Large-scale Distributed Systems and Networks (TDDE35)

Slides by Niklas Carlsson (including slides based on slides by Carey Williamson)

PERFORMANCE EVALUATION

- Often in Computer Science you need to:
  - demonstrate that a new concept, technique, or algorithm is feasible
  - demonstrate that a new method is better than an existing method
  - understand the impact of various factors and parameters on the performance, scalability, or robustness of a system

PERFORMANCE EVALUATION

- There is a whole field of computer science called computer systems performance evaluation that is devoted to exactly this
- One classic book is Raj Jain's "The Art of Computer Systems Performance Analysis", Wiley & Sons, 1991
- Much of what is outlined in this presentation is described in more detail in [Jain 1991]
- The ACM SIG for Performance is ACM SIGMETRICS (who also have a yearly flag-ship conference)

PERF EVAL: THE BASICS

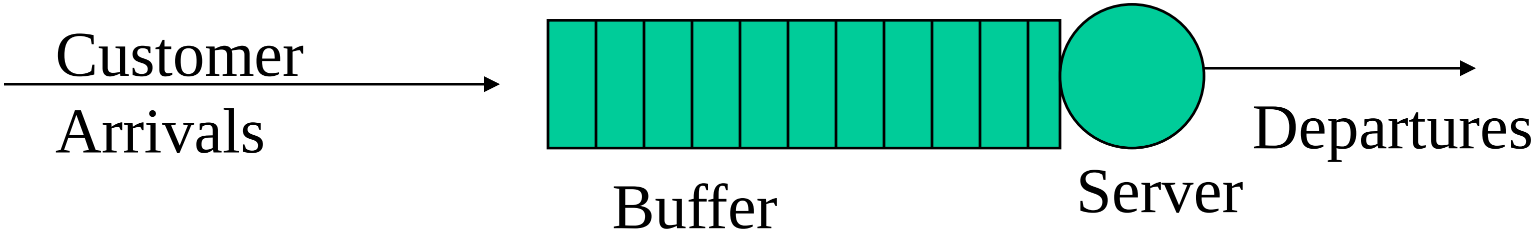
- There are three main methods used in the design of performance evaluation studies:
- Analytic approaches
    - the use of mathematics, Markov chains, queueing theory, Petri Nets, abstract models…
- Simulation approaches
    - design and use of computer simulations and simplified models to assess performance
- Experimental approaches
    - measurement and use of a real system

Analytical Example: Queueing Theory

- Queueing theory is a mathematical technique that specializes in the analysis of queues; e.g.,
  - customer arrivals at a bank,
  - jobs arriving at CPU,
  - I/O requests arriving at a disk subsystem,
  - lineup at the cafeteria
  - etc. …

Queue-based Models

- Queueing model represents:
  - Arrival of jobs (customers) into system
  - Service time requirements of jobs
  - Waiting of jobs for service
  - Departures of jobs from the system
- Typical diagram:



Why Queue-based Models?

- In many cases, the use of a queuing model provides a quantitative way to assess system performance
  - Throughput (e.g., job completions per second)
  - Response time (e.g., Web page download time)
  - Expected waiting time for service
  - Number of buffers required to control loss
- Reveals key system insights (properties)
- Often with efficient, closed-form calculation

Caveats and Assumptions

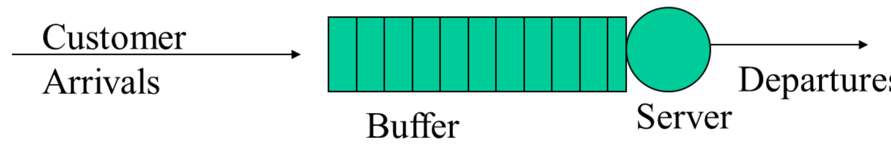
- In many cases, using a queuing model has the following implicit underlying assumptions:
  - Poisson arrival process
    1. Exponential interarrival times
    2. Independent interarrival times
  - Exponential service time distribution
  - Single server
  - Infinite capacity queue
  - First-Come-First-Serve (FCFS) discipline (also known as FIFO: First-In-First-Out)
- Note: important role of memoryless property!

Advanced Queueing Models

- There is TONS of published work on variations of the basic model:
  - Correlated arrival processes
  - General (G) service time distributions
  - Multiple servers
  - Finite capacity systems
  - Other scheduling disciplines (non-FIFO)
- We will start with the basics!

Queue Notation

- Queues are concisely described using the Kendall notation, which specifies:
  - Arrival process for jobs {M, D, G, …}
  - Service time distribution {M, D, G, …}
  - Number of servers {1, n}
  - Storage capacity (buffers) {B, infinite}
  - Service discipline {FIFO, PS, SRPT, …}
- Examples: M/M/1, M/G/1, M/M/c/c

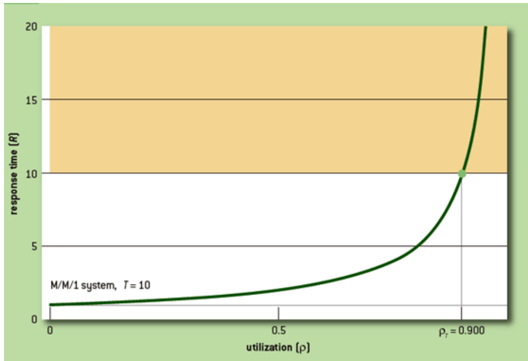


The M/M/1 Queue

- Assumes:
  - Poisson arrival process, exponential service times, single server, FCFS service discipline, infinite capacity for storage, with no loss
- Notation: M/M/1
  - Markovian arrival process (Poisson)
  - Markovian service times (exponential)
  - Single server (FCFS, infinite capacity)

The M/M/1 Queue (cont'd)

- Arrival rate: λ (e.g., customers/sec)
  - Inter-arrival times are exponentially distributed (and independent) with mean $1 / \lambda$
- Service rate: μ (e.g., customers/sec)
  - Service times are exponentially distributed (and independent) with mean $1 / \mu$

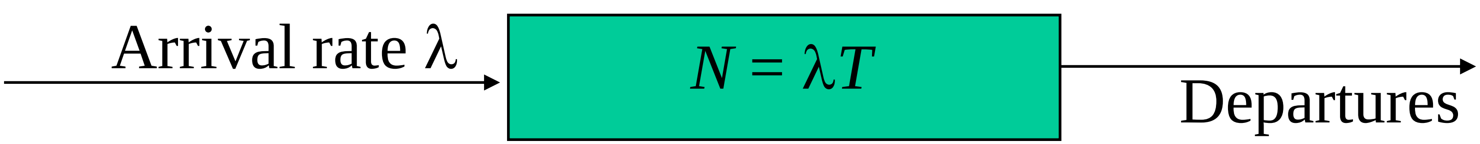


- System load: $\rho = \lambda / \mu$

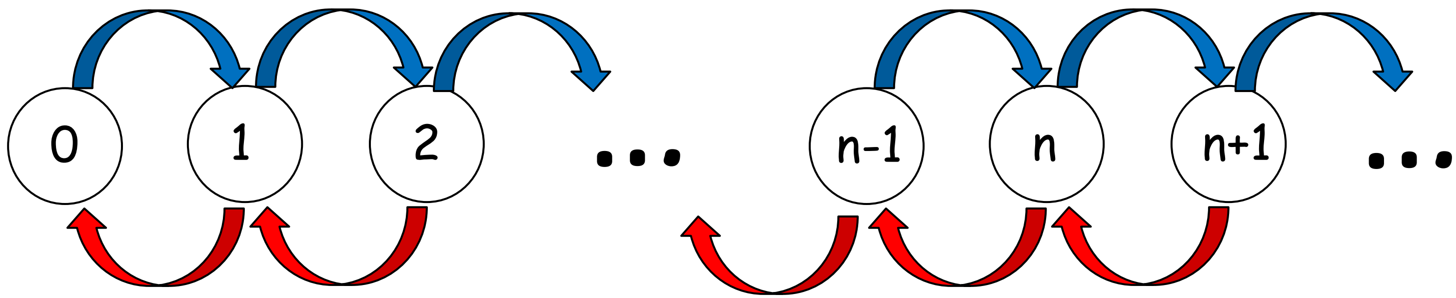
  $0 \leq \rho \leq 1$ (also known as utilization factor)
- Stability criterion: $\rho < 1$ (single server systems)

Queue Performance Metrics

- N: Avg number of customers in system as a whole, including any in service
- Q: Avg number of customers in the queue (only), excluding any in service
- W: Avg waiting time in queue (only)
- T: Avg time spent in system as a whole, including wait time plus service time
- Note: Little's Law: N = λ T (on average)



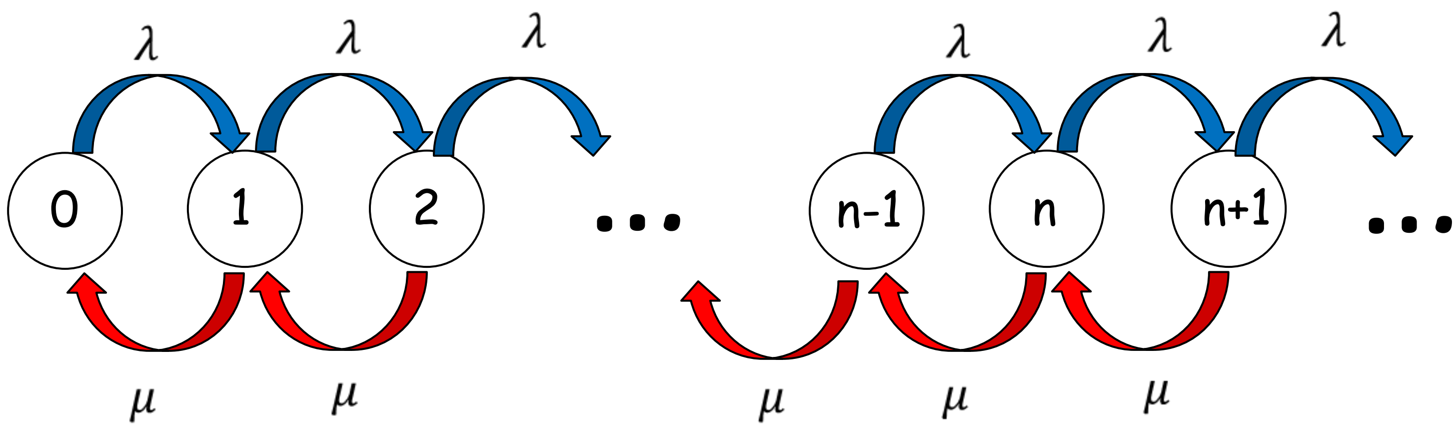
M/M/1 Queue Example



Consider system state (# of customers in system)

