COMP 190-088: Systems Performance Analysis

System Modeling

Multi-server Analysis

Jasleen Kaur

Department of Computer Science
The University of North Carolina at Chapel Hill

Spring 2005

System Modeling Overview

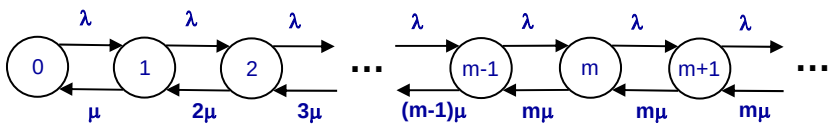
- Queuing Basics
- Single-server Analysis
- **Multiple-server Analysis**
- Operational Laws
- Case Studies
  - Processor Scheduling
  - Disk Scheduling
  - Memory Management
- Network of Queues

Multi-server Systems

- More than one server available for serving requests
  - eg, multiple processors in a multi-processor computing system
- Focus:
  - How to estimate throughput, response time, wait time, …
    - **With infinite buffers (M/M/m)**
    - With finite buffers (M/M/m/B)
  - How to design division of labor among the multiple servers?
    - Common-queue vs. separate queues
- Assume exponential arrival and service times

M/M/m Queue

- m servers, each with service rate μ
  - If any server is idle, an arriving job is serviced immediately
  - If all m servers are busy, arriving job waits in a (common) queue
- System state:
  - Represented by the number of jobs in the system
- State transition diagram:



$$\lambda p_0 = \mu p_1$$

$$\lambda p_{i-1} + (i+1)\mu p_{i+1} = \lambda p_i + (i+1)\mu p_i \text{ ; for } i < m$$

$$\lambda p_{i-1} + m\mu p_{i+1} = \lambda p_i + m\mu p_i \text{ ; for } i \geq m$$

Steady-state Probabilities for M/M/m Queues

- Steady-state probabilities (if $\rho = \lambda/m\mu$):
  - Can be obtained by using the formulation for birth-death processes

$$p_n = \frac{(m\rho)^n}{n!} p_0, \quad n = 1, 2, \ldots, m-1$$

$$p_n = \frac{\rho^n m^m}{m!} p_0, \quad n = m, m+1, \ldots, \infty$$

$$p_0 = \frac{1}{1 + \frac{(m\rho)^m}{m!(1-\rho)} + \sum_{n=1}^{m-1} \frac{(m\rho)^n}{n!}}$$

Erlang's C formula

- Probability that an arriving job has to wait in the queue
  - Known as Erlang's C formula

$$\begin{aligned}
\psi &= P(\geq m.jobs) \\
&= p_m + p_{m+1} + p_{m+2} + \cdots \\
&= p_0 \frac{(m\rho)^m}{m!} \sum_{n=m}^{\infty} \rho^{n-m} \\
&= \frac{(m\rho)^m}{m!(1-\rho)} p_0 = \frac{m p_m}{1-\rho}
\end{aligned}$$

Mean Number of Jobs in Queue and System

$$E[Ns] = \sum_{n=1}^{m-1} np_n + \sum_{n=m}^{\infty} mp_n$$

$$= p_0\left(\frac{(m\rho)}{1!} + 2\frac{(m\rho)^2}{2!} + \cdots + (m-1)\frac{(m\rho)^{m-1}}{(m-1)!}\right) + m\psi$$

$$= m\rho(p_0 + p_1 + p_2 + \cdots + p_{m-2}) + m\psi$$

$$= m\rho(1 - p_{m-1} - \psi) + m\psi$$

$$= m\rho - mp_m - m\psi(1-\rho)$$

$$= m\rho$$

$$\boxed{\psi = \frac{mp_m}{1-\rho}}$$

$$E[Nq] = \sum_{n=m+1}^{\infty} (n-m)p_n$$

$$= p_0 \frac{(m\rho)^m}{m!} \sum_{n=m+1}^{\infty} (n-m)\rho^{n-m}$$

$$= \frac{(m\rho)^m \rho}{m!(1-\rho)^2} p_0$$

$$= \frac{\rho\psi}{1-\rho}$$

$$E[N] = E[Nq] + E[Ns] = m\rho + \frac{\rho\psi}{1-\rho}$$

Other Properties

- System Utilization (U):
  If system is observed for a long time T
  - Total number of jobs served = λT
  - Total busy time of m servers = $\lambda T/\mu$
  - Utilization of each server

