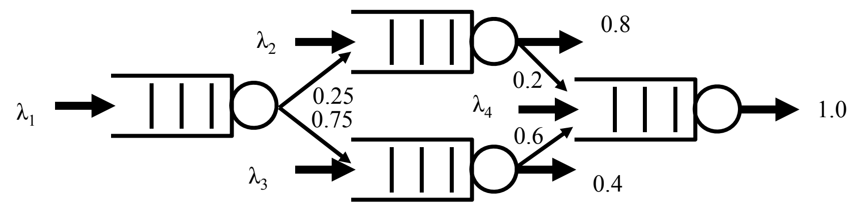
- **Normally the inputs and the routing matrix are known, the total flow to each node can be found using**

$$\Lambda = \lambda[I - Q]^{-1}$$

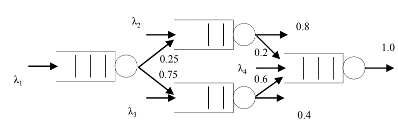
where *I* is the NxN identity matrix

Example:

- **<u>Problem</u>: Consider the network of queue depicted in figure. If the arrival rates are given by λ = [2.0, 1.0, 0.5, 3.0], and the service rates are μ = [4.0, 6.0, 11.0, 9.9],**
  - **a) compute the total flow into each node**
  - **b) Find the joint pmf for number of customers in queues**



Example:



- **<u>Solution</u>:**

a) The routing matrix for the network is given by

$$Q = \begin{bmatrix} 0 & 0.25 & 0.75 & 0 \\ 0 & 0 & 0 & 0.2 \\ 0 & 0 & 0 & 0.6 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Therefore, total flows are given by

Λ = [2.0, 1.5, 2.0, 4.5]

b) The loads for the queues are given by

R = Λ./μ (./ is the element-by-element division – Matlab notation)

= [1/2 , 1/4, 2/11, 5/11]

The joint pmf for the number of customers is given by

$$P(Q_1 = k_1, Q_2 = k_2, Q_3 = k_3, Q_4 = k_4) = \prod_{i=1}^{4} (1 - R_i) R_i^{k_i}$$
$$= (81/484)(1/2)^{k_1} (1/4)^{k_2} (2/11)^{k_3} (5/11)^{k_4}$$

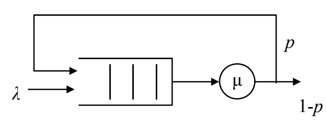
Open Network – Flows with Feedback Paths

- **Open networks – at least one external source of arrivals ➔ there must be a flow to outside network (exit path)**
  - **i.e. sum $q_{ki} < 1$ for at least one k = 1, 2, …, N**

Example: Single M/M/1 queue with Feedback

- **Problem: Consider the following system – Find the pmf for number of customers in the system.**
- **Solution:**

$\Lambda = \lambda + p\,\Lambda$ ➔ $\Lambda = \lambda/(1-p)$
Therefore, traffic load, R is given by
$R = \Lambda/\mu = \lambda/[\mu(1-p)]$
$Prob(N = k) = (1-R)R^k$ k=0, 1, 2, …



Note $R < 1$ ➔ $\lambda/[\mu(1-p)] < 1$ or $\lambda < \mu(1-p)$ – this imposes a limit on the maximum arrival rate

What is the average number of customer visits to the queue?

$E[N] = R/(1-R)$
$E[T] = E[N] / \lambda$ - direct application of Little's formula

For a general solution of an M/G/1 with Bernoulli feedback check: L. Takács, "A Single-Server Queue with Feedback," Bell Technical Journal, March 1963, pp. 505-519.