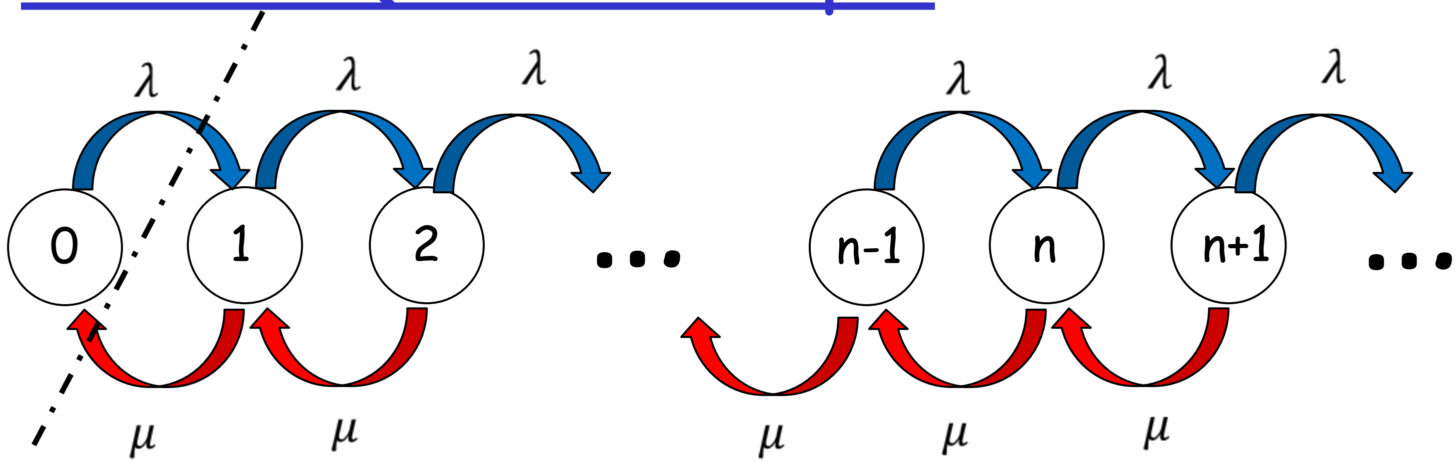
- If arrival, then move up one state …
- If departure, then move down one state …

M/M/1 Queue Example



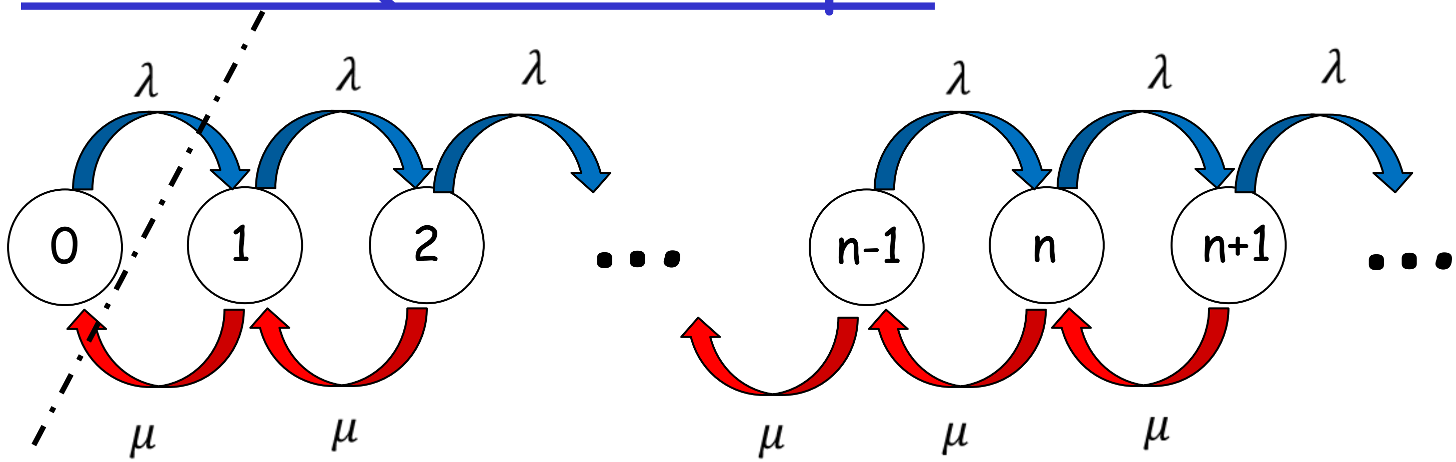
$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

