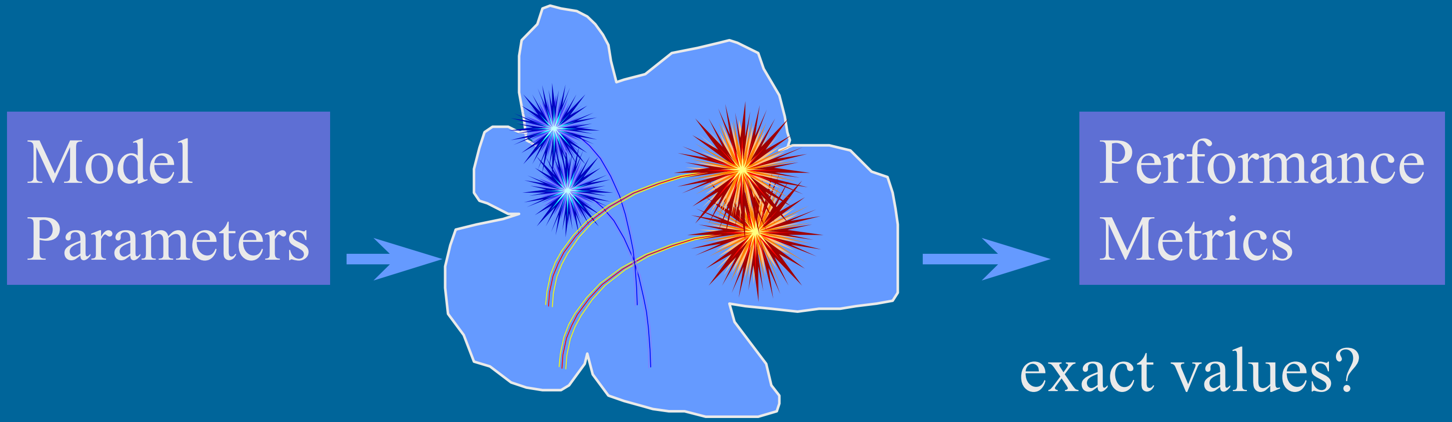
Lecture 7

Analytical Solution Method for Complex Models

Multiple Class Markov Chain Models

Convolution

Generic Solution Method



Model Parameters

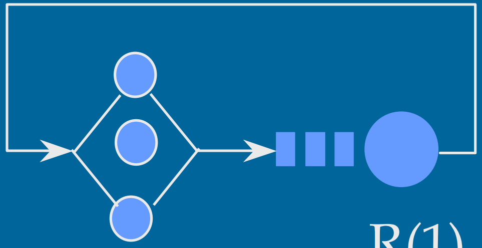
Performance Metrics

exact values?
estimates?
confidence intervals?
bounds?

analytical methods
simulation
    queueing network
    Petri net

Markov Chain Solution

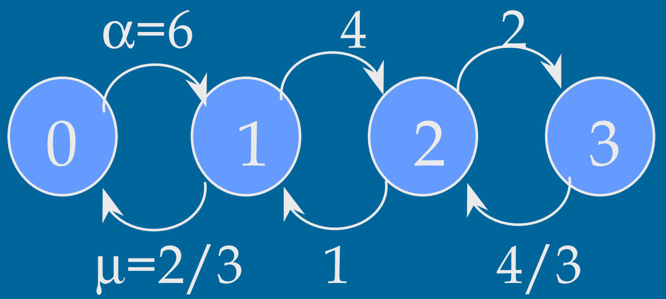
- Birth-Death process
- Stochastic process
- Large state space?
- Large normalizing constant?



M=3, Z = 0.5

U, R, X, N_{sys}?

R(1) = 1.5
R(2) = 1.0
R(3) = 0.75



Birth-Death Process

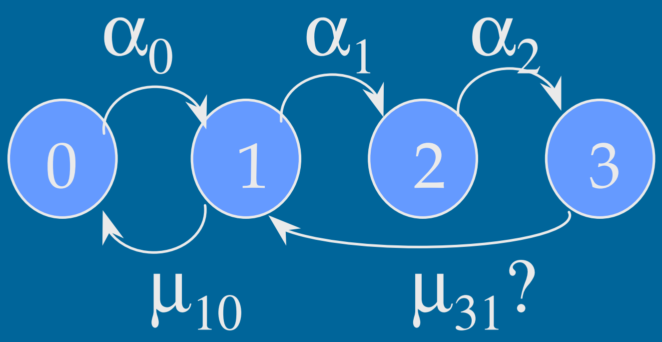
Prob = P = 0.01 0.09 0.36 0.54

U = 1-P0 = 0.99, $X = \Sigma\mu_i P_i$ = 1.14 tps

$N_{sys} = \Sigma iP_i$ = 2.43, R = N/X = 2.13 sec

Markov Chain Solution

- 1. State description
  - finite, infinite? state space?
  - multiple classes? multiple phases?
- 2. State enumeration
- 3. Transition rates



