Convolution Algorithm

Raj Jain
Washington University in Saint Louis
Saint Louis, MO 63130
Jain@cse.wustl.edu

Audio/Video recordings of this lecture are available at:

http://www.cse.wustl.edu/~jain/cse567-08/

1. Distribution of Jobs in a system
2. Computing the normalizing constant G(N)
3. Computing performance using G(N)
4. Modeling timesharing systems

Convolution Algorithm

- Mean value analysis (MVA) provides only average queue lengths and response times.
- Convolution Algorithm provides more detailed information, e.g., Distribution or variance of queue lengths and response times.
  - What is the probability of two disks being busy at the same time?
  - What is the probability is of having more than five jobs at a disk?

Distribution of Jobs in A System

- Assume load-independent servers.
  (Terminals are considered later.)
- At any instant, the state of such a network

$$\mathbf{n} = \{n_1, n_2, \ldots, n_M\}$$

- Gordon and Newell (1967) showed that the probability of the system being in state **n** is:

$$P(n_1, n_2, \ldots, n_M) = \frac{D_1^{n_1} D_2^{n_2} \cdots D_M^{n_M}}{G(N)}$$

- Here, D_i is the total service demand per job for the *i*-th device, $N = \sum n_i$ is the total number of jobs in the system, and *G(N)* is a normalizing constant such that the probabilities for all states add up to one.

Distribution of Jobs in A System (Cont)

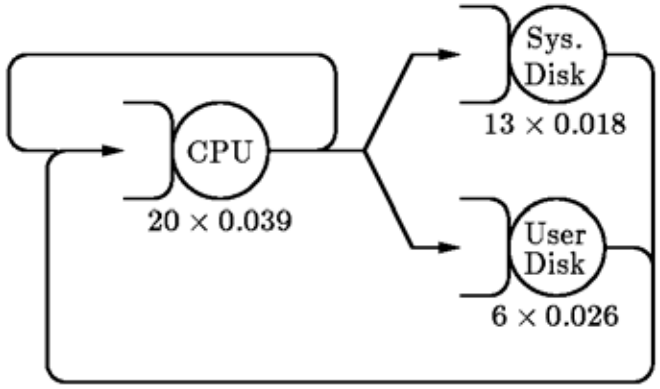
$$G(N) = \sum_{\mathbf{n}} (D_1^{n_1} D_2^{n_2} \cdots D_M^{n_M})$$

- *G(N)* can become too large or too small causing overflow/underflow.
- Avoided by scaling all service demands D_i's by α before computing *G(N)*.
- Let $y_i = \alpha D_i$. Then, $P(n_1, n_2, \ldots, n_M) = \dfrac{y_1^{n_1} y_2^{n_2} \cdots y_M^{n_M}}{G(N)}$

$$G(N) = \sum_{\mathbf{n}} (y_1^{n_1} y_2^{n_2} \cdots y_M^{n_M})$$

- One possible choice for α would be: $\alpha = \dfrac{1}{\frac{1}{M}\sum_{i=1}^{M} D_i}$
- In any case, the choice of α should not affect the probabilities.

Example 35.1



- Consider a batch computer system consisting of a processor, and two disks.
- The service times are 39 milliseconds per visit to CPU, 180 milliseconds per visit to disk A, and 260 milliseconds to disk B.
- Each batch job makes 13 I/O requests to disk A, and 6 I/O requests to the disk B.
- Compute the state probabilities when the multiprogramming level is 3.

Example 35.1 (Cont)

- The service times are: $S_{CPU}=0.039$, $S_A=0.18$, $S_B=0.26$
- The visit ratios are: $V_{CPU}=13+6+1=20$, $V_A=13$, $V_B=6$
- Total service demands are:

$$D_{CPU} = 20 \times 0.039 = 0.78$$
$$D_A = 13 \times 0.18 = 2.34$$
$$D_B = 6 \times 0.26 = 1.56$$

- For the scaling factor, we arbitrarily choose $\alpha=1/0.78$.
- This results in $y_{CPU}=1$, $y_A=3$, *and* $y_B=2$.