M/M/1 Queue Example



$$N = E[n] = \sum_{n=0}^{\infty} n \, p_n$$

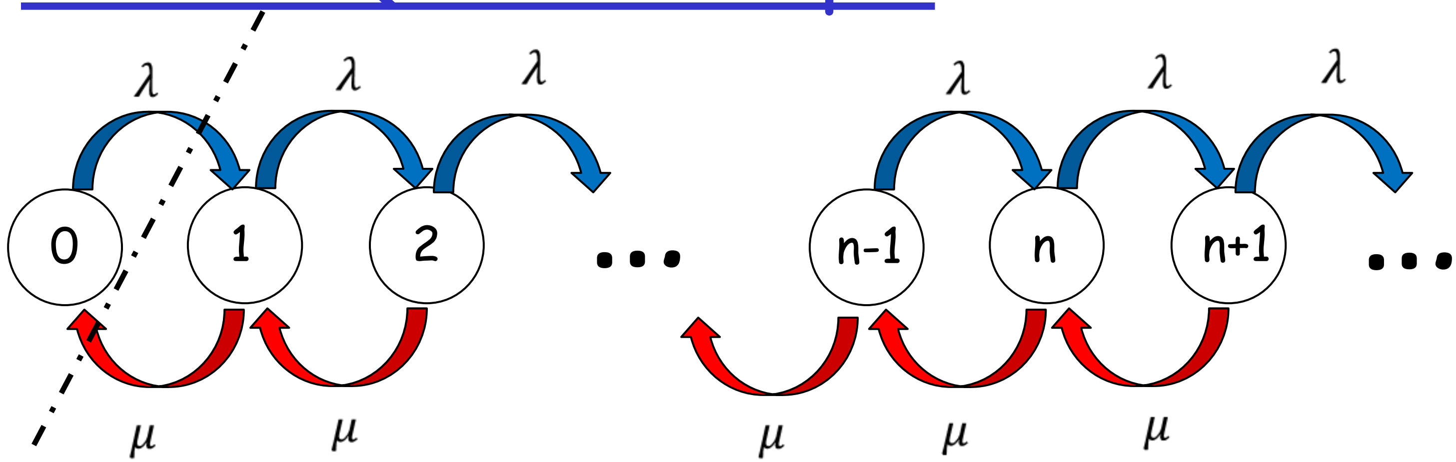
M/M/1 Queue Example



$$p_0\lambda = p_1\mu$$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

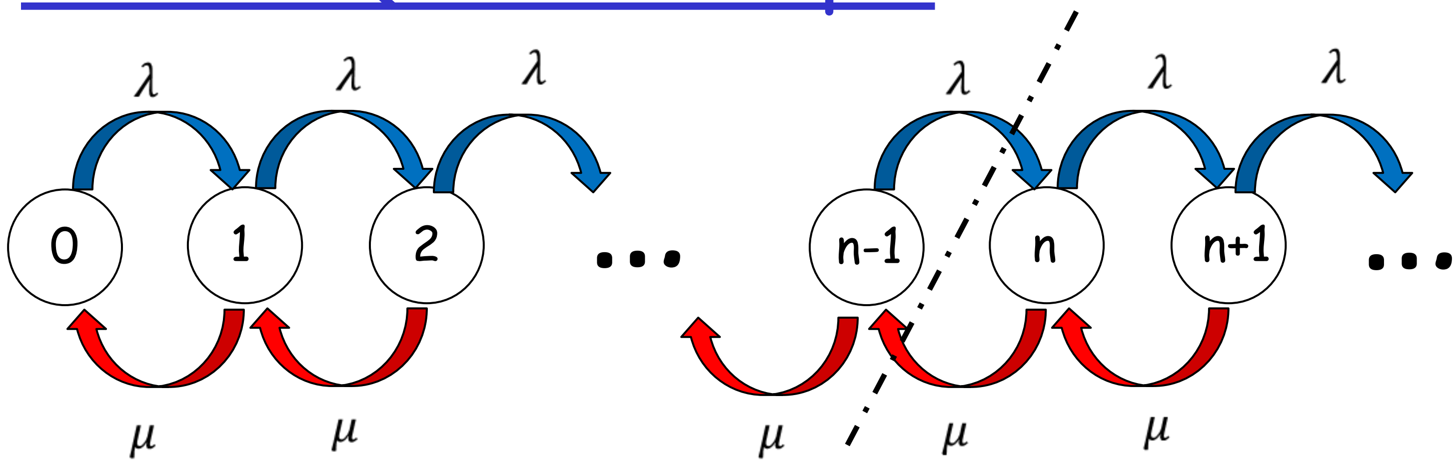
M/M/1 Queue Example



$$p_0\lambda = p_1\mu$$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

M/M/1 Queue Example

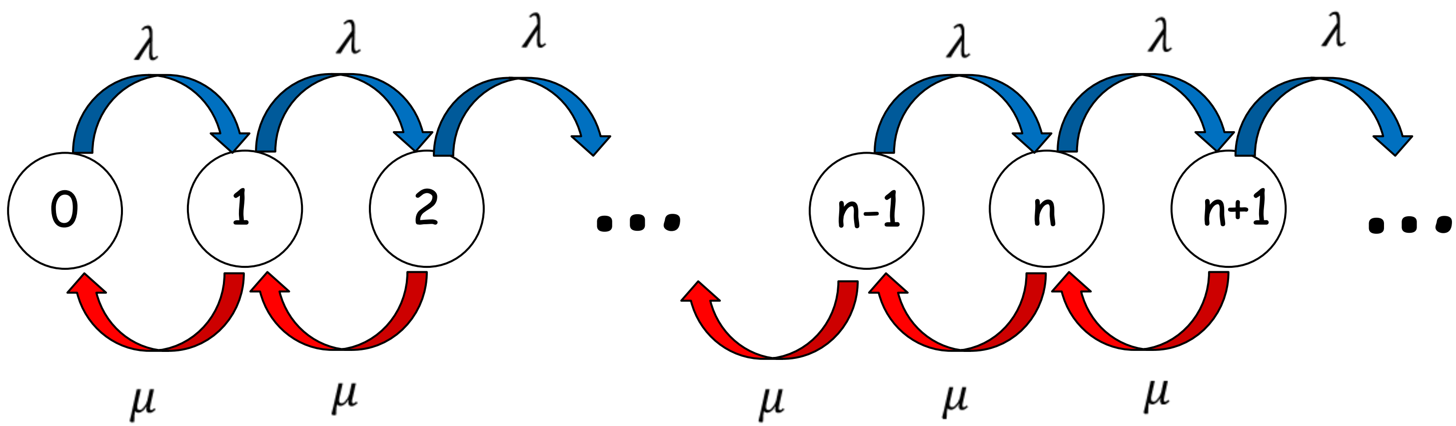


$p_0\lambda = p_1\mu$

$$\boxed{p_{n-1}\lambda = p_n\mu}$$

$$\boxed{N = E[n] = \sum_{n=0}^{\infty} n\, p_n}$$

M/M/1 Queue Example

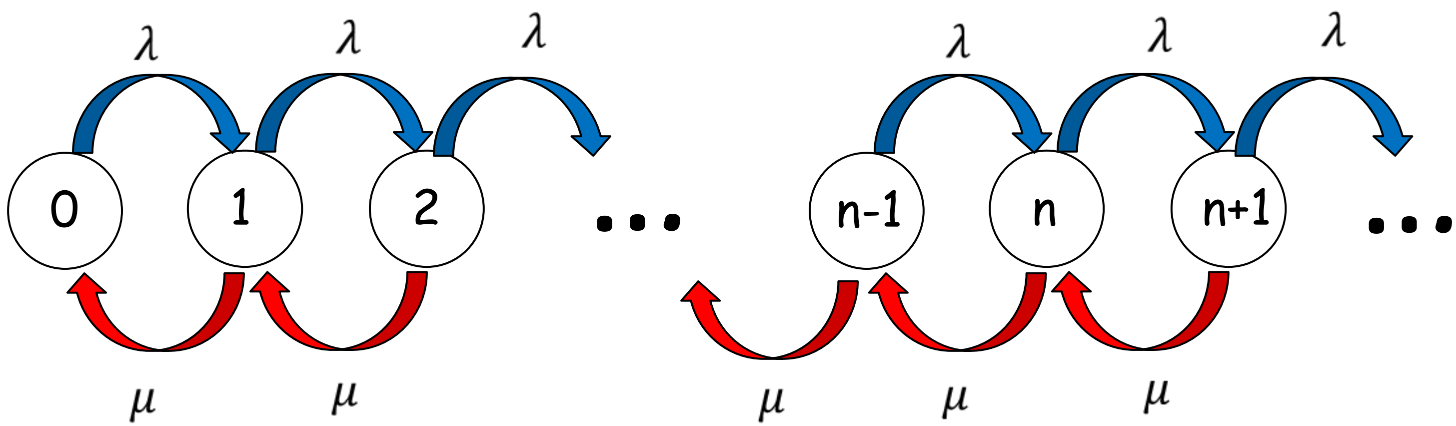


$$p_0\lambda = p_1\mu$$

$$\boxed{p_{n-1}\lambda = p_n\mu}$$

$$\boxed{N = E[n] = \textstyle\sum_{n=0}^{\infty} n\, p_n}$$

M/M/1 Queue Example



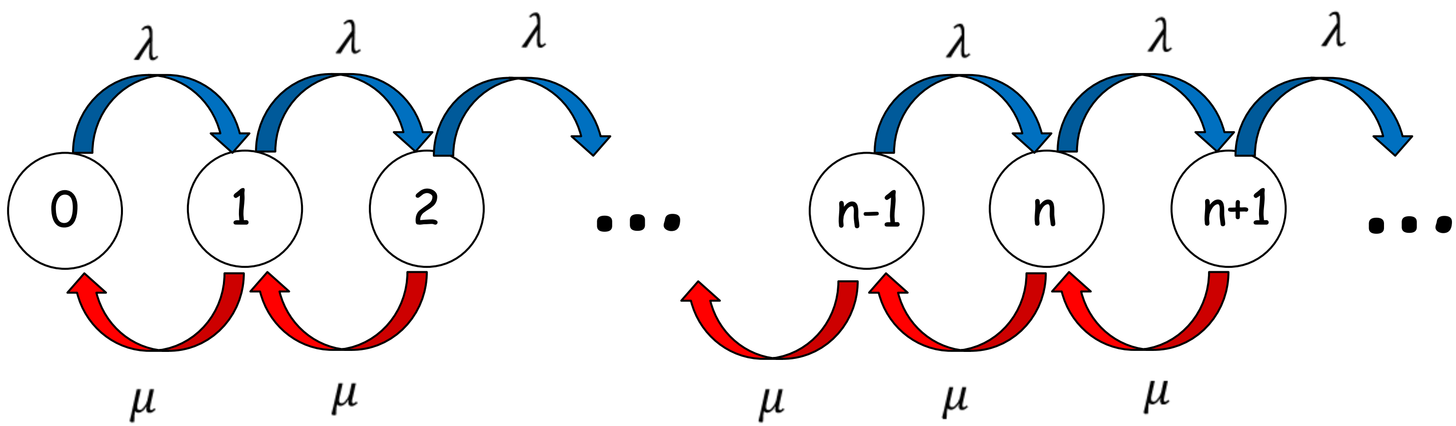
$$p_0\lambda = p_1\mu$$

$$p_{n-1}\lambda = p_n\mu$$

$$p_n = \frac{\lambda}{\mu}p_{n-1}$$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

M/M/1 Queue Example

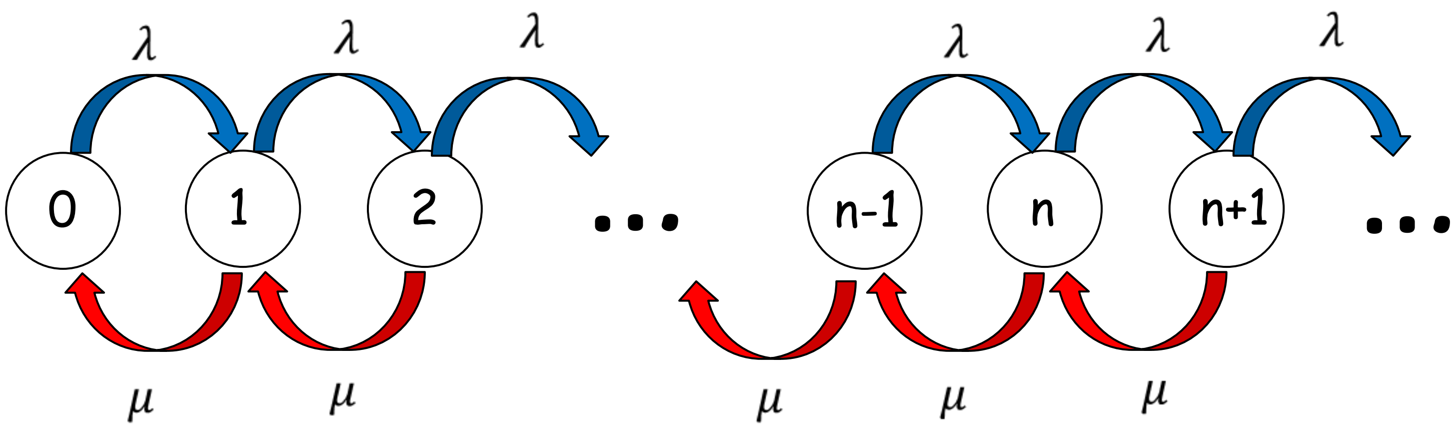


$$p_0\lambda = p_1\mu \qquad\qquad p_{n-1}\lambda = p_n\mu$$

$$p_n = \frac{\lambda}{\mu}p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2}$$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

M/M/1 Queue Example

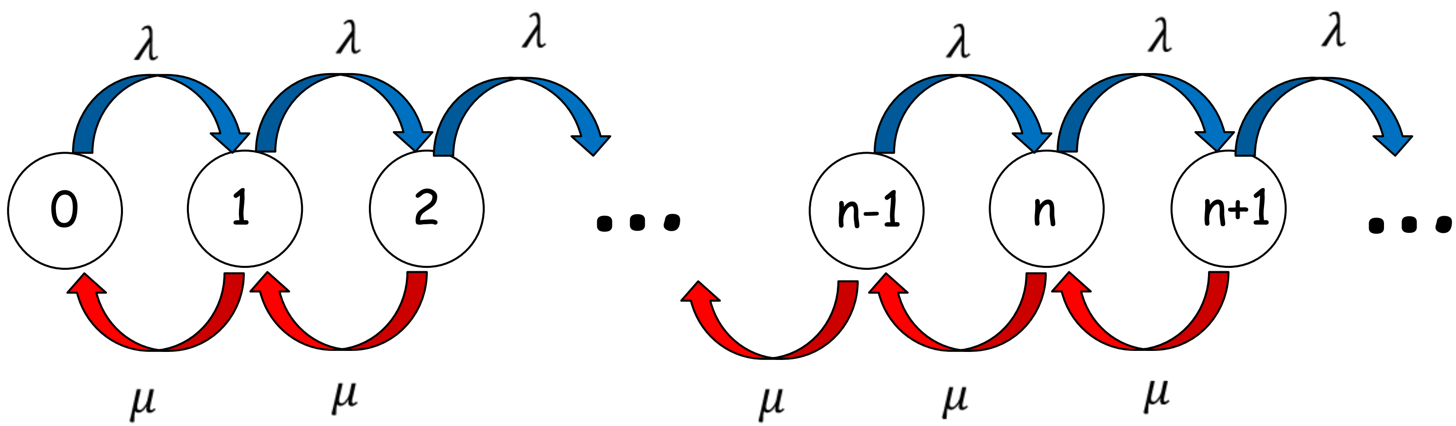


$p_0\lambda = p_1\mu$ $\qquad$ $p_{n-1}\lambda = p_n\mu$

$$p_n = \frac{\lambda}{\mu} p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0$$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

M/M/1 Queue Example



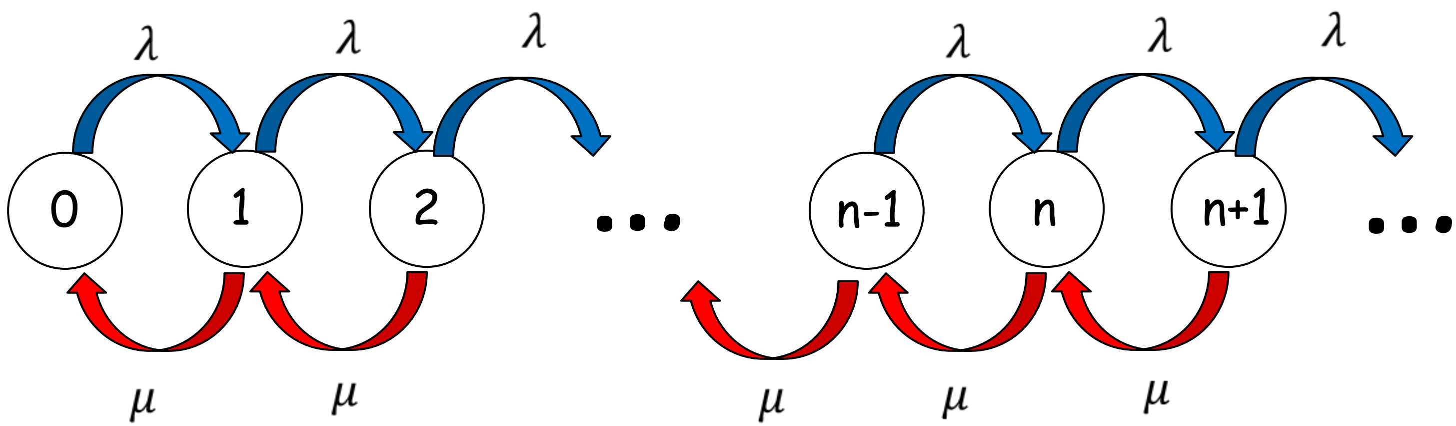
$$p_0\lambda = p_1\mu$$

$$p_{n-1}\lambda = p_n\mu$$

$$p_n = \frac{\lambda}{\mu} p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0$$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