    U = (busy time per server)/(total time)
    = $[(\lambda T/\mu)/m]/T$
    = $\lambda/m\mu = \rho$
- Mean response time:
  - Use Little's Law
  - $E[R] = E[N]/\lambda = 1/\mu*(1+\psi/m(1-\rho))$
- Mean waiting time:
  - $E[W] = E[Nq]/\lambda = \psi/m\,\mu(1-\rho)$

Distribution of Response Time

$$F(r) = \begin{cases} 1 - e^{-\mu r} - \dfrac{\psi}{1 - m + m\rho} e^{-m\mu(1-\rho)r} - e^{-\mu r}, & \rho \neq m - 1/m \\ 1 - e^{-\mu r} - \psi \mu r e^{-\mu r}, & \rho = m - 1/m \end{cases}$$

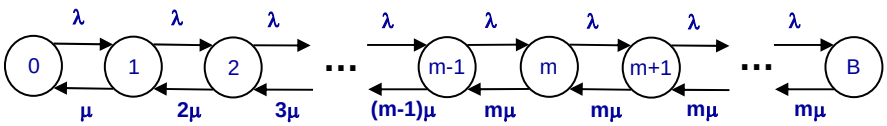
- Response time is not exponentially distributed
  - Unless m = 1
- Coefficient of variation is less than 1
  - Should total computing power be fragmented into multiple smaller processors?

Multi-server Systems

- More than one server available for serving requests
  - eg, multiple processors in a multi-processor computing system
- Focus:
  - How to estimate throughput, response time, wait time, …
    - With infinite buffers (M/M/m)
    - **With finite buffers (M/M/m/B)**
  - How to design division of labor among the multiple servers?
    - Common-queue vs. separate queues
- Assume exponential arrival and service times

M/M/m/B Queue

- B is finite
  - When B buffers are full, arrivals are lost
- Assume that B ≥ m
  - Else, some servers will not be able to operate due to lack of buffers
  - System will effectively operate as an M/M/B/B queue
- State transition diagram:



Steady-state M/M/m/B Probabilities

- Steady-state probabilities (if ρ = λ/mμ):
  - Can be obtained by using the formulation for birth-death processes

$$p_n = \frac{(m\rho)^n}{n!} p_0, \quad n = 1, 2, \ldots, m-1$$

$$p_n = \frac{\rho^n m^m}{m!} p_0, \quad n = m, m+1, \ldots, B$$

$$p_0 = \frac{1}{1 + \frac{(1-\rho^{B-m+1})(m\rho)^m}{m!(1-\rho)} + \sum_{n=1}^{m-1} \frac{(m\rho)^n}{n!}}$$

Properties of M/M/m/B Queues

- Mean number of jobs:

$$E[N] = \sum_{n=1}^{B} np_n = \frac{\rho}{1-\rho} - \frac{(B+1)\rho^{B+1}}{1-\rho^{B+1}}$$

$$E[Nq] = \sum_{n=m+1}^{B} (n-m)p_n = \frac{\rho}{1-\rho} - \rho\frac{1+B\rho^{B}}{1-\rho^{B+1}}$$

- Effective arrival rate (λ')
  - Arrivals in state n=B are lost
  - Rate at which jobs enter the system: $\lambda' = \sum_{n=0}^{B-1} \lambda p_n = \lambda(1 - p_B)$
- Mean response and waiting times:
  - E[R] = E[N] / λ'
  - E[W] = E[Nq] / λ'
- Utilization:
  - If system is observed for a long time T
    - Total number of jobs receiving service = $\lambda' T$
    - Total busy time of m servers = $\lambda' T/\mu$
    - $U = ((\lambda' T/\mu)/m)/T = \rho(1-p_B)$