Example 35.1 (Cont)

- The system can be in any one of the ten states.
- For each state, we first determine the product $\prod_i y_i^{n_i}$, add all the products to find *G(N)*, and then divide each individual product by *G(N)* to get the desired probability.

Number of Jobs at CPU	Number of Jobs at B Disk	Number of Jobs at A Disk	Numerator $\prod_i y_i^{n_i}$	Probability
0	0	3	8	0.089
0	1	2	12	0.133
0	2	1	18	0.200
0	3	0	27	0.300
1	0	2	4	0.044
1	1	1	6	0.067
1	2	0	9	0.100
2	0	1	2	0.022
2	1	0	3	0.033
3	0	0	1	0.011
		$\sum$	$G(N) = 90$	1.000

Convolution Algorithm for Computing G(N)

- Gordon and Newell's method requires us to enumerate all possible states.
- For a system with N jobs and M devices, the number of states is given by:

$$\begin{pmatrix} N+M-1 \\ M-1 \end{pmatrix}$$

- This number is of the order of N^{M-1} and is usually very large.

Convolution Algorithm (Cont)

- Buzen (1973)'s *convolution* method is based on the following mathematical identity, which is true for all k and y_i's:

$$\sum_{\mathbf{n}} \prod_{i=1}^{k} (y_i)^{n_i} = \sum_{\mathbf{n}|n_k=0} \prod_{i=1}^{k} (y_i)^{n_i} + \sum_{\mathbf{n}|n_k>0} \prod_{i=1}^{k} (y_i)^{n_i}$$

$$= \sum_{\mathbf{n}|n_k=0} \prod_{i=1}^{k} (y_i)^{n_i} + y_k \sum_{\mathbf{n}^-} \prod_{i=1}^{k} (y_i)^{n_i}$$

- Here, ***n*** is the set of all possible state vectors $\{n_1, n_2, \ldots, n_k\}$ such that $\sum_{i=1}^{k} n_i = n$; and n^- is the set of all possible state vectors such that $\sum_i n_i = n - 1$

Convolution Algorithm (Cont)

- First break the sum into two sums -- a sum belonging to states in which n_k is zero and a sum belonging to the remaining states.
- Then the second sum is rewritten so that y_k, which is common to all terms in the second sum, is factored out.

Example 35.2

- Consider the central server model of Example 35.1.
  To compute *G(N)* we need to compute the following sum:

$$G(N) = y_1^0y_2^0y_3^3 + y_1^0y_2^1y_3^2 + y_1^0y_2^2y_3^1 + y_1^0y_2^3y_3^0 + y_1^1y_2^0y_3^2 + y_1^1y_2^1y_3^1 + y_1^1y_2^2y_3^0 + y_1^1y_2^0y_3^1 + y_1^2y_2^1y_3^0 + y_1^3y_2^0y_3^0$$

- The powers of y_i's in the above sum correspond to the ten states (specified by number of jobs at various devices)

$$G(N) = (y_1^0y_2^3 + y_1^1y_2^2 + y_1^2y_2^1 + y_1^3y_2^0) + y_3(y_1^0y_2^0y_3^2 + y_1^0y_2^1y_3^1 + y_1^0y_2^2y_3^0 + y_1^1y_2^0y_3^1 + y_1^1y_2^1y_3^0 + y_1^1y_2^0y_3^0)$$

- By separating out the terms containing y_3^0, the above sum can be rewritten as follows:

$$G(N) = (y_1^0y_2^3y_3^0 + y_1^1y_2^2y_3^0 + y_1^2y_2^1y_3^0 + y_1^3y_2^0y_3^0) + (y_1^0y_2^0y_3^3 + y_1^0y_2^1y_3^2 + y_1^0y_2^2y_3^1 + y_1^1y_2^0y_3^2 + y_1^1y_2^1y_3^1 + y_1^1y_2^0y_3^1)$$

Example 35.2 (Cont)