M/M/1 Queue Example



$p_0\lambda = p_1\mu$

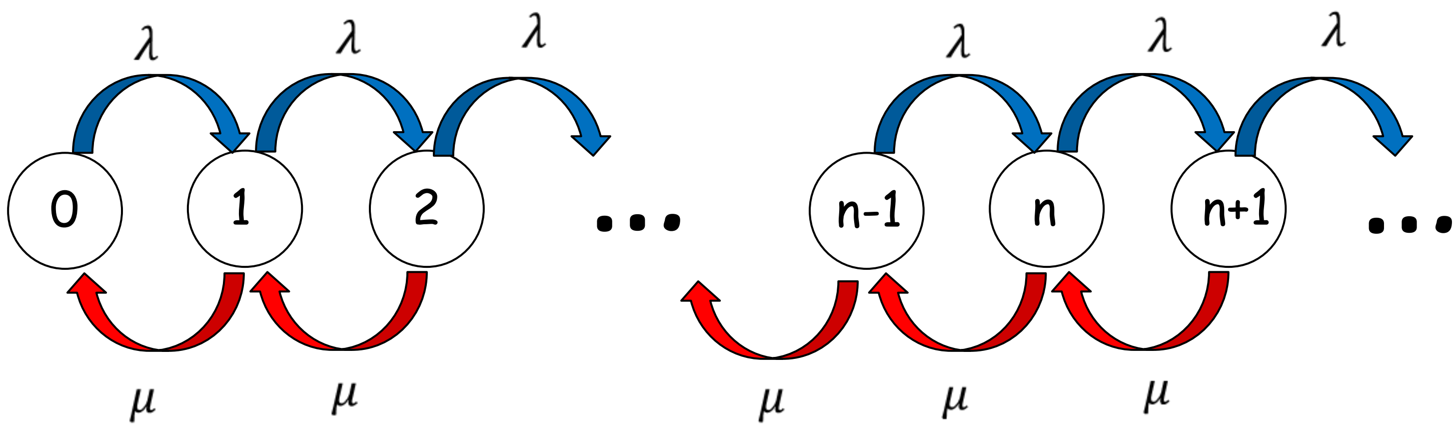
$p_{n-1}\lambda = p_n\mu$

$$p_n = \frac{\lambda}{\mu} p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0$$

$1 = \sum_{n=0}^{\infty} p_n$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

M/M/1 Queue Example



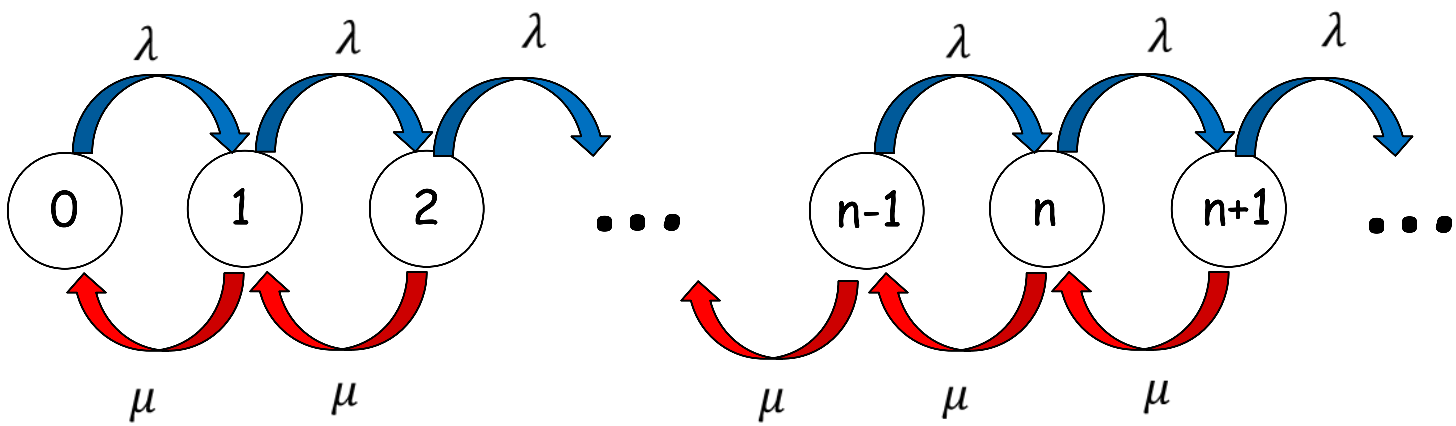
$$p_0\lambda = p_1\mu \qquad\qquad p_{n-1}\lambda = p_n\mu$$

$$\boxed{p_n = \frac{\lambda}{\mu}p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0}$$

$$1 = \sum_{n=0}^{\infty} p_n$$

$$\boxed{N = E[n] = \sum_{n=0}^{\infty} n\, p_n}$$

M/M/1 Queue Example



$p_0\lambda = p_1\mu$

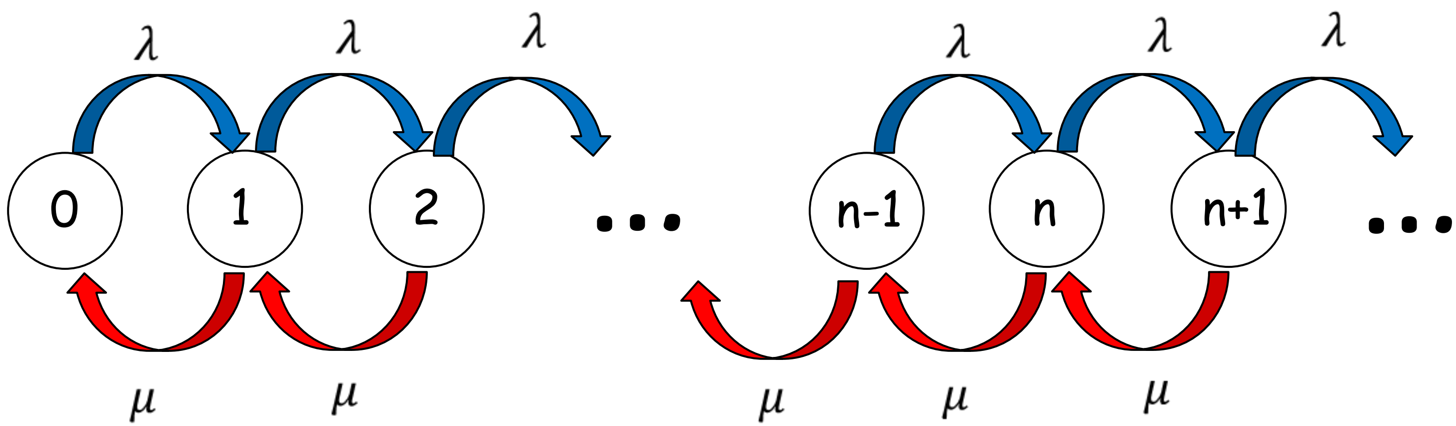
$p_{n-1}\lambda = p_n\mu$

$$p_n = \frac{\lambda}{\mu} p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0 \sum_{n=0}^{\infty} \rho^n$$

$$N = E[n] = \sum_{n=0}^{\infty} n \, p_n$$

M/M/1 Queue Example



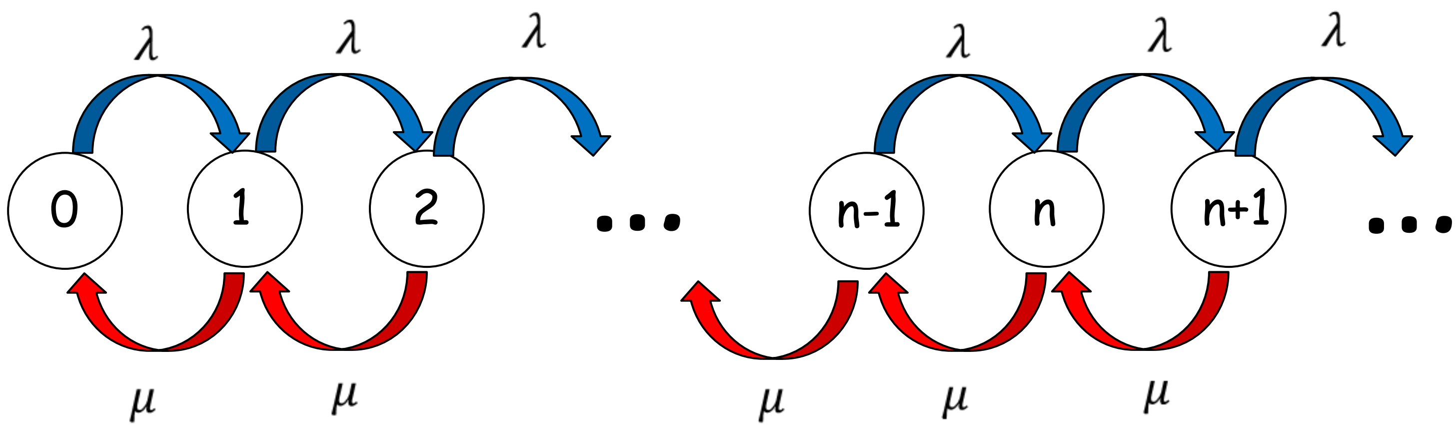
$$p_0\lambda = p_1\mu \qquad\qquad p_{n-1}\lambda = p_n\mu$$

$$\boxed{p_n = \frac{\lambda}{\mu}p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0}$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0 \sum_{n=0}^{\infty} \rho^n = p_0 \frac{1}{1-\rho}$$

$$\boxed{N = E[n] = \sum_{n=0}^{\infty} n\, p_n}$$

M/M/1 Queue Example



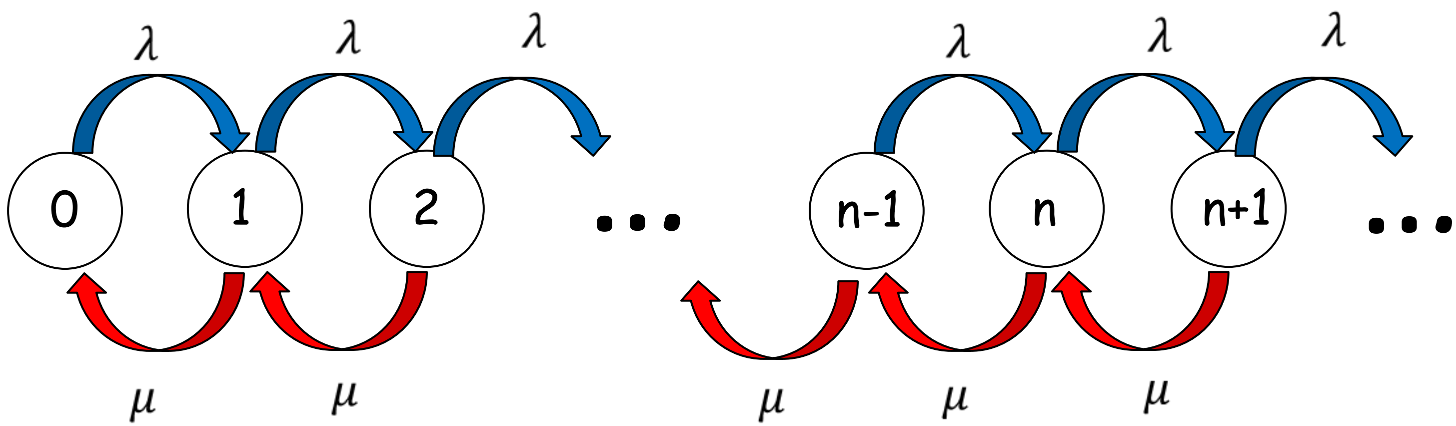
$$p_0\lambda = p_1\mu \qquad\qquad p_{n-1}\lambda = p_n\mu$$

$$p_n = \frac{\lambda}{\mu} p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0 \sum_{n=0}^{\infty} \rho^n = p_0 \frac{1}{1-\rho} \Longrightarrow p_0 = 1 - \rho$$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