Open Network with Feedback – 2 Node example

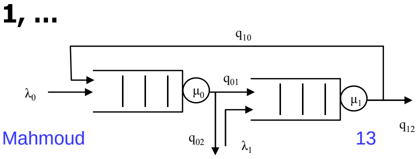
- **Consider the two queues network depicted in figure**
- **The total flow equations are given by**

$$[\Lambda_0 \quad \Lambda_1] = [\lambda_0 \quad \lambda_1] + [\Lambda_0 \quad \Lambda_1]\begin{bmatrix} 0 & q_{01} \\ q_{10} & 0 \end{bmatrix}$$

- **System state: (k_0, k_1) where k_0 is number of customers in queue 0 while k_1 is number of customers in queue 1**
- **Define P($Q_0(t)=k_0$, $Q_1(t)=k_1$) = P(k_0, k_1; t) – as the probability of k_i customers in the respective queue at time t.**
- **One can show that the product form still holds, i.e.**

$$P(k_0,k_1) = (1-\rho_0)(1-\rho_1)\,\rho_0^{k0}\,\rho_1^{k1}$$
$$k_0,k_1 = 0, 1, \ldots$$

where ρ_i is given by Λ_i/μ_i



Example: Two-Node Network

- **<u>Problem</u>: Let λ = [2.0, 1.0], and the service rates are μ = [15.625, 3.75] – Let the routing parameters q01 = 0.4 and q10 = 0.5.**
  - **A) compute the total flow into each queue**
  - **B) Compute the traffic utilization of each queue – write an expression for the joint pmf of number of customers in the network**
  - **C) Compute the average number of messages in node 0 and the average number of messages in node 1**
  - **D) Calculate the average end-to-end delay for a customer**

Example: Two-Node Network – cont'd

- **Solution:**

A) The total flow is found by solving the following set of equations:

$$[\Lambda_0\ \Lambda_1] = [\lambda_0\ \lambda_1] + [\Lambda_0\ \Lambda_1] \begin{bmatrix} 0 & q01 \\ q10 & 0 \end{bmatrix}$$

Therefore, $[\Lambda_0\ \Lambda_1] = [3.125\ 2.25]$

B) Traffic Utilization: $R = \Lambda./\mu$

$= [0.2\ 0.6]$

$P(k_0, k_1) = 0.32(0.2)^{k0}(0.6)^{k1}$ for k0, k1 = 0, 1, …

C) $E[N_0] = R_0/(1 - R_0) = 0.25$

$E[N_1] = R_1/(1 - R_1) = 1.5$

Note that $E[N] = E[N_0] + E[N_1]$

$= 1.75$

D) $E[T] = E[N]/(\lambda_0+\lambda_1) = 0.583$ seconds

N-Node Open Jackson Networks – Problem Specification

- **Consider an N-Node open network that is characterized by**
  - **Probabilistic routing matrix $Q = \{q_{ij}\}$**
  - **Set of external flows λ_i; i=1, 2, …, N - Poisson processes**
  - **Infinite storage at each node**
- **Assume S_i servers at each node i – each having exponentially distributed service time**
  - **Departure rate from state k_i (i.e. k_i customers in node i) is equal to**

$$\mu_i d(k_i) = \begin{cases} \mu_i k_i & k_i \le S_i \\ \mu_i S_i & k_i > S_i \end{cases}$$

N-Node Open Jackson Networks – Problem Specification

- **The total flow to the ith node is computed using:**

$$\Lambda_i = \lambda_i + \sum_{j=1}^{N} q_{ji}\Lambda_j$$

- **The queue at node i is stable if $\Lambda_i < \mu_i S_i$; i=1,2, …, N**
- **Using the global balance equation one can show that**

$$P(k_1,k_2,\ldots,k_N) = G^{-1}\prod_{i=1}^{N}\frac{R_i^{k_i}}{\prod_{j=1}^{k_i} d(j)}; \quad \forall k_1,k_2,\ldots,k_N \geq 0$$

where $R_i = \Lambda_i / \mu_i$, and G^{-1} is the normalization constant

N-Node Open Jackson Networks – Joint Probability Mass Function – Single Server Nodes