- Balance equations
  - flow in to state = flow out of state, $\Sigma P_i = 1$
- Solve balance equations
- Use P_i's to get performance metrics $U = 1 - P_0$

Markov Example

- Central server model, Fig. 5.1 [Men 94]

number of jobs in CPU

- State = (n_c, n_f, n_s), $\Sigma n_i = 2$
- State enumeration: $n_i \in \{0, 1, 2\}$

  S = { (2,0,0), (1,1,0), (1,0,1), (0,2,0), (0,1,1), (0,0,2) }
  - state space S can be large (<u>very</u> large)
  - K devices, max mpl=N

$$|S| = \binom{N-K-1}{K-1} = \frac{(N-K-1)!}{N!\ (K-1)!}$$

Markov Example (contd)

- State space diagram, Fig. 5.2
- Transition rates

cpu completes, request fast disk

fast disk completes

Rate((1,1,0) -> (0,2,0)) = μ_f = 3

Rate((0,2,0) -> (1,1,0)) = μ_c p = 6 * 0.5 = 3

Rate((2,0,0) -> (1,0,1)) = μ_c (1-p) = 6 * 0.5 = 3

Rate((1,0,1) -> (2,0,0)) = μ_s = 2

….

Markov Example (contd)

- Notation

P_{011} = Prob { in state (0,1,1) }

- Local balance Fig 5.3 [Men 94]
- Global balance equations

$\mu_c(1\text{-}p)P_{110} + \mu_c p P_{101} = (\mu_s + \mu_f)\, P_{011}$

…

"similarly for some other 4 states"

…

$\sum Pi = 1$

flow in = flow out

one global balance equation is redundant!

Markov Example (contd)

- Solve balance equations
  - – brute force approach
  - – 6 equations, 6 unknowns, OK
  - – 92378 equations, 92378 unknowns, ????
  - – not always practical,
  - – can be very time consuming
- Better method: transform set of equations into simpler form: local balance equations

Local Balance Equations

- Equations for local balanced transitions between neighboring states
- Each equation in terms of
  - relative device utilizations
  - normalizing constant
    - needs to be computed
  - can be tricky
    - can be time consuming
    - “the miracle occurs here”

from transition probabilities and service rates

Relative Utilization (2)

$$\mu_f P_{110} = \boxed{\mu_c p} P_{200}$$

$$\mu_s P_{101} = \mu_c (1 - p) P_{200}$$

$$\mu_f P_{020} = \mu_c p P_{110}$$

$$\mu_s P_{011} = \mu_c (1 - p) P_{110}$$

$$\mu_f P_{011} = \mu_c p P_{101}$$

$$\mu_s P_{002} = \mu_c (1 - p) P_{101}$$

$$\sum P_i = 1$$

?

Notation:
”Relative Utilization”

$$U_f = \frac{\mu_c p}{\mu_f}$$

$$U_s = \frac{\mu_c (1 - p)}{\mu_s}$$

$$U_c = 1$$

(relative to CPU,
serv time & util are
inversely relative to μ)

Modified Local Balance Equs (1)

$$\mu_f P_{110} = \mu_c p\, P_{200}$$
$$\mu_s P_{101} = \mu_c (1-p)\, P_{200}$$
$$\mu_f P_{020} = \mu_c p\, P_{110}$$
$$\mu_s P_{011} = \mu_c (1-p)\, P_{110}$$
$$\mu_f P_{011} = \mu_c p\, P_{101}$$
$$\mu_s P_{002} = \mu_c (1-p)\, P_{101}$$

$$P_{110} = U_f P_{200}$$
$$P_{101} = U_s P_{200}$$
$$P_{020} = U_f P_{110} = U_f^2 P_{200}$$
$$P_{011} = U_s P_{110} = U_s U_f P_{200}$$
$$P_{011} = U_f P_{101} = U_f U_s P_{200}$$
$$P_{002} = U_s P_{101} = U_s^2 P_{200}$$