M/M/1 Queue Example



$$p_0\lambda = p_1\mu$$

$$p_{n-1}\lambda = p_n\mu$$

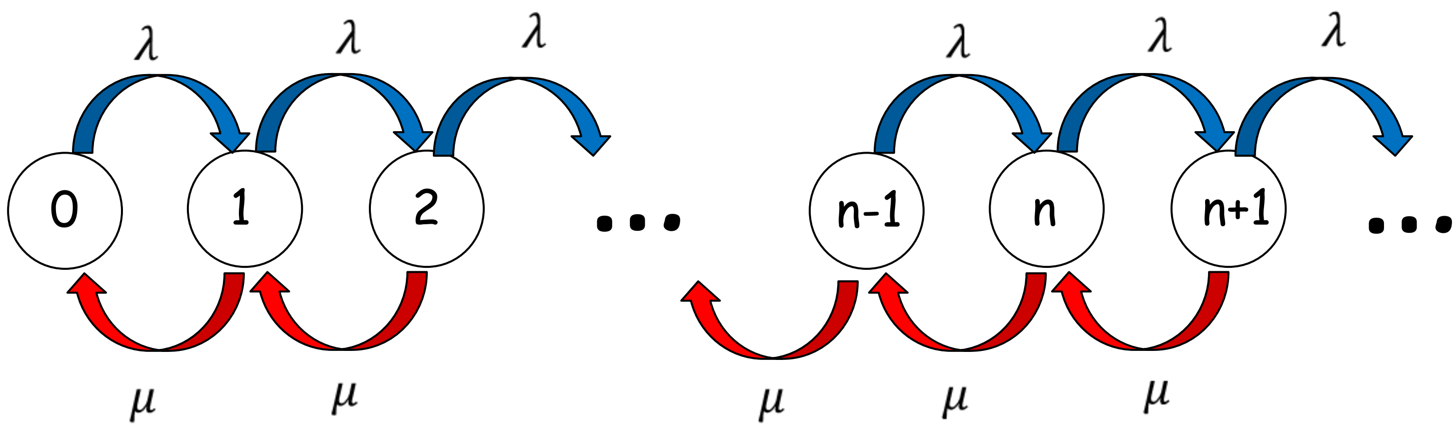
$$p_n = \frac{\lambda}{\mu} p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0 \sum_{n=0}^{\infty} \rho^n = p_0 \frac{1}{1-\rho}$$

$$p_0 = 1 - \rho$$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

M/M/1 Queue Example



$$p_0\lambda = p_1\mu \qquad\qquad p_{n-1}\lambda = p_n\mu$$

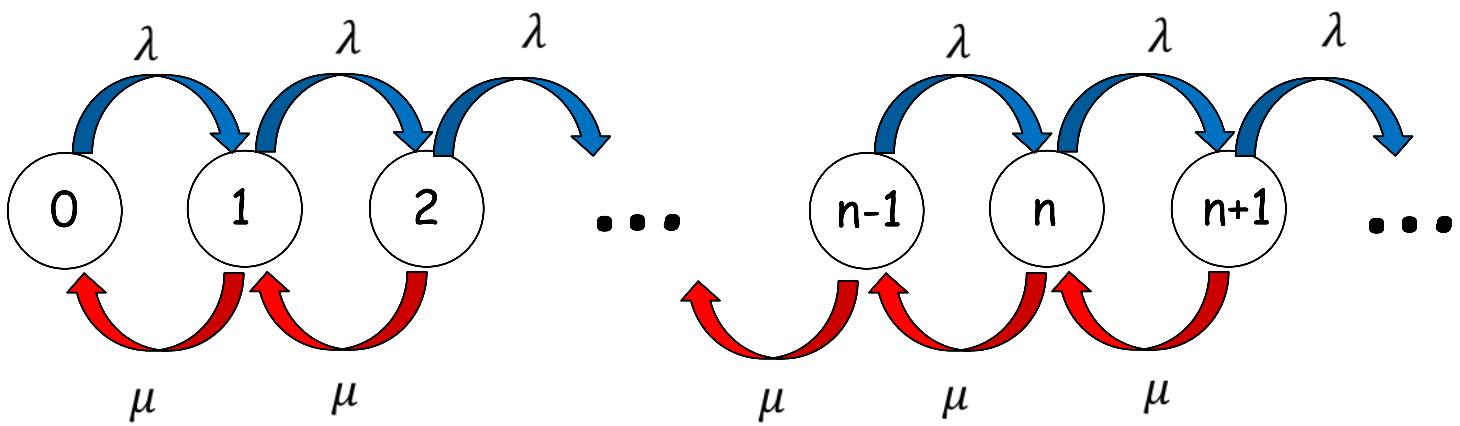
$$p_n = \frac{\lambda}{\mu}p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0 \sum_{n=0}^{\infty} \rho^n = p_0 \frac{1}{1-\rho}$$

$$p_0 = 1 - \rho$$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

M/M/1 Queue Example



$$p_0\lambda = p_1\mu \qquad\qquad p_{n-1}\lambda = p_n\mu$$

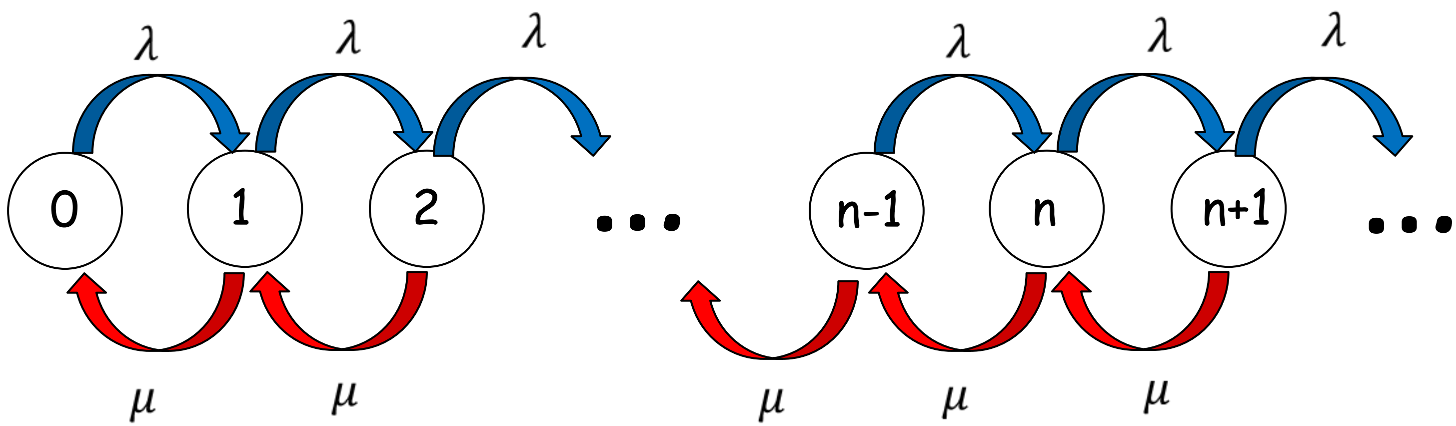
$$\boxed{p_n = \frac{\lambda}{\mu}p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0 = (1-\rho)\rho^n}$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0\sum_{n=0}^{\infty}\rho^n = p_0\frac{1}{1-\rho}$$

$$\boxed{p_0 = 1-\rho}$$

$$\boxed{N = E[n] = \sum_{n=0}^{\infty} n\, p_n}$$

M/M/1 Queue Example



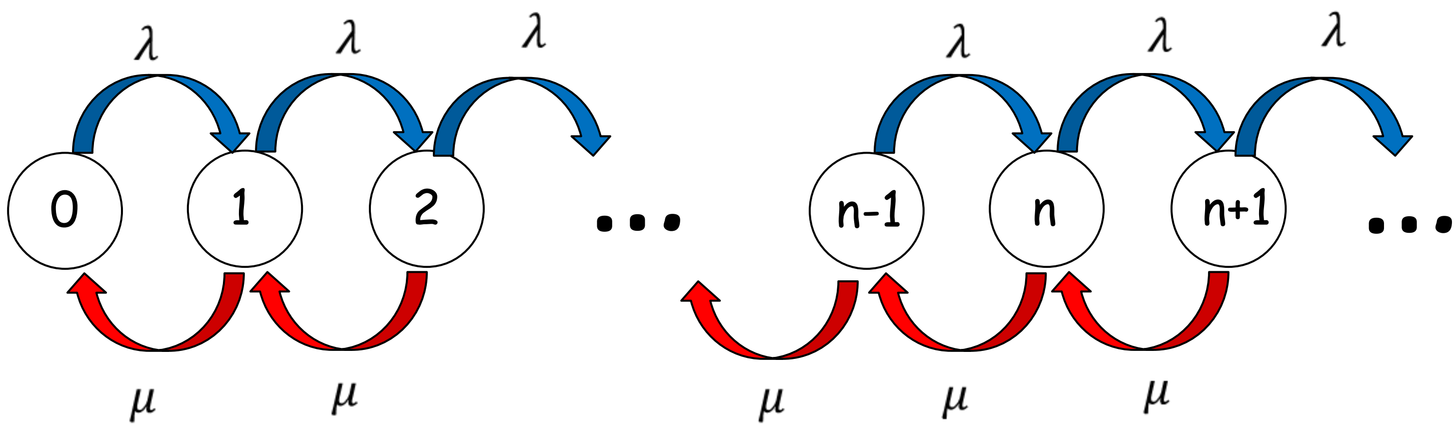
$$p_0\lambda = p_1\mu \qquad\qquad p_{n-1}\lambda = p_n\mu$$

$$p_n = \frac{\lambda}{\mu}p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0 = (1-\rho)\rho^n$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0 \sum_{n=0}^{\infty} \rho^n = p_0 \frac{1}{1-\rho} \Rightarrow p_0 = 1-\rho$$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n$$

M/M/1 Queue Example



$p_0 \lambda = p_1 \mu$

$p_{n-1} \lambda = p_n \mu$

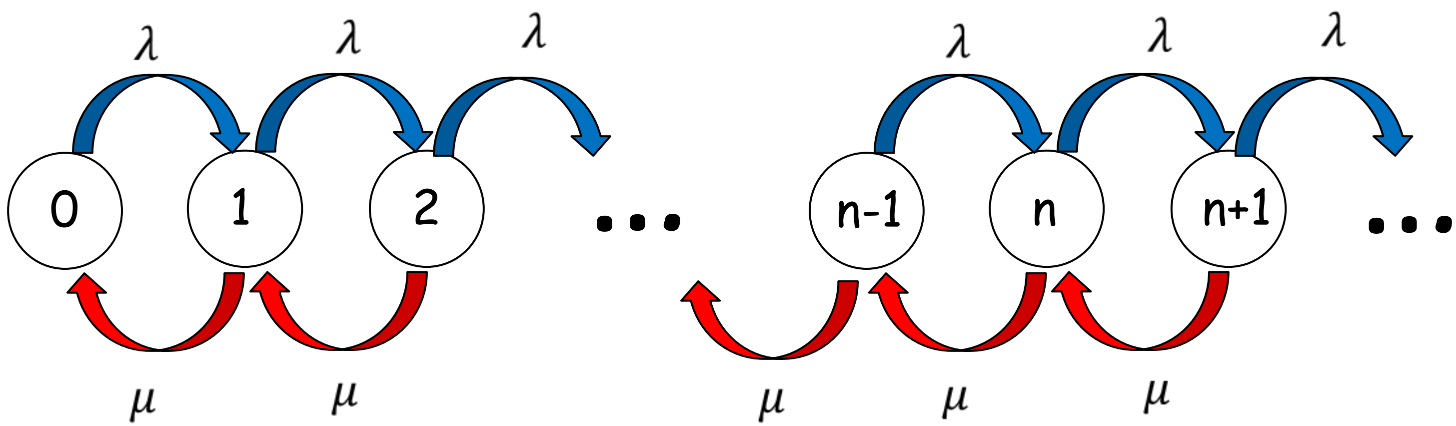
$$p_n = \frac{\lambda}{\mu} p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0 = (1-\rho)\rho^n$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0 \sum_{n=0}^{\infty} \rho^n = p_0 \frac{1}{1-\rho}$$

$$p_0 = 1 - \rho$$

$$N = E[n] = \sum_{n=0}^{\infty} n\, p_n = (1-\rho) \sum_{n=0}^{\infty} n\, \rho^n$$