Erlang's Loss Formula for M/M/m/m Queues

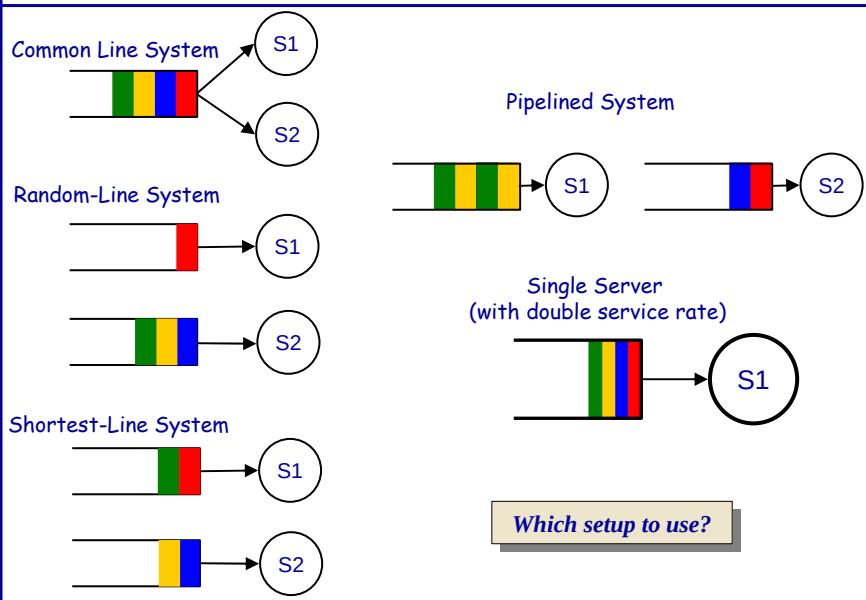
- Number of buffers is equal to the number of servers (B = m)
- Loss probability:
  - Known as Erlang's Loss Formula
  - Also works for M/G/m/m queues

$$\begin{aligned} p_B &= p_m \\ &= \frac{(m\rho)^m}{m!} p_0 \\ &= \frac{(m\rho)^m / m!}{\sum_{j=0}^{m} \left[(m\rho)^j / j!\right]} \end{aligned}$$

Multi-server Systems

- More than one server available for serving requests
  - eg, multiple processors in a multi-processor computing system
- Focus:
  - How to estimate throughput, response time, wait time, …
    - With infinite buffers (M/M/m)
    - With finite buffers (M/M/m/B)
  - **How to design division of labor among the multiple servers?**
    - **Common-queue vs. separate queues**
- Assume exponential arrival and service times

Alternate Server and Queue Configurations



Which setup to use?

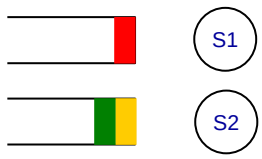
Considerations in Selecting a Setup

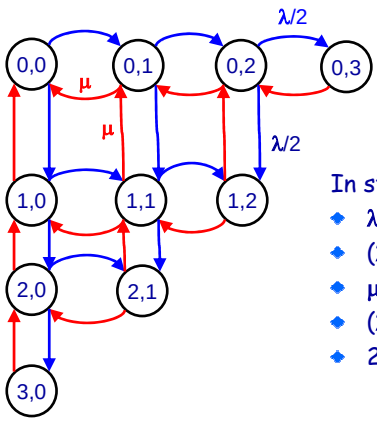
- Objectives:
  - Maximizing system throughput
  - Minimizing average response time
  - Minimizing average waiting time
- Resources and Load:
  - Server capacity (service times) and arrival rate
    - Assume: λ=1/2 and μ=1/2 (request per second)
  - Buffer capacity
    - Assume 3 buffers
  - Distribution of arrival and service times
    - Assume IID exponential distributions
- Configuration Models:
  - Single-server system: M/M/1/3 (with double service rate)
  - Common-line system: M/M/2/3

How to model random-line and shortest-line systems?