- Since y_3 is common to all terms in the second part, the sum can also be written as:

$$G(N) = (y_1^0 y_2^3 + y_1^1 y_2^2 + y_1^2 y_2^1 + y_1^3 y_2^0) + y_3(y_1^0 y_2^0 y_3^2 + y_1^0 y_2^1 y_3^1 + y_1^0 y_2^2 y_3^0 + y_1^1 y_2^0 y_3^1 + y_1^1 y_2^1 y_3^0 + y_1^1 y_2^0 y_3^0)$$

- Notice that the first part contains terms corresponding to the states of a system with one less device (there is no y_3).
- The terms inside the paranthesis in the second part correspond to the states of a system with one less user (two users instead of three).
- This allows the problem of determining *G(N)* to be divided into two smaller problems.

Convolution Algorithm (Cont)

- If we define *g(n, k)* as follows: $g(n,k) = \sum_{\mathbf{n}} \prod_{i=1}^{k} y_i^{n_i}$
- Then,

$$g(n,k) = g(n,k-1) + y_k g(n-1,k)$$

- The initial values for the auxiliary function are:

$$\begin{aligned} g(n,0) &= 0 \quad n = 1,2,\ldots,N \\ g(0,k) &= 1 \quad k = 1,2,\ldots,M \end{aligned}$$

- This results in a simple algorithm for computing *g(N, M)*.

Convolution Algorithm: Tabular Format

	y_1	y_2	$\cdots$	y_k	$\cdots$	y_M
0	1	1	$\cdots$	1	$\cdots$	1
1	y_1					
2	y_1^2					
3	y_1^3					
$\vdots$	$\vdots$			$g(n-1,k)$		
				$\downarrow \times y_k$		
n	y_1^n		$g(n,k-1) \rightarrow$	$g(n,k)$		
$\vdots$	$\vdots$					
N	y_1^N					$g(N,M)$

- The (n, k)th entry in the table is g(n, k).
- It is is obtained by adding together the value immediately to its left and the value immediately above multiplied by the corresponding column variable y_k.

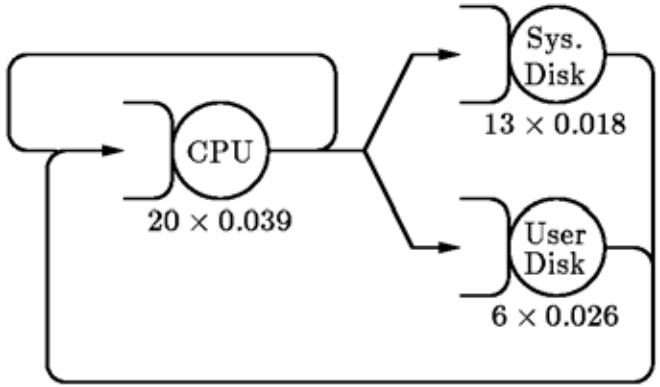
Convolution Algorithm (Cont)

- It is sufficient to store just one column *{G[0], G[1], …, G[N]}* and use the following algorithm:

$$\text{FOR k} \leftarrow 1 \text{ TO M DO}$$
$$\text{FOR n} \leftarrow 1 \text{ TO N DO}$$
$$\text{G[n]} \leftarrow \text{G[n]} + \text{y[k]} \times \text{G[n-1]};$$

Example 35.3

- Consider the central server model of Example 35.1 again.



n	y_{CPU} =1	y_A =3	y_B =2
0	1	1	1
1	1	4	6
2	1	13	25
3	1	40	90

Computing Performance Using G(N)

- Queue Length Distributions:

$$P(n_i \geq j) = \sum_{\mathbf{n}|n_i \geq j} \frac{y_1^{n_1} y_2^{n_2} \cdots y_M^{n_M}}{G(N)} = y_i^j \frac{G(N-j)}{G(N)}$$

$$\begin{aligned} P(n_i = j) &= P(n_i \geq j) - P(n_i \geq j+1) \\ &= \frac{y_i^j}{G(N)} \left(G(N-j) - y_i G(N-j-1)\right) \end{aligned}$$

- Mean number of jobs at the *i*th device $E[n_i]$ is:

$$Q_i = E[n_i] = \sum_{j=1}^{N} P(n_i \geq j) = \sum_{j=1}^{N} y_i^j \frac{G(N-j)}{G(N)}$$