M/M/1 Queue Example



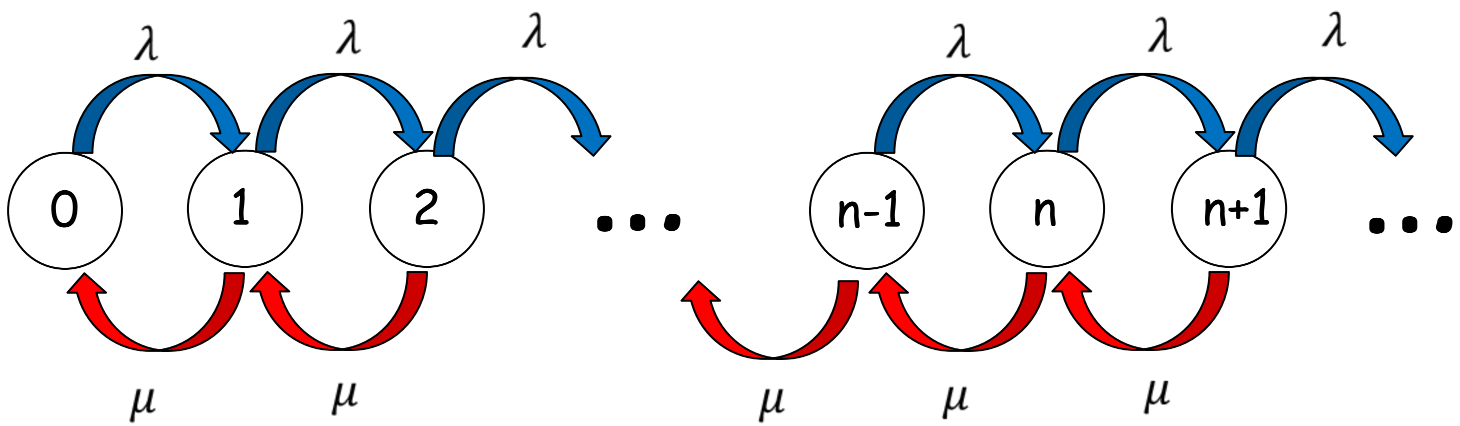
$$p_0\lambda = p_1\mu \qquad\qquad p_{n-1}\lambda = p_n\mu$$

$$\boxed{p_n = \frac{\lambda}{\mu}p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0 = (1-\rho)\rho^n}$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0 \sum_{n=0}^{\infty} \rho^n = p_0 \frac{1}{1-\rho} \Longrightarrow p_0 = 1-\rho$$

$$\boxed{N = E[n] = \sum_{n=0}^{\infty} n\, p_n = (1-\rho)\sum_{n=0}^{\infty} n\,\rho^n = (1-\rho)\rho\sum_{n=0}^{\infty} n\,\rho^{n-1}}$$

M/M/1 Queue Example



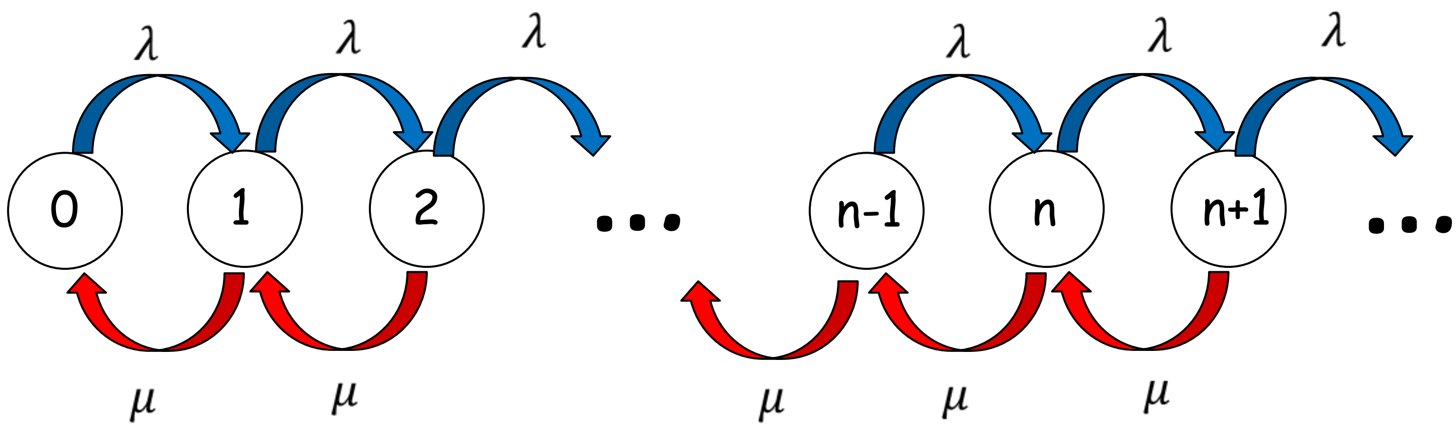
$$p_0\lambda = p_1\mu \qquad\qquad p_{n-1}\lambda = p_n\mu$$

$$\boxed{p_n = \frac{\lambda}{\mu}p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0 = (1-\rho)\rho^n}$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0 \sum_{n=0}^{\infty} \rho^n = p_0 \frac{1}{1-\rho} \Longrightarrow p_0 = 1-\rho$$

$$\begin{aligned} N = E[n] &= \sum_{n=0}^{\infty} n\, p_n = (1-\rho)\sum_{n=0}^{\infty} n\,\rho^n = (1-\rho)\rho\sum_{n=0}^{\infty} n\,\rho^{n-1} \\ &= (1-\rho)\rho\frac{d}{d\rho}\left(\sum_{n=0}^{\infty} \rho^n\right) \end{aligned}$$

M/M/1 Queue Example



$p_0\lambda = p_1\mu$

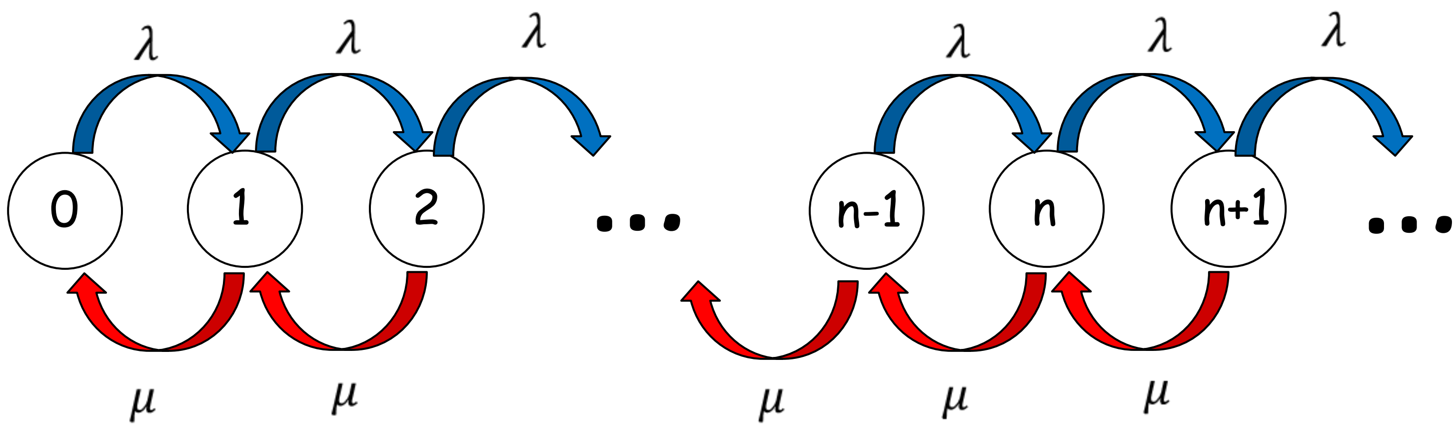
$p_{n-1}\lambda = p_n\mu$

$$\boxed{p_n = \frac{\lambda}{\mu}p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 \ = \rho^n p_0 = (1-\rho)\rho^n}$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0 \sum_{n=0}^{\infty} \rho^n = p_0 \frac{1}{1-\rho} \Longrightarrow p_0 = 1-\rho$$

$$\boxed{\begin{aligned} N = E[n] &= \sum_{n=0}^{\infty} n\, p_n = (1-\rho)\sum_{n=0}^{\infty} n\,\rho^n = (1-\rho)\rho \sum_{n=0}^{\infty} n\,\rho^{n-1} \\ &= (1-\rho)\rho\frac{d}{d\rho}\left(\sum_{n=0}^{\infty} \rho^n\right) = (1-\rho)\rho\frac{d}{d\rho}\left(\frac{1}{1-\rho}\right) \end{aligned}}$$

M/M/1 Queue Example



$$p_0\lambda = p_1\mu \qquad\qquad p_{n-1}\lambda = p_n\mu$$

$$\boxed{p_n = \frac{\lambda}{\mu}p_{n-1} = \left(\frac{\lambda}{\mu}\right)^2 p_{n-2} = \left(\frac{\lambda}{\mu}\right)^n p_0 = \rho^n p_0 = (1-\rho)\rho^n}$$

$$1 = \sum_{n=0}^{\infty} p_n = p_0 \sum_{n=0}^{\infty} \rho^n = p_0 \frac{1}{1-\rho} \Rightarrow p_0 = 1-\rho$$

$$\boxed{\begin{aligned} N = E[n] &= \sum_{n=0}^{\infty} n\, p_n = (1-\rho)\sum_{n=0}^{\infty} n\,\rho^n = (1-\rho)\rho\sum_{n=0}^{\infty} n\,\rho^{n-1} \\ &= (1-\rho)\rho\frac{d}{d\rho}\left(\sum_{n=0}^{\infty} \rho^n\right) = (1-\rho)\rho\frac{d}{d\rho}\left(\frac{1}{1-\rho}\right) = \frac{\rho}{1-\rho} \end{aligned}}$$

M/M/1 Queue Results

- Average number of customers in the system: $N = \rho / (1 - \rho)$
- Variance: $Var(N) = \rho / (1 - \rho)^2$

- Waiting time: $W = \rho / (\mu (1 - \rho))$
- Time in system: $T = 1 / (\mu (1 - \rho))$

  - Note: Little's Law: $N = \lambda T$

The M/D/1 Queue

- Assumes:
  - Poisson arrival process, **deterministic (constant) service times**, single server, FCFS service discipline, infinite capacity for storage, no loss
- Notation: **M/D/1**
  - Markovian arrival process (Poisson)
  - **Deterministic service times (constant)**
  - Single server (FCFS, infinite capacity)

M/D/1 Queue Results

- Average number of customers: $Q = \rho/(1 - \rho) - \rho^2 / (2 (1 - \rho))$
- Waiting time: $W = x \rho / (2 (1 - \rho))$ where x is the mean service time
- Note that lower variance in service time means less queueing occurs ☺
  - E.g., M/M/1 has $W = (1/\mu) \rho / (1 - \rho)$

Queueing Theory (cont'd)

- These simple models can be cascaded in series and in parallel to create arbitrarily large complicated queueing network models
- Two main types:
  - closed queueing network model (finite pop.)
  - open queueing network model (infinite pop.)
- Software packages exist for solving these types of models to determine steady-state performance (e.g., delay, throughput, util.)