$$U_f = \frac{\mu_c p}{\mu_f}$$

$$U_s = \frac{\mu_c (1-p)}{\mu_s} \qquad U_c = 1$$

Normalizing Constant (2)

$$P_{110} = U_f P_{200}$$

$$P_{101} = U_s P_{200}$$

$$P_{020} = U_f P_{110} = U_f^2 P_{200}$$

$$P_{011} = U_s P_{110} = U_s U_f P_{200}$$

$$P_{011} = U_f P_{101} = U_f U_s P_{200}$$

$$P_{002} = U_s P_{101} = U_s^2 P_{200}$$

= 1

$$P_{110} = \boxed{U_c^1}\, U_f^1 U_s^0 P_{200}$$

$$P_{101} = U_c^1 U_f^0 U_s^1 P_{200}$$

$$P_{020} = U_c^0 U_f^2 U_s^0 P_{200}$$

$$P_{011} = U_c^0 U_f^1 U_s^1 P_{200}$$

$$P_{002} = U_c^0 U_f^0 U_s^2 P_{200}$$

$$\sum P_i = 1$$

$$P_{200} = \frac{1}{\sum\limits_{\substack{t \in S \\ t=(i,j,k) \\ i+j+k=2}} U_c^i U_f^j U_s^k} = \frac{1}{G(2)}$$

normalizing constant

2 jobs

Local Balance Equations (contd)

$$P_{ijk}(2) = \frac{U_c^i U_f^j U_s^k}{G(2)}$$

when mpl = i+j+k=2

$$P_{ijk}(N) = \frac{U_c^i U_f^j U_s^k}{G(N)}$$

when mpl = i+j+k=N

$$G(N) = \sum_{\substack{t \in S \\ t=(i,j,k) \\ i+j+k=N}} U_c^i U_f^j U_s^k$$

product form

sum over all states (complex)

Performance Measures (1)

CPU utilization for mpl=2 ≠ CPU relative utilization

$$U_c(2) = \sum_{i\ge 1} P_{ijk} = P_{110} + P_{101} + P_{200}$$

$$= \frac{\sum_{\substack{(i,j,k)\\ \boxed{i\ge 1}\\ i+j+k=2}} U_c^i U_f^j U_s^k}{G(2)} = \frac{U_c \sum_{\substack{(i,j,k)\\ i+j+k=\boxed{1}}} U_c^i U_f^j U_s^k}{G(2)} = \frac{U_c G(1)}{G(2)}$$

$$U_c(N) = U_c \frac{G(N-1)}{G(N)}$$

Performance Measures (contd) (3)

$$U_c(N) = U_c \frac{G(N-1)}{G(N)}$$

$1/S_c$

$$X_c(N) \overset{Little}{=} U_c(N)\mu_c = \mu_c U_c \frac{G(N-1)}{G(N)}$$

$$\bar{n}_c(2) = \sum_{\substack{i,j,k\\ i>0}} iP_{ijk} = \sum_{\substack{i,j,k\\ i=1}} iP_{ijk} + \sum_{\substack{i,j,k\\ i=2}} iP_{ijk} \overset{homework}{=} U_c^1 \frac{G(1)}{G(2)} + U_c^2 \frac{G(0)}{G(2)} \quad \text{(7-1)}$$

In general, $$\bar{n}_c(N) = \sum_{m=1}^{N} U_c^m \frac{G(N-m)}{G(N)}$$

Performance Measures (contd) (3)

mean device response time:

$$R_c(N) \overset{Little}{=} \frac{\overline{n}_c(N)}{X_c(N)} = \frac{\sum_{m=1}^{N} U_c^{m-1} G(N-m)}{\mu_c G(N-1)}$$

mean device residence time:

$$R_c^{'}(N) = V_c R_c(N)$$

mean system response time:

$$R\ (N) = \sum_c R_c^{'}(N) = \sum_c V_c R_c(N)$$

mean system throughput:

$$X_0(N) \overset{FFL}{=} \frac{X_c(N)}{V_c(N)}$$

How to Compute Relative U_c's and G (2)

in steady state:

$$\underset{\text{vector}}{X_i}\;\underset{\text{matrix}}{P_{ij}} = \underset{\text{vector}}{X_i}$$

transition matrix

relative flow through each device

Solve for X, use Little to get U_c's:

$$U_c \overset{Little}{=} X_c S_c = \frac{X_c}{\mu_c}$$

(many solutions for relative X's)