Computing Performance Using G(N) (Cont)

- The joint probability of having *j* or more jobs at *i*th device and l or more jobs at *k*th device is:

$$P(n_i \geq j, n_k \geq l) = y_i^j y_k^l \frac{G(N-j-l)}{G(N)}$$

- Utilizations: The device utilization is the same as the probability of having one or more jobs at the device:

$$U_i = P(n_i \geq 1) = y_i \frac{G(N-1)}{G(N)}$$

- Throughputs: From utilization law:

$$X_i = \frac{U_i}{S_i}$$

Computing Performance Using G(N) (Cont)

- The system throughput is given by the forced flow law:

$$X = \frac{X_i}{V_i} = \frac{U_i}{D_i} = \alpha \frac{G(N-1)}{G(N)}$$

- Response Times: The mean response time of a device is given by Little's law:

$$R_i = \frac{Q_i}{X_i} = \frac{Q_i}{XV_i}$$

- The mean system response time is given by the general response time law:

$$R = \sum_{i=1}^{M} R_i V_i$$

Example 35.4

- Consider the central server model of Example 35.1
- For this system: *G(1)=6, G(2)=25, and G(3)=90.*
- Queue Length Distributions:
- The probability of having two or more jobs at disk A is:

$$P(n_A \geq 2) = y_A^2 \frac{G(N-2)}{G(N)} = 3^2 \frac{6}{90} = 0.6$$

- The probability of exactly one job at disk A is:

$$\begin{aligned} P(n_A = 1) &= \frac{y_A}{G(N)} (G(N-1) - y_A G(N-2)) \\ &= \frac{3}{90}(25 - 3 \times 6) = \frac{21}{90} = 0.233 \end{aligned}$$

Example 35.4 (Cont)

- The probability of exactly zero, two, and three jobs:

$$P(n_A = 0) = \frac{y_A^0}{G(N)}(G(N-0) - y_A G(N-1))$$

$$= \frac{90 - 3 \times 25}{90} = 0.166$$

$$P(n_A = 2) = \frac{y_A^2}{G(N)}(G(N-2) - y_A G(N-3))$$

$$= 3^2 \frac{6 - 3 \times 1}{90} = 0.3$$

$$P(n_A = 3) = \frac{y_A^3}{G(N)}(G(N-3) - y_A G(N-4))$$

$$= 3^3 \frac{1 - 3 \times 0}{90} = 0.3$$

Example 35.4 (Cont)

- We can compute its mean, variance, and any higher-order statistics of queue length at disk A:

$$\begin{aligned} E[n_A] &= \sum_{j=1}^{N} jP(n_A = j) \\ &= 1 \times 0.233 + 2 \times 0.3 + 3 \times 0.3 = 1.733 \\ E[n_A^2] &= \sum_{j=1}^{N} j^2 P(n_A = j) \\ &= 1^2 \times 0.233 + 2^2 \times 0.3 + 3^2 \times 0.3 = 4.133 \\ \text{Var}[n_A] &= E[n_A^2] - (E[n_A])^2 \\ &= 4.133 - 1.733^2 = 1.13 \end{aligned}$$

Example 35.4 (Cont)

- The probability that both disks are busy:

$$P(n_A \geq 1, n_B \geq 1) = y_A^1 y_B^1 \frac{G(N-2)}{G(N)} = \frac{3 \times 2 \times 6}{90} = \frac{36}{90} = 0.4$$

- The mean queue lengths at the three service centers are:

$$Q_{CPU} = \sum_{j=1}^{N} y_{CPU}^j \frac{G(N-j)}{G(N)} = \frac{1 \times 25 + 1^2 \times 6 + 1^3 \times 1}{90} = \frac{25+6+1}{90} = \frac{32}{90} = 0.356$$

$$Q_A = \sum_{j=1}^{N} y_A^j \frac{G(N-j)}{G(N)} = \frac{3 \times 25 + 3^2 \times 6 + 3^3 \times 1}{90} = \frac{75+54+27}{90} = \frac{156}{90} = 1.733$$

$$Q_B = \sum_{j=1}^{N} y_B^j \frac{G(N-j)}{G(N)} = \frac{2 \times 25 + 2^2 \times 6 + 2^3 \times 1}{90} = \frac{50+24+8}{90} = \frac{82}{90} = 0.911$$