Simulation Example: TCP Throughput

- Can use an existing simulation tool, or design and build your own custom simulator
- Example: ns-3 network simulator
  - A discrete-event simulator with detailed TCP protocol models
  - Configure network topology and workload
  - Run simulation using pseudo-random numbers and produce statistical output

OTHER ISSUES

- Simulation run length
  - choosing a long enough run time to get statistically meaningful results (equilibrium)
- Simulation start-up effects and end effects
  - deciding how much to “chop off” at the start and end of simulations to get proper results
- Replications
  - ensure repeatability of results, and gain greater statistical confidence in the results given

Experimental Example: Benchmarking

- The design of a performance study requires great care in experimental design and methodology
- Need to identify
  - experimental factors to be tested
  - levels (settings) for these factors
  - performance metrics to be used
  - experimental design to be used

FACTORS

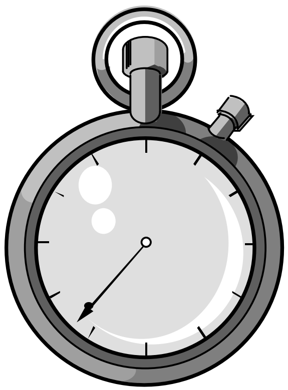
- Factors are the main "components" that are varied in an experiment, in order to understand their impact on performance
  - Examples: request rate, request size, read/write ratio, num concurrent clients
- Need to choose factors properly, since the number of factors affects size of study

LEVELS

- Levels are the precise settings of the factors that are to be used in an experiment
  - Examples: req size S = 1 KB, 10 KB, 1 MB
  - Example: num clients C = 10, 20, 30, 40, 50
- Need to choose levels realistically
- Need to cover useful portion of the design space

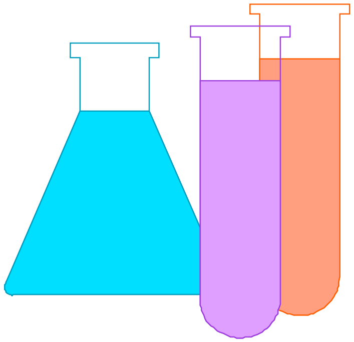
PERFORMANCE METRICS

- Performance metrics specify what you want to measure in your performance study
  - Examples: response time, throughput, pkt loss
- Must choose your metrics properly and instrument your experiment accordingly



EXPERIMENTAL DESIGN

- Experimental design refers to the organizational structure of your experiment
- Need to methodically go through factors and levels to get the full range of experimental results desired
- There are several "classical" approaches to experimental design



EXAMPLES

- One factor at a time
  - vary only one factor through its levels to see what the impact is on performance
- Two factors at a time
  - vary two factors to see not only their individual effects, but also their interaction effects, if any
- Full factorial
  - try every possible combination of factors and levels to see full range of performance results

SUMMARY

- Computer systems performance evaluation defines standard methods for designing and conducting performance studies
- Great care must be taken in experimental design and methodology if the experiment is to achieve its goal, and if results are to be fully understood
- Very many examples of these important methodologies and their applications …

Scalability example: Broadcast protocol

Streaming Popular Content

- Consider a popular media file
  - Playback rate: 1 Mbps
  - Duration: 90 minutes
  - Request rate: once every minute

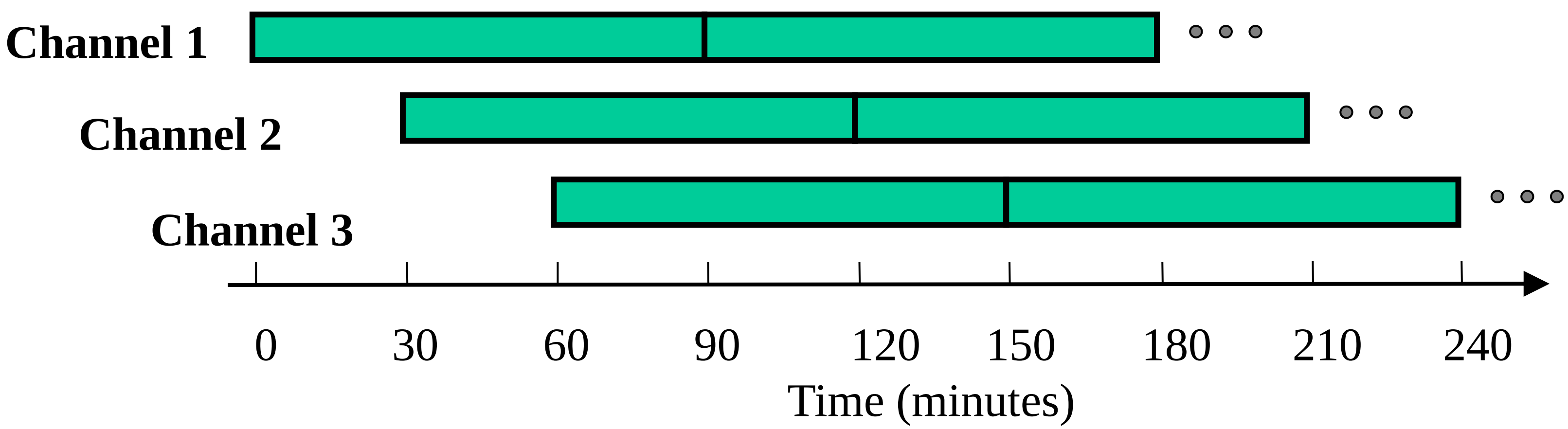
- How can a video server handle such high loads?
  - Approach 1: Start a new "stream" for each request
  - Allocate server and disk I/O bandwidth for each request
  - Bandwidth required at server= 1 Mbps x 90

Streaming Popular Content using Batching

- Approach 2: Leverage the multipoint delivery (e.g., multicast/broadcast) capability of modern networks
- Playback rate = 1 Mbps, duration = 90 minutes

Streaming Popular Content using Batching

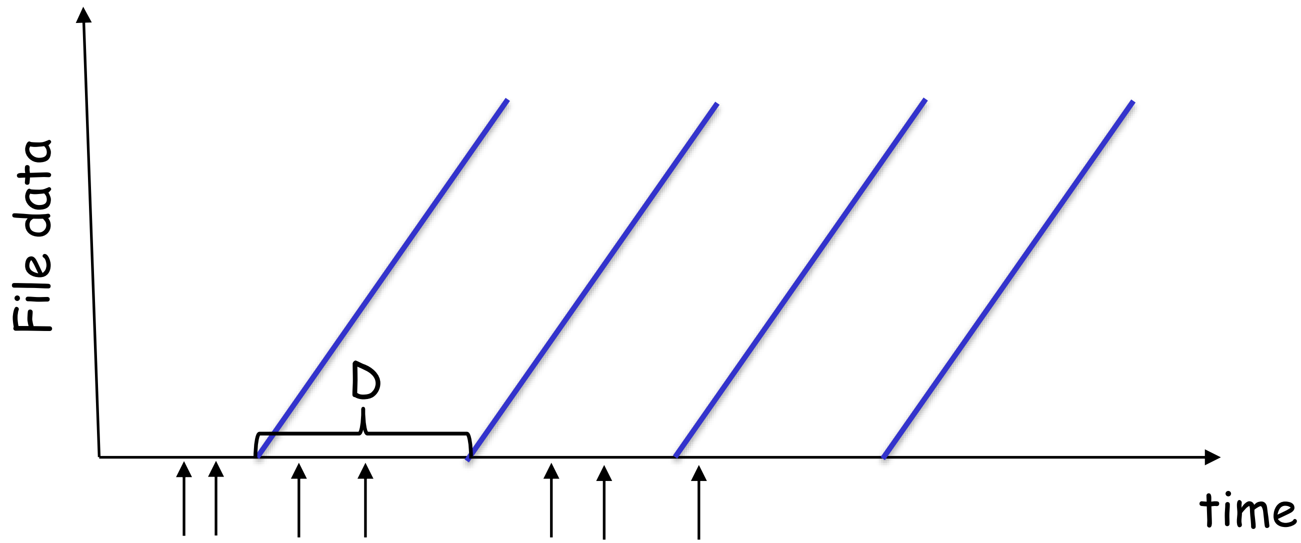
- Approach 2: Leverage the multipoint delivery (e.g., multicast/broadcast) capability of modern networks
- Playback rate = 1 Mbps, duration = 90 minutes
- Consider case of high request rate and D=30min…
  - Max. start-up delay = 30 minutes
  - Group requests in non-overlapping intervals of 30 min
  - Bandwidth required = 3 channels = 3 Mbps



Streaming Popular Content using Batching

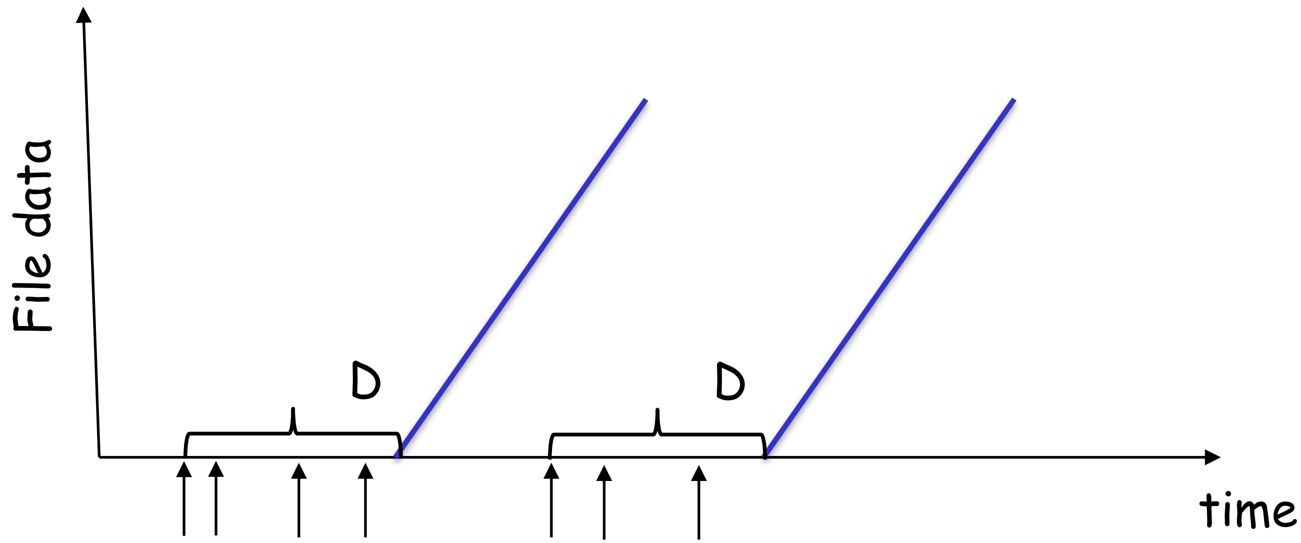
- Approach 2: Leverage the multipoint delivery (e.g., multicast/broadcast) capability of modern networks
- Playback rate = 1 Mbps, duration = 90 minutes
- An optimal batching protocol (and analysis)???
  - Define protocol
  -

N Carlsson, D. Eager, and M. K. Vernon, Multicast Protocols for Scalable On-demand Download, Performance Evaluation, 2006.