And, finally get

$$G(N) = \sum_{\substack{t\in S\\ t=(i,j,k)\\ i+j+k=N}} U_c^i U_f^j U_s^k$$

Example (4)

(relative X_i, relative U_i)

$$(X_c, X_f, X_s)\begin{bmatrix}0 & p & 1-p\\ 1 & 0 & 0\\ 1 & 0 & 0\end{bmatrix} = (X_c, X_f, X_s)$$

$$X_f + X_s = X_c$$
$$pX_c = X_f$$
$$(1-p)X_c = X_s$$

$$X_c = \mu_c$$
$$X_f = \mu_c p$$
$$X_s = \mu_c(1-p)$$

$$U_c = X_c / \mu_c = 1$$
$$U_f = \mu_c p / \mu_f = 0.5\,\mu_c / \mu_f = 1$$
$$U_s = \mu_c(1-p) / \mu_s = 0.5\,\mu_c / \mu_s = 1.5$$

$$G(2) = \sum_{\substack{i,j,k\\ N=2}} 1^i 1^j 1.5^k = \underset{(1,0,1)}{1.5} + \underset{(0,1,1)}{1.5} + \underset{(0,0,2)}{2.25} = 5.25$$

State Space Explosion

K	N	$\|S\|$
3	2	$\binom{4}{2} = \frac{4!}{2!2!} = 6$
10	3	$\binom{12}{9} = 220$
3	10	$\binom{12}{2} = 66$
10	10	$\binom{19}{9} = 92378$

Convolution Algorithm

- How to compute G(N)?
  - problem: sum over all states? many states...
  - Buzen: Convolution algorithm

G(N)=g(N,K)

for population N
for K devices

$$g(2,3) = \sum_{\substack{t=(i,j,k)\\|t|=2}} U_c^i U_f^j U_s^k = \sum_{\substack{t=(i,j,k)\\|t|=2\\i=0}} U_c^i U_f^j U_s^k + \sum_{\substack{t=(i,j,k)\\|t|=2\\i>0}} U_c^i U_f^j U_s^k$$

$$= \sum_{\substack{t=(0,j,k)\\|t|=2}} U_c^i U_f^j U_s^k + U_c \sum_{\substack{t=(i,j,k)\\|t|=1}} U_c^i U_f^j U_s^k = g(2,2) + U_c g(1,3)$$

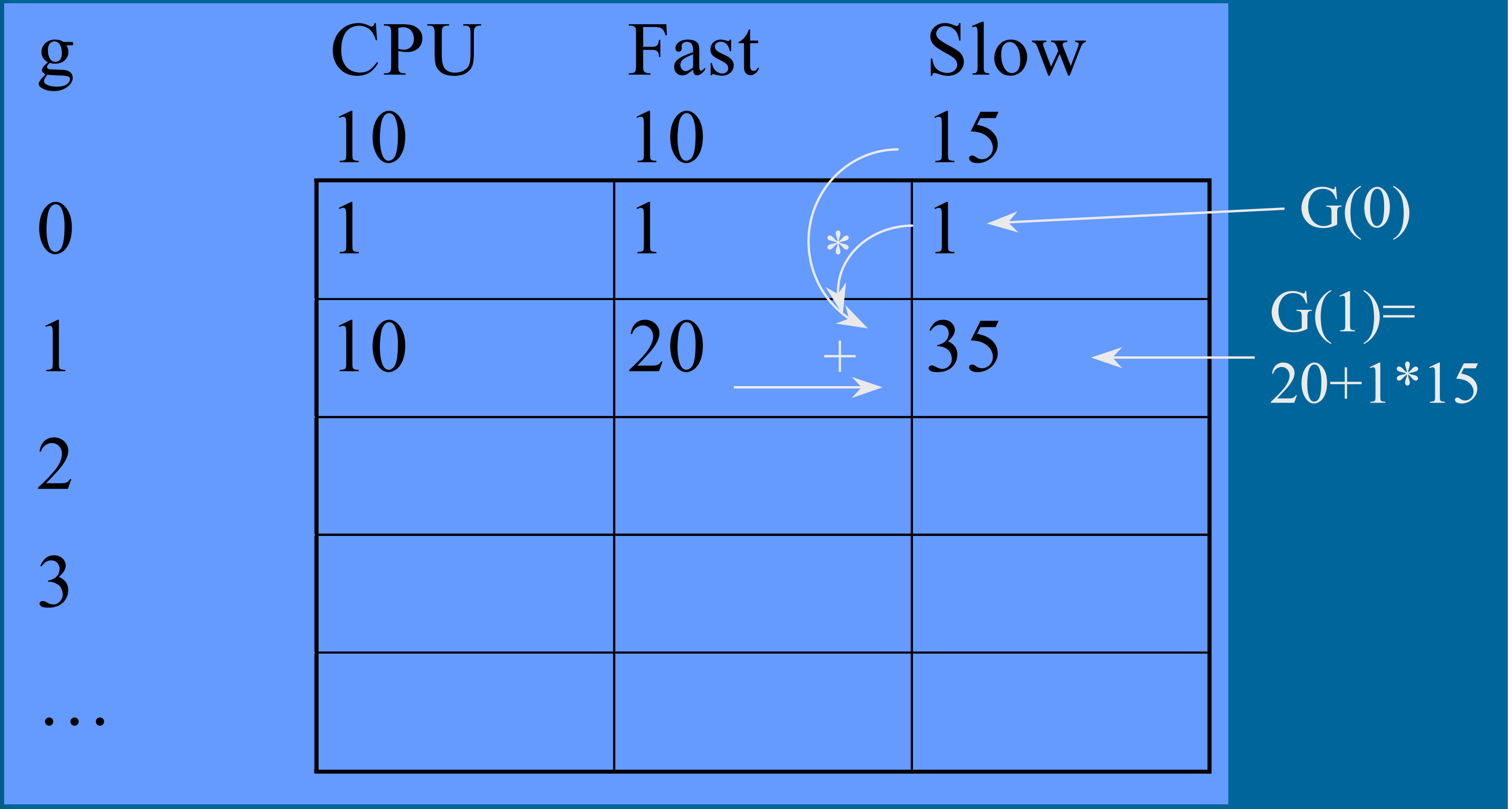
  - Fig 5.4 [Men 94]