Example 35.4 (Cont)

- As a check, note that the three queue lengths sum up to 3 which is the total number of jobs in the system.
- Utilizations: The CPU utilization is:

$$U_{CPU} = P(n_{CPU} \geq 1) = y_{CPU}\frac{G(N-1)}{G(N)} = \frac{25}{90} = 0.278$$

- Throughputs: The system throughput is:

$$X = \alpha\frac{G(N-1)}{G(N)} = \frac{1}{0.78} \times \frac{25}{90} = 0.356 \text{ jobs/second}$$

- The CPU throughput is:

$$X_{CPU} = XV_{CPU} = 0.356 \times 20 = 7.12 \text{ jobs/second}$$

Example 35.4 (Cont)

- Response Times:

$$R_{CPU} = \frac{Q_{CPU}}{XV_{CPU}} = \frac{0.356}{0.356 \times 20} = 0.05 \text{ seconds}$$

$$R_A = \frac{Q_A}{XV_A} = \frac{1.733}{0.356 \times 13} = 0.37 \text{ seconds}$$

$$R_B = \frac{Q_B}{XV_B} = \frac{0.911}{0.356 \times 6} = 0.43 \text{ seconds}$$

Example 35.4 (Cont)

- The mean system response time as given by the general response time law is:

$$R = \sum_i R_i V_i = 0.05 \times 20 + 0.37 \times 13 + 0.43 \times 6 = 8.42 \text{ sec}$$

- As a check, we can use Little's law to find the mean number of jobs in the system:

$$N = XR = 0.356 \times 8.42 = 3$$

Timesharing Systems

- One set of terminals that are modeled as delay centers with think-time Z.
- Assume that the terminals are zeroth device and:

$$g(n,0) = \frac{y_0^n}{n!} \quad n = 1, 2, \ldots, N$$

- Here, $y_0 = \alpha Z$ is the scaled value of the think time.
- The system state is now represented by a M+1 component vector $\boldsymbol{n} = \{n_0, n_1, \ldots, n_M\}$, where n_i is the number of jobs at the ith device, n_0 is the number of jobs at the terminals.

$$n_i \geq 0, \qquad \sum_{i=0}^{M} n_i = N$$

Timesharing Systems (Cont)

- The state probabilities are given by:

$$P(n_0, n_1, \ldots, n_M) = \frac{y_0^{n_0} y_1^{n_1} y_2^{n_2} \cdots y_M^{n_M}}{n_0! G(N)}$$

- If there is more than one delay center in the network, they can all be combined into one service center by adding their total demands D_i to Z.
- The computed value of *G(N)* is still correct and n_0 now represents the sum of queue lengths at all delay centers.
- The mean queue lengths at individual delay centers are proportional to their service demands D_i.
- The equations for system performance metrics given earlier are still valid.

Timesharing Systems (Cont)

- The queue length distribution formulas apply to devices other than terminals.
- See Buzen 1973 for queue length distributions for terminals
- The mean queue lengths and device utilizations for the terminals:
- Mean Queue Length: Using Little's law