Random Line System

- Arrival rate into each queue: $\lambda/2$
- Combined capacity of the two queues: 3
- State represented by: (n1, n2)
- State transition diagram:





In steady-state:

- $\lambda P_{0,0} = \mu P_{0,1} + \mu P_{1,0}$
- $(\lambda + \mu)P_{0,1} = \lambda/2 P_{0,0} + \mu P_{0,2} + \mu P_{1,1}$
- $\mu P_{0,3} = \lambda/2 P_{0,2}$
- $(\lambda + 2\mu)P_{1,1} = \lambda/2\, P_{0,1} + \lambda/2\, P_{1,0} + \mu P_{1,2} + \mu P_{2,1}$
- $2\,\mu P_{1,2} = \lambda/2\, P_{0,2} + \lambda/2\, P_{1,1}$

Random Line: Steady-state Probabilities

- Steady-state probabilities :

$$p_{0,0} = \frac{4\mu^3}{2\lambda^3 + 3\lambda^2\mu + 4\lambda\mu^2 + 4\mu^3}$$

$$p_{0,1} = p_{1,0} = \frac{2\lambda\mu^2}{2\lambda^3 + 3\lambda^2\mu + 4\lambda\mu^2 + 4\mu^3}$$

$$p_{0,2} = p_{1,1} = p_{2,0} = \frac{\lambda^2\mu}{2\lambda^3 + 3\lambda^2\mu + 4\lambda\mu^2 + 4\mu^3}$$

$$p_{0,3} = p_{1,2} = p_{2,1} = p_{3,0} = \frac{\lambda^3/2}{2\lambda^3 + 3\lambda^2\mu + 4\lambda\mu^2 + 4\mu^3}$$

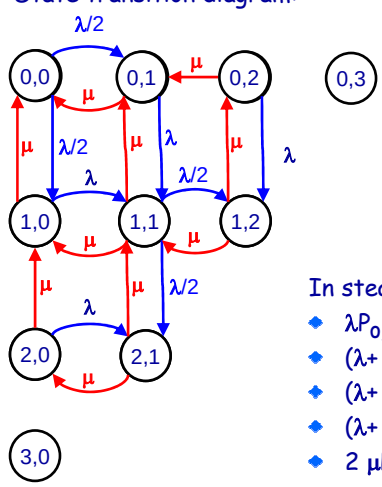
Random Line: Properties

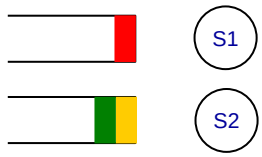
$\lambda = 1/2$
$\mu = 1/2$

- Utilization (U):
  - First server:
    - $1 - (P_{0,0} + P_{0,1} + P_{0,2} + P_{0,3}) = 11/26$
  - Second server:
    - $1 - (P_{0,0} + P_{1,0} + P_{2,0} + P_{3,0}) = 11/26$
- Throughput (X):
  - Utilization/mean service time = $U\mu$
  - $(11/26)/2 + (11/26)/2 = 11/26$
- Mean response time (E[R]):
  - $E[R] = E[N] / X = 2.909$
- Mean waiting time (E[W]):
  - $E[W] = E[R] - 1/\mu = 0.909$

Shortest Line System

- State transition diagram:





In steady-state:

- $\lambda P_{0,0} = \mu P_{0,1} + \mu P_{1,0}$
- $(\lambda + \mu) P_{0,1} = \lambda/2 P_{0,0} + \mu P_{0,2} + \mu P_{1,1}$
- $(\lambda + \mu) P_{0,2} = \mu P_{1,2}$
- $(\lambda + 2\mu) P_{1,1} = \lambda P_{0,1} + \lambda P_{1,0} + \mu P_{1,2} + \mu P_{2,1}$
- $2 \mu P_{1,2} = \lambda P_{0,2} + \lambda/2 P_{1,1}$