Convolution Example

Use D_i's as (relative) U_i's:

	Vi	Si	Ui=ViSi
CPU	1	10	10
Fast	0.5	20	10
Slow	0.5	30	15

Convolution Example (1/3)

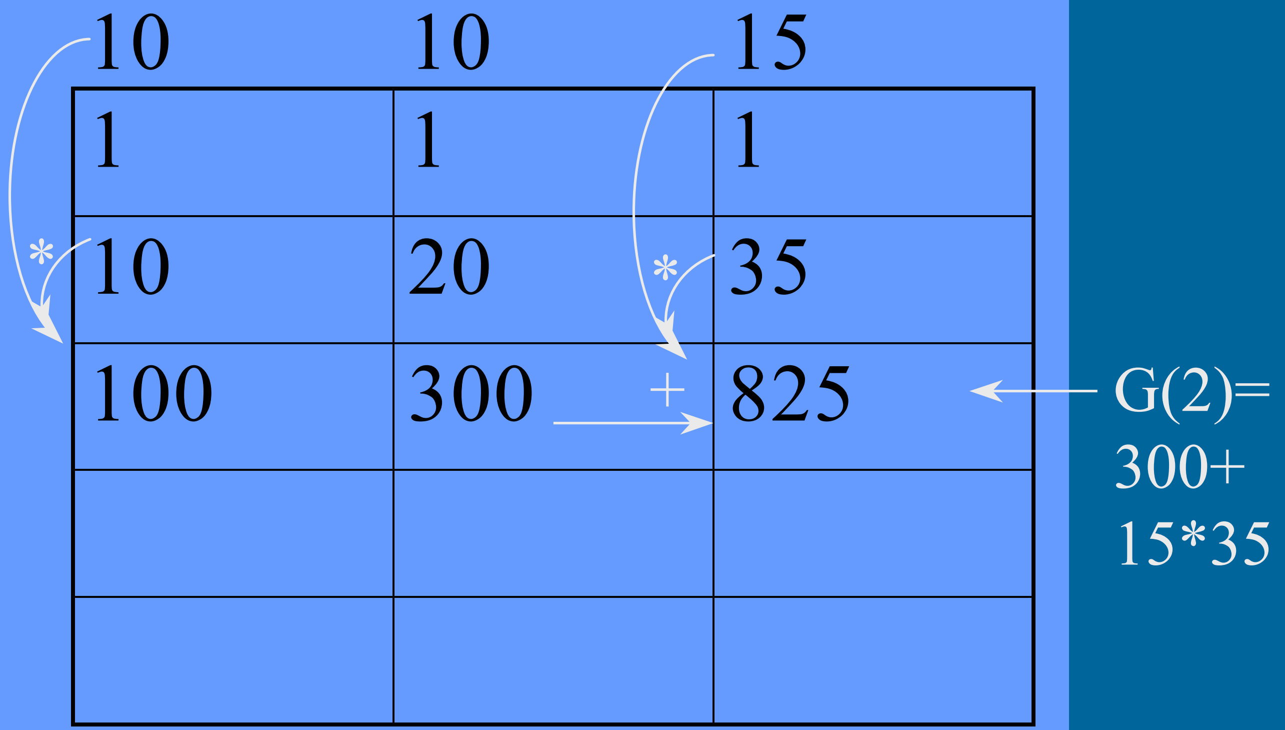


g	CPU	Fast	Slow
	10	10	15
0	1	1	1
1	10	20	35
2			
3			
…			

G(0)

G(1)=
20+1*15

Convolution Example (2/3)



g	CPU	Fast	Slow
	10	10	15
0	1	1	1
1	10	20	35
2	100	300	825
3			
…			

G(2)=
300+
15*35

Convolution Example (3/3)

g	CPU 10	Fast 10	Slow 15
0	1	1	1
1	10	20	35
2	100	300	825
3	1000	4000	16375
…			

← G(3)=
4000+
15*825

Response Time

mean device response time

$$R_c(2) = \frac{U_c^0 G(1) + U_c^1 G(0)}{\mu_c G(1)} = \frac{35+10}{\frac{1}{10}35} = \frac{45*10}{35} = 12.9\,\mathrm{sec}$$

$$R_f(2) = \frac{U_f^0 G(1) + U_f^1 G(0)}{\mu_f G(1)} = \frac{35+10}{\frac{1}{20}35} = \frac{45*20}{35} = 25.7\,\mathrm{sec}$$

$$R_s(2) = \frac{35+15}{\frac{1}{30}35} = \frac{50*30}{35} = 42.9\,\mathrm{sec}$$

$$R(2) = \sum R_i^{'}(2) = \sum V_i R_i\,(2) = 47.2\,\mathrm{sec}$$

mean system response time

Utilization & Throughput (2)

$$U_c(2) = U_c \frac{G(1)}{G(2)} = 10 \frac{35}{825} = 0.424$$

$$U_f(2) = 10 \frac{35}{825} = 0.424$$

$$U_s(2) = 15 \frac{35}{825} = 0.636$$

true utilization (not relative utilization)