$$Q_0 = X\,Z$$

- Utilizations: Using the utilization law

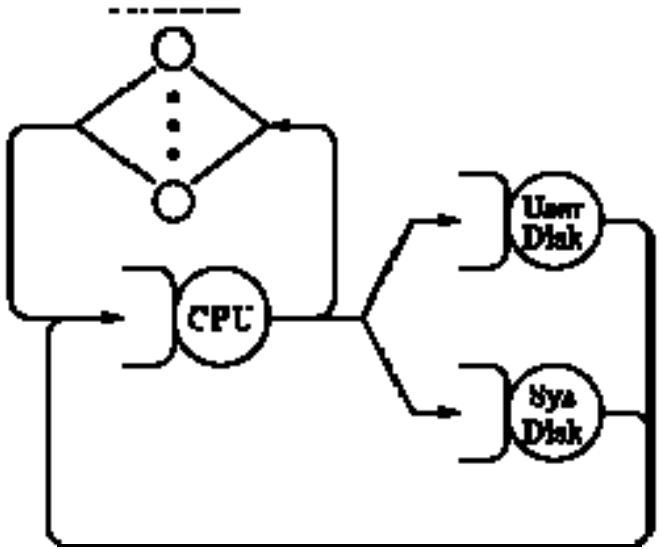
$$U_0 = X\,Z$$

- Utilization of individual terminals is:

$$U_Z = \frac{U_0}{N} = \frac{X\,Z}{N} = \frac{Z}{R+Z}$$

Example 35.5

- Consider a timesharing system with 3 terminals. An average user makes 13 I/O requests to disk A, and six I/O requests to disk B. The service times per visit to the CPU, disk A, and disk B are 39 milliseconds, 180 milliseconds, and 260 milliseconds, respectively. The users have an average think time of 4.68 seconds.
- For this system:
- The service times are:
  $S_{CPU}=0.039$, $S_A=0.18$, $S_B=0.26$
- The visit ratios are:
  $V_{CPU}=13+6+1=20$, $V_A=13$, $V_B=6$



Example 35.5 (Cont)

- Total service demands are:

$$D_{CPU} = 20 \times 0.039 = 0.78$$
$$D_A = 13 \times 0.18 = 2.34$$
$$D_B = 6 \times 0.26 = 1.56$$

- and $Z=4.68$
- For the scaling factor, we once again choose $\alpha=1/0.78$.
- This results in $y_0=6$, $y_{CPU}=1$, $y_A=3$, *and* $y_B=2$.
- The nth row of the terminal column is $y_0^n/n!$.

Example 35.5 (Cont)

n	y_0 $=6$	y_{CPU} $=1$	y_A $=3$	y_B $=2$
0	1	1	1	1
1	6	7	10	12
2	18	25	55	79
3	36	61	226	384

- *G(0)=1, G(1)=12, G(2)=79, G(3)=384.*
- The total number of system states is $\begin{pmatrix} 3+4-1 \\ 4-1 \end{pmatrix}$ or *20*.

Example 35.5 (Cont)

- Can compute any desired statistics on the queue length of any device including terminals.
- For example, the probability of two jobs at the terminals is *0.094 + 0.141 + 0.047 = 0.282.*
- Similarly, the probability of one job at disk A is *0.031+0.016+0.008+0.094+0.047+0.141=0.336.*

Number of Jobs at				Probability
Terminals	CPU	Disk A	Disk B	
0	0	0	3	0.021
0	0	1	2	0.031
0	0	2	1	0.047
0	0	3	0	0.070
0	1	0	2	0.010
0	1	1	1	0.016
0	1	2	0	0.023
0	2	0	1	0.005
0	2	1	0	0.008
0	3	0	0	0.003
1	0	0	2	0.063
1	0	1	1	0.094
1	0	2	0	0.141
1	1	0	1	0.031
1	1	1	0	0.047
1	2	0	0	0.016
2	0	0	1	0.094
2	0	1	0	0.141
2	1	0	0	0.047
3	0	0	0	0.094
				Total = 1.000

State Probabilities for Example

- Without the table, the probability of one job at disk A:

$$P(n_A \geq 1) = y_A^1 \frac{G(N-1)}{G(N)} = 3^1 \frac{79}{384} = 0.617$$

$$P(n_A \geq 2) = y_A^2 \frac{G(N-2)}{G(N)} = 3^2 \frac{12}{384} = 0.281$$

$$P(n_A = 1) = P(n_A \geq 1) - P(n_A \geq 2) = 0.617 - 0.281 = 0.336$$

- Similarly, probabilities of *0*, *2*, and *3* jobs at disk A can be shown to be *0.383*, *0.211*, and *0.070*.
- Using these values, the average queue length at disk A can be shown to be *0.97*
- Variance of the queue length is *0.87*.