- **Assume single server nodes – i.e. $S_i = 1 \ \forall$ i = 1, 2, …, N**
- **Therefore,** $d(j) = 1; \quad j = 1,2,\ldots,N$
- **The joint PMF is given by**

$$P(k_1,k_2,\ldots,k_N) = G^{-1}\prod_{i=1}^{N} R_i^{k_i}; \quad \forall k_1,k_2,\ldots,k_N \geq 0$$

- **Hence, the normalization constant is given by**

$$G^{-1} = \prod_{i=1}^{N}(1-R_i)$$

- **Rewriting the PMF results in**

$$P(k_1,k_2,\ldots,k_N) = \prod_{i=1}^{N}(1-R_i)R_i^{k_i}; \quad \forall k_1,k_2,\ldots,k_N \geq 0$$

N-Node Open Jackson Networks – Joint Probability Mass Function – Infinite Server Nodes

- **Assume infinite server nodes – i.e. $S_i = \infty$ $\forall$ i = 1, 2, …, N**
- **Therefore,** $d(j) = j; \quad j = 1,2,\ldots,N$
  and $\prod_{j=1}^{k_i} d(j) = k_i!$
- **The joint PMF is given by**
$$P(k_1,k_2,\ldots,k_N) = G^{-1}\prod_{i=1}^{N}\frac{R_i^{k_i}}{k_i!}; \quad \forall k_1,k_2,\ldots,k_N \ge 0$$
- **Hence, the normalization constant is given by**
$$G^{-1} = \prod_{i=1}^{N} e^{-R_i}$$
- **Rewriting the joint PMF results in**
$$P(k_1,k_2,\ldots,k_N) = \prod_{i=1}^{N}\frac{e^{-R_i}}{k_i!}R_i^{k_i}; \quad \forall k_1,k_2,\ldots,k_N \ge 0$$

N-Node Open Jackson Networks – Performance Calculations

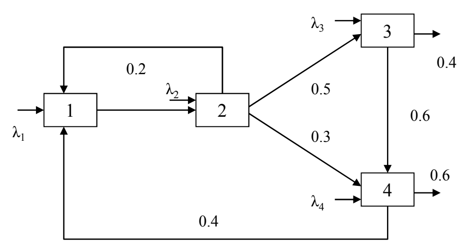
- **The marginal probability of node i having k_i messages is given by**
$$P(Q_i = k_i) = (1 - R_i^{k_i})R_i^{k_i}; \quad i = 1,2,\ldots,N$$
- **If the nodes in the system have limited buffer size M, then probability of buffer overflow may be approximated by**
$$P(Q_i \ge M) = \sum_{k_i=M}^{\infty}(1 - R_i^{k_i})R_i^{k_i} = R_i^{M}; \quad i = 1,2,\ldots,N$$
- **The mean and variance of total number of customers in N nodes is given by**
$$E[k_1 + k_2 + \ldots + k_N] = \sum_{i=1}^{N}\frac{R_i}{1-R_i}$$
$$Var[k_1 + k_2 + \ldots + k_N] = \sum_{i=1}^{N}\frac{R_i}{(1-R_i)^2}$$

- Prove these two equations?
- For large number of nodes, one may invoke the central-limit theorem to approximate the distribution of the total number of customers in the system by a Gaussian distribution

Example: N-Node Open Jackson Networks

- **<u>Problem</u>: Consider the network of queues depicted in figure. Assume λ = [2.0, 1.0, 0.5, 0.3] and μ = [0.1, 0.07, 0.03, 0.075]**

a) **Find the routing matrix**
b) **Calculate the total traffic flow vector, and the resulting loads at each queue node**
c) **Approximate the probability mass function of the total number of customers in the system using the Gaussian distribution**
d) **Find the exact probability mass function of the total number of customers and compare it to the one obtained in part (c).**