$$X_c(2) = \mu_c U_c \frac{G(1)}{G(2)} \overset{U_c = V_c S_c}{=} V_c \frac{G(1)}{G(2)} = 1 * \frac{35}{825} = 0.042$$

Little: $X_0(2)\, R\ (2) = \frac{X_c(2)}{V_c} R\ (2) = \frac{0.042}{1} 47.2 = 1.98 \approx 2$ OK

Average Number of Jobs at Each Server (in Queue and in Service)

Job distribution in network

$$\bar{n}_c(2) = U_c \frac{G(1)}{G(2)} + U_c^2 \frac{G(0)}{G(2)}$$

$$= 10 * \frac{35}{825} + 100 * \frac{1}{825} = \frac{450}{825} \qquad = 0.545$$

$$\bar{n}_f(2) \qquad = 0.545$$

$$\bar{n}_s(2) = 15 * \frac{35}{825} + 225 * \frac{1}{825} = \frac{750}{825} \qquad = 0.91$$

total = 2 OK

Device Queue Lengths

Queue = aver. population – utilization:

$$\bar{n}_c(2) - U_c(2) = 0.545 - 0.424 = 0.125$$

$$\bar{n}_f(2) - U_f(2) = 0.545 - 0.424 = 0.125$$

$$\bar{n}_s(2) - U_s(2) = 0.91 - 0.636 = 0.27$$

Convolution Summary

- Jeff Buzen
  - founded BGS Corporation with two friends
  - BGS merged to BMC Software in 1998
- Novel practical way to compute normalizing constant for closed queueing networks
- Get all standard measures
  - Qi, Ni, Ui, Ri, Xi
  - Qir, Nir, Uir, Rir, Xir
  - Xsystem, Rsystem

(Multiple class version exists!)

- Practical computational problem
  - G can overflow or underflow!