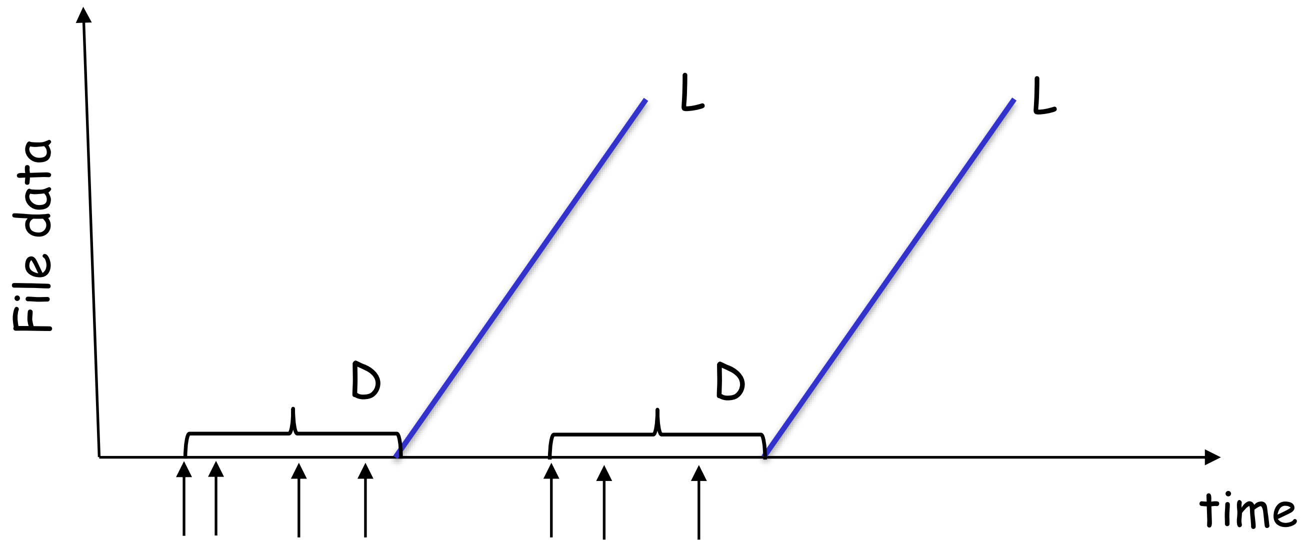
- An optimal batching protocol (and analysis)???
  - Define protocol
  -

N Carlsson, D. Eager, and M. K. Vernon, Multicast Protocols for Scalable On-demand Download, Performance Evaluation, 2006.



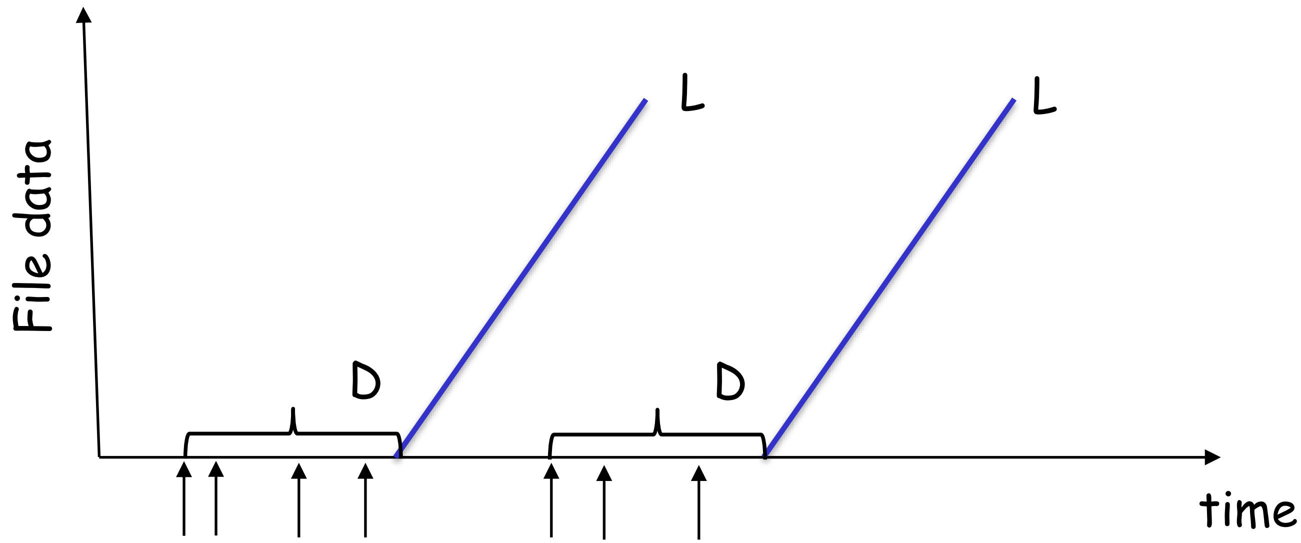
- An optimal batching protocol (and analysis)???
  - Define protocol
  - How to evaluate?
    - Analytically?
    - Simulations?
    - Experiments?

N Carlsson, D. Eager, and M. K. Vernon, Multicast Protocols for Scalable On-demand Download, Performance Evaluation, 2006.



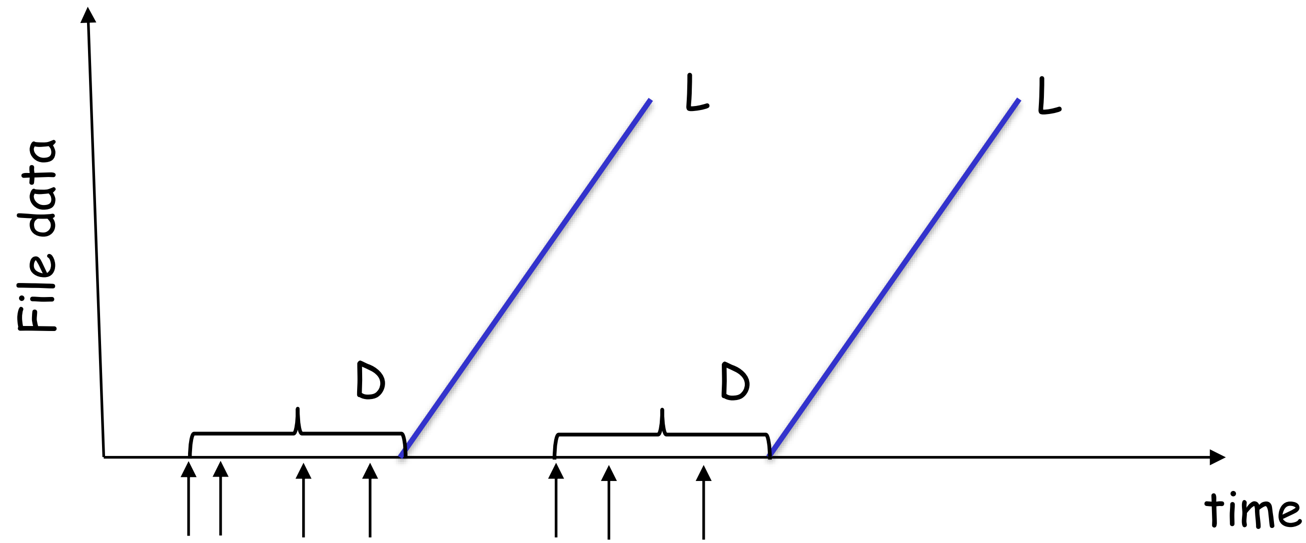
- Optimal batching protocol
  - Max delay = D
- Poisson process
  - Inter-arrival times (i) exponentially distributed and (ii) independent
  - Memory less arrival process

N Carlsson, D. Eager, and M. K. Vernon, Multicast Protocols for Scalable On-demand Download, Performance Evaluation, 2006.



- Renewal process
  - Identify and analyze “renewal periods” (statistically the same)
  - $B = L / (D+1/\lambda)$
- Poisson Arrivals See Time Average (PASTA) property
  - $A = [D(1+\lambda D/2)]/[1+\lambda D]$

N Carlsson, D. Eager, and M. K. Vernon, Multicast Protocols for Scalable On-demand Download, Performance Evaluation, 2006.



- Little's law
  - # in system = (arrival rate into system) x (average time in system)
- Systems considered where
  - System = "waiting queue" (for first bit)
    - Average time in system = A; Arrival rate = λ
    - E[# in system] = λ [D(1+λD/2)]/[1+λD]
  - System = "queue or being served" (to get all bits)
    - Average time in system A+L/r; Arrival rate = λ
    - E[#in system] = λ [D(1+λD/2)]/[1+λD] + λL/r

More slides …

Batching Issues

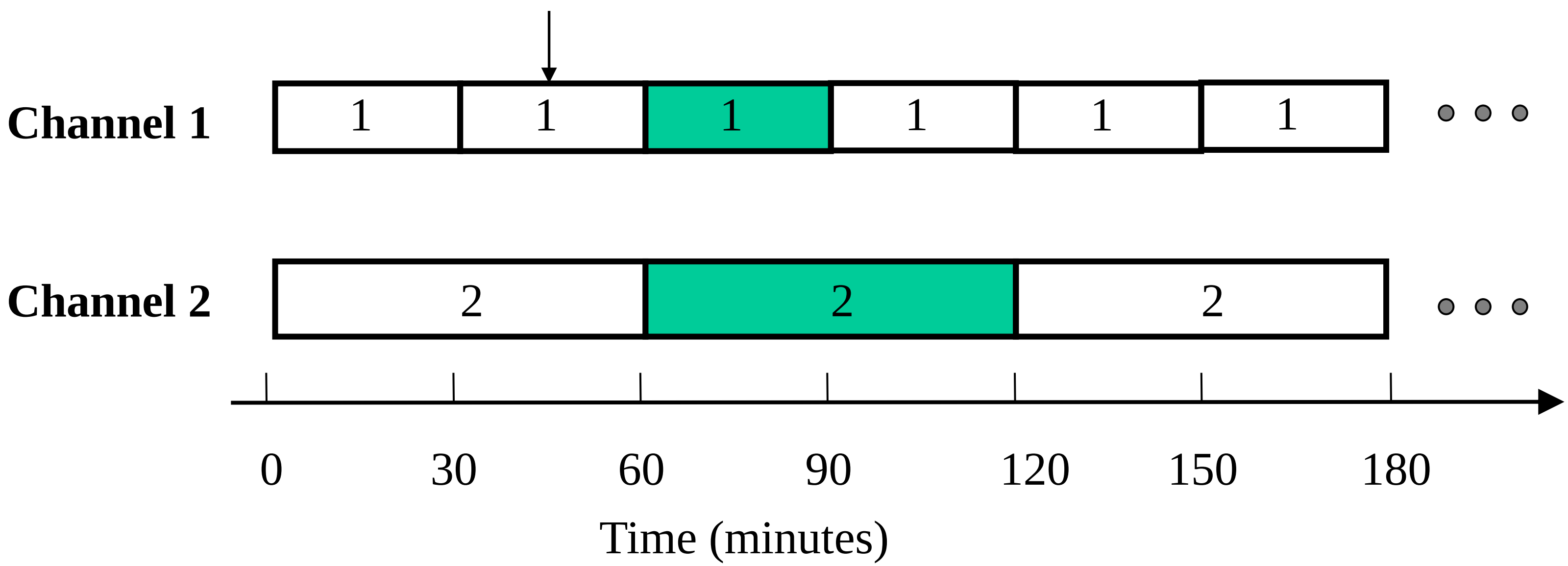
- Bandwidth increases linearly with decrease in start-up delays

- Can we reduce or eliminate "start-up" delays?
  - Periodic Broadcast Protocols

  - Stream Merging Protocols

Periodic Broadcast Example