Example: N-Node Open Jackson Networks – cont'd

- **<u>Solution</u>:**

a) **The routing matrix, Q, is given by**

$$Q = \begin{bmatrix} 0 & 1.0 & 0 & 0 \\ 0.2 & 0 & 0.5 & 0.3 \\ 0 & 0 & 0 & 0.6 \\ 0.4 & 0 & 0 & 0 \end{bmatrix}$$

b) **Therefore the total traffic flow is given by and the loads are**

$$\Lambda = \lambda[I - Q]^{-1} = [4.7857 \quad 5.7857 \quad 3.3929 \quad 4.0714]$$

$$R = [0.4786 \quad 0.4050 \quad 0.1018 \quad 0.3054]$$

c) **The mean total messages in system is given by while the variance is given**

$$E[k_1 + k_2 + k_3 + k_4] = m = \sum_{i=1}^{4} \frac{R_i}{1 - R_i} = 2.1514$$

$$Var[k_1 + k_2 + k_3 + k_4] = \sum_{i=1}^{4} \frac{R_i}{(1 - R_i)^2} = 3.6632$$

or the standard deviation is $\sigma = \sqrt{\sum_{i=1}^{4} \frac{R_i}{(1 - R_i)^2}} = 1.9139$

Closed Jackson Networks

- **Closed: fixed number of messages circulate within the network with neither arrivals to nor departures from the network**
- **Classic application – computer system**
  - **Over a short period it can be assumed that tasks/processes/customers neither enter nor leave the system**

Closed Jackson Networks – Traffic Equation

- **Since there are no external arrivals, the traffic equation reduces to**

$$\Lambda_i = \sum_{j=1}^{N} q_{ji}\Lambda_j$$

$$\begin{bmatrix}\Lambda_1 & \Lambda_2 & \cdots & \Lambda_N\end{bmatrix} = \begin{bmatrix}\Lambda_1 & \Lambda_2 & \cdots & \Lambda_N\end{bmatrix}\begin{bmatrix}0 & q_{12} & \cdots & q_{1N}\\ q_{21} & 0 & \cdots & q_{2N}\\ \vdots & & 0 & \\ q_{N1} & q_{N2} & \cdots & 0\end{bmatrix}$$

$$\Lambda = \Lambda Q$$

$$\Lambda^T = Q^T \Lambda^T$$

Note that Λ is the transpose of the eigenvector for the matrix Q^T corresponding to the eigenvalue 1!!

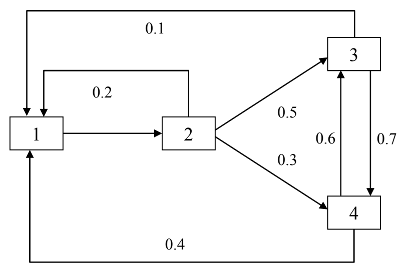
For a nice and very breif introduction to eigenvalues and eigenvectors please refer to http://www.sosmath.com/matrix/eigen0/eigen0.html or
http://maths.ucd.ie/courses/math1200/algebra/algebranotes7-1.pdf
In Matlab do “help eig” to get help regarding eigenvalues/vectors calculations

Example: Traffic Equation for Closed Networks

- **<u>Problem</u>: For the closed network shown in figure,**

a) Find the routing matrix, Q?

b) Compute the total flows into each node?

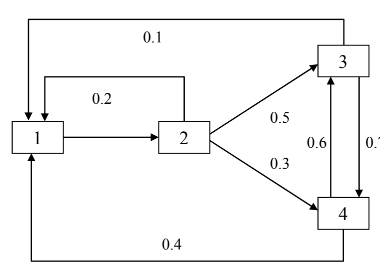


Example: Traffic Equation for Closed Networks

- **<u>Solution</u>:**

A) The routing matrix, Q:

$$Q = \begin{bmatrix} 0 & 1.0 & 0 & 0 \\ 0.2 & 0 & 0.5 & 0.3 \\ 0.1 & 0.2 & 0 & 0.7 \\ 0.4 & 0 & 0.6 & 0 \end{bmatrix}$$