Shortest Line: Steady-state Probabilities

- Steady-state probabilities :

$$p_{0,0} = \frac{\mu^3(3\lambda + 4\mu)}{\lambda^4 + 3\lambda^3\mu + 5\lambda^2\mu^2 + 7\lambda\mu^3 + 4\mu^4}$$

$$p_{0,1} = p_{1,0} = \frac{\lambda\mu^2(3\lambda + 4\mu)/2}{\lambda^4 + 3\lambda^3\mu + 5\lambda^2\mu^2 + 7\lambda\mu^3 + 4\mu^4}$$

$$p_{0,2} = p_{2,0} = \frac{\lambda^3\mu/2}{\lambda^4 + 3\lambda^3\mu + 5\lambda^2\mu^2 + 7\lambda\mu^3 + 4\mu^4}$$

$$p_{1,1} = \frac{\lambda^2\mu(\lambda + 2\mu)}{\lambda^4 + 3\lambda^3\mu + 5\lambda^2\mu^2 + 7\lambda\mu^3 + 4\mu^4}$$

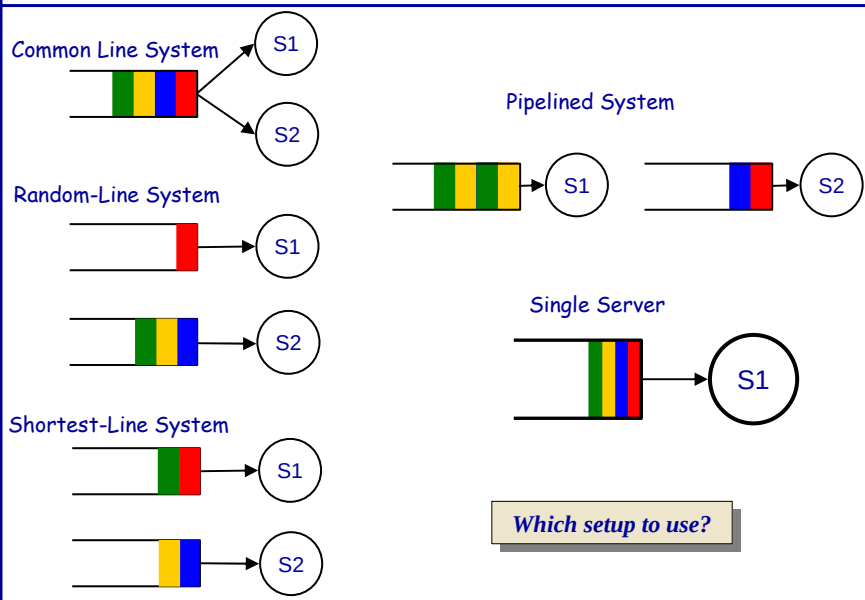
$$p_{1,2} = p_{2,1} = \frac{\lambda^3(\lambda + \mu)/2}{\lambda^4 + 3\lambda^3\mu + 5\lambda^2\mu^2 + 7\lambda\mu^3 + 4\mu^4}$$

Shortest Line: Properties

$\lambda = 1/2$
$\mu = 1/2$

- Utilization (U):
  - First server:
    - $1 - (P_{0,0} + P_{0,1} + P_{0,2}) = 9/20$
  - Second server:
    - $1 - (P_{0,0} + P_{1,0} + P_{2,0}) = 9/20$
- Throughput (X):
  - Utilization/mean service time
  - $(9/20)/2 + (9/20)/2 = 9/20$
- Mean response time (E[R]):
  - $E[R] = E[N] / X = 2.333$
- Mean waiting time (E[W]):
  - $E[W] = E[R] - 1/\mu = 0.333$

Alternate Server and Queue Configurations



Which setup to use?

Comparison of Alternate Configurations

Metric	Configuration			
	Random Line	Shortest Line	Common Line	Single Server
Throughput	0.423	0.450	0.454	0.466
Response Time	2.909	2.333	2.202	1.571
Waiting Time	0.909	0.333	0.202	0.571

Shortest-line outperforms random-line in timeliness

It is better to have a common wait queue in a multi-server system

Single servers with same raw capacity yield better throughput and response times