Example 35.5 (Cont)

- The system throughput is:

$$X = \alpha \frac{G(N-1)}{G(N)} = \frac{1}{0.78} \times \frac{79}{384} = 0.264$$

- The device utilizations are:

$$U_{CPU} = XD_{CPU} = 0.264 \times 0.78 = 0.206$$

$$U_A = XD_A = 0.264 \times 2.34 = 0.618$$

$$U_B = XD_B = 0.264 \times 1.56 = 0.412$$

Example 35.5 (Cont)

- The average number of jobs at the devices are:

$$Q_{CPU} = \sum_{j}^{N} y_{CPU}^{j} \frac{G(N-j)}{G(N)} = \frac{1^1 \times 79 + 1^2 \times 12 + 1^3 \times 1}{384} = \frac{92}{384} = 0.240$$

$$Q_A = \sum_{j}^{N} y_A^j \frac{G(N-j)}{G(N)} = \frac{3^1 \times 79 + 3^2 \times 12 + 3^3 \times 1}{384} = \frac{372}{384} = 0.969$$

$$Q_B = \sum_{j}^{N} y_B^j \frac{G(N-j)}{G(N)} = \frac{2^1 \times 79 + 2^2 \times 12 + 2^3 \times 1}{384} = \frac{214}{384} = 0.557$$

$$Q_{term} = N - (Q_{CPU} + Q_A + Q_B) = 3 - (0.240 + 0.969 + 0.557) = 1.234$$

Example 35.5 (Cont)

- The device response times are:

$$R_{CPU} = Q_{CPU}/(XV_{CPU}) = \frac{0.240}{0.264 \times 20} = 0.045 \text{ seconds}$$

$$R_A = Q_A/(XV_A) = \frac{0.969}{0.264 \times 13} = 0.283 \text{ seconds}$$

$$R_B = Q_B/(XV_B) = \frac{0.557}{0.264 \times 6} = 0.352 \text{ seconds}$$

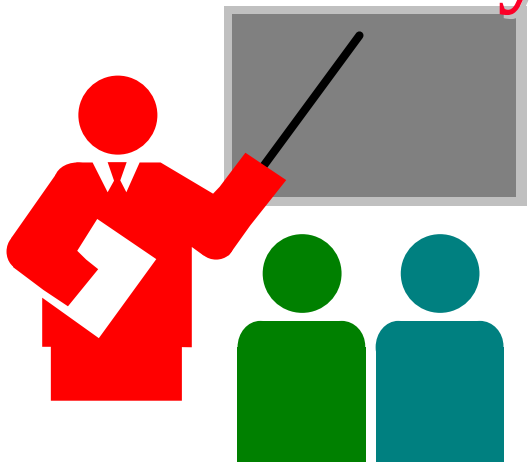
- The system response time is:

$$R = \sum_i R_i V_i = 0.045 \times 20 + 0.283 \times 13 + 0.352 \times 6 = 6.694 \text{ seconds}$$

- We can check the computation by computing the average number of jobs in the system:

$$N = X(R + Z) = 0.264(6.694 + 4.68) = 3$$

Summary



- Gordon and Newell: $$P(n_1, n_2, \ldots, n_M) = \frac{D_1^{n_1} D_2^{n_2} \cdots D_M^{n_M}}{G(N)}$$
- A system with *N* jobs and *M* devices has approx N^{M-1} states
- Convolution algorithm allows G(N) to be computed using *NM* computations

$$g(n,k) = g(n,k-1) + y_k g(n-1,k)$$

- Allows getting variance and other higher order statistics

Homework 35

For a timesharing system with two disks (user and system), the probabilities for jobs completing the service at the CPU were found to be 0.**75** to disk A, **0.15** to disk B, and **0.1** to the terminals. The user think time was measured to be 5 seconds, the disk service times are 30 milliseconds and 25 milliseconds, while the average service time per visit to the CPU was 40 milliseconds.

Analyze this system using convolution method and determine the distribution of queue lengths when there are 3 users on the system.