Example: Traffic Equation for Closed Networks

- **Solution:**

B) The eigenvectos/values are calculated as shown

Hence, the *relative* flows are

Λ = [1.0 1.31 1.53 1.46]

```
>> [Vectors, Values] = eig(Q');
>> Vectors

Vectors =

   0.3728             0.0490 - 0.2601i   0.0490 + 0.2601i  -0.2709
   0.4870            -0.7672            -0.7672             0.5362
   0.5710             0.2160 + 0.1956i   0.2160 - 0.1956i  -0.6822
   0.5458             0.5022 + 0.0645i   0.5022 - 0.0645i   0.4169

>> Values

Values =

   1.0000                  0                  0                  0
        0            -0.1202 + 0.2880i        0                  0
        0                  0            -0.1202 - 0.2880i        0
        0                  0                  0            -0.7596

>> Vectors(:,1)./Vectors(1,1)

ans =

    1.0000
    1.3063
    1.5315
    1.4640
```


Closed Jackson Networks

- **Same assumptions as before**
  - **Exponential and independent service time**
  - **Si servers at node i**
- **K – total number of customers**
- **An easy extension to the equations derived for open networks**
- **It can be shown (following the same derivation process as that for open networks), the joint pmf is given by**

$$P(k_1, k_2, \ldots, k_N) = G(K,N)^{-1} \prod_{i=1}^{N} \left[R_i^{k_i} \Big/ \prod_{j=1}^{k_i} d(j) \right]$$

$$= \begin{cases} G(K,N)^{-1} \prod_{i=1}^{N} R_i^{k_i} & \text{single server nodes} \\ G(K,N)^{-1} \dfrac{\prod_{i=1}^{N} R_i^{k_i}}{k_i!} & \text{infinite server nodes} \end{cases}$$

The normalization constant G(K,N) must be computed for complete determination of the joint pmf!!

where Λ_1, Λ_2, …, Λ_N, is the solution to the traffic equation. $R_i = \Lambda_i/\mu_i$ and G(K,N) is a the normalization constant

Convolution Algorithm

- **How to calculate the normalization constant?**
- **Exhaustive method: find all (k_1, k_2, …, k_N) such that Σk_i = K – substitute in joint pmf and compute the constant G(K,N) such that the sum is equal to 1.**
  - **There are (N+K-1)!/(K!(N-1)!) ways - e.g. N = 4, K = 7 → 120 combinations!!**
  - **Prohibitive!!**
- **Use convolution algorithms**

Convolution Algorithm – Buzen – Simplified Version

- **Single server nodes → service rate is always μ – does not depend on number of customers at node**
- **Define S(k,n) = {k_1, k_2, …, k_n / Σk_i = k; 0≤k≤K; 1≤n≤N}**
- **Define G(k,n) by summing over the set S(k,n)**

$$G(k,n) = \sum_{S(k,n)} \prod_{i=1}^{n} R_i^{k_i}$$

- **G(k, n) is the sum over all possible ways of dispersing k messages among n nodes.**

Convolution Algorithm – Buzen – Simplified Version – cont'd

- **How to compute G(k,n)? – consider splitting the summation into**
  - **kn = 0**
  - **kn > 0**
- **Therefore we can write G(k,n) as**

$$G(k,n) = \sum_{\substack{S(k,n) \\ k_n=0}} \prod_{i=1}^{n} R_i^{k_i} + \sum_{\substack{S(k,n) \\ k_n>0}} \prod_{i=1}^{n} R_i^{k_i}$$

- **But the first summation is just the sum over the first n-1 nodes since the nth node is empty. i.e. G(k,n-1)**
- **For the second summation – there is at least one message in node n – i.e. there are at most k-1 *other* messages in the total network. i.e. G(k-1,n)**
- **Hence, the G(k,n) can be written as**

$$G(k,n) = G(k,n-1) + R_n G(k-1,n)$$

Convolution Algorithm – Buzen – Simplified Version – cont'd

- **We can show that**

$$G(k,n) = G(k,n-1) + R_n G(k-1,n)$$

- **The initiating values:**

$$G(k,1) = R_1^k ; \quad k = 1,2,\ldots,K$$

$$G(0,n) = 1; \quad n \geq 1$$

- **What is G(1,n) equal to for n >0?**

Example: Convolution Algorithm

- **Problem**: Assume a four node network with the routing matrix Q

  Assume μ = [2.5 2.5 2.5 2.5] and finite population of K = 7.

  A) Find the relative total flows

  B) Compute the joint distribution $P(k_1, k_2, k_3, k_4)$

$$Q = \begin{bmatrix} 0 & 0.75 & 0.25 & 0 \\ 0.05 & 0 & 0.15 & 0.8 \\ 0.25 & 0.25 & 0 & 0.5 \\ 0.4 & 0.35 & 0.25 & 0 \end{bmatrix}$$

Example: Convolution Algorithm

- **Solution**: **Closed Network K = 7, N = 4**

  A) Relative flows (found in the same manner as previous example

  Λ = [1.0000 1.5844 0.9195 1.7273]

  B) To compute the joint distribution $P(k_1, k_2, k_3, k_4)$, we need to compute:

  R = Λ./μ

  = [0.4000 0.6338 0.3678 0.6909]

  Note R is the RELATIVE loading

  We also need to compute G(K,N) using Buzen's convolution algorithm

Example: Convolution Algorithm – cont'd

- **<u>Solution</u>: cont'd**

  Using the recursive algorithm (Refer to Matlab code) – G(7,4) = 1.7036

  Therefore the joint pmf is equal to

$$
\begin{aligned}
P(k_1, k_2, \ldots, k_N) &= \prod_{i=1}^{N} R_i^{k_i} \Big/ G(K, N) \\
&= (0.4)^{k_1}(0.6338)^{k_2}(0.0.3678)^{k_3}(0.0.6910)^{k_4} \Big/ 1.7036
\end{aligned}
$$

Example: Convolution Algorithm – cont'd

- **<u>Solution</u>: cont'd**

The following code implements the recursive alogorithm:

```
%
% Example 4.8

K = 7;
N = 4;
M = 2.5*ones(1,N);

Q = [0    0.75  0.25  0; ...
     0.05 0     0.15  0.8; ...
     0.25 0.25  0     0.5; ...
     0.4  0.35  0.25  0];

[Vectors, Values] = eig(Q');
RFlows = Vectors(:,1)'./Vectors(1,1); % relative flows
RR     = RFlows./M; % compute relative loads

ks = 0:K;
ns = 1:N;
G_K_N = zeros(K+1,N);
%
% fill initial values
G_K_N(1,:) = ones(1,N);
G_K_N(:,1) = RR(1).^ks';
%
% fill the remaining of the matrix
for n=2:N
    for k=1:K
        G_K_N(k+1, n) = G_K_N(k+1, n-1) + RR(n)*G_K_N(k, n);
    end
end
```

<u>Program output:</u>

```
>> RFlows

RFlows =

    1.0000    1.5844    0.9195    1.7273

>> RR

RR =

    0.4000    0.6338    0.3678    0.6909

>> G_K_N

G_K_N =

    1.0000    1.0000    1.0000    1.0000
    0.4000    1.0338    1.4016    2.0925
    0.1600    0.8152    1.3306    2.7764
    0.0640    0.5806    1.0700    2.9882
    0.0256    0.3936    0.7871    2.8517
    0.0102    0.2597    0.5492    2.5195
    0.0041    0.1687    0.3707    2.1114
    0.0016    0.1085    0.2449    1.7036

>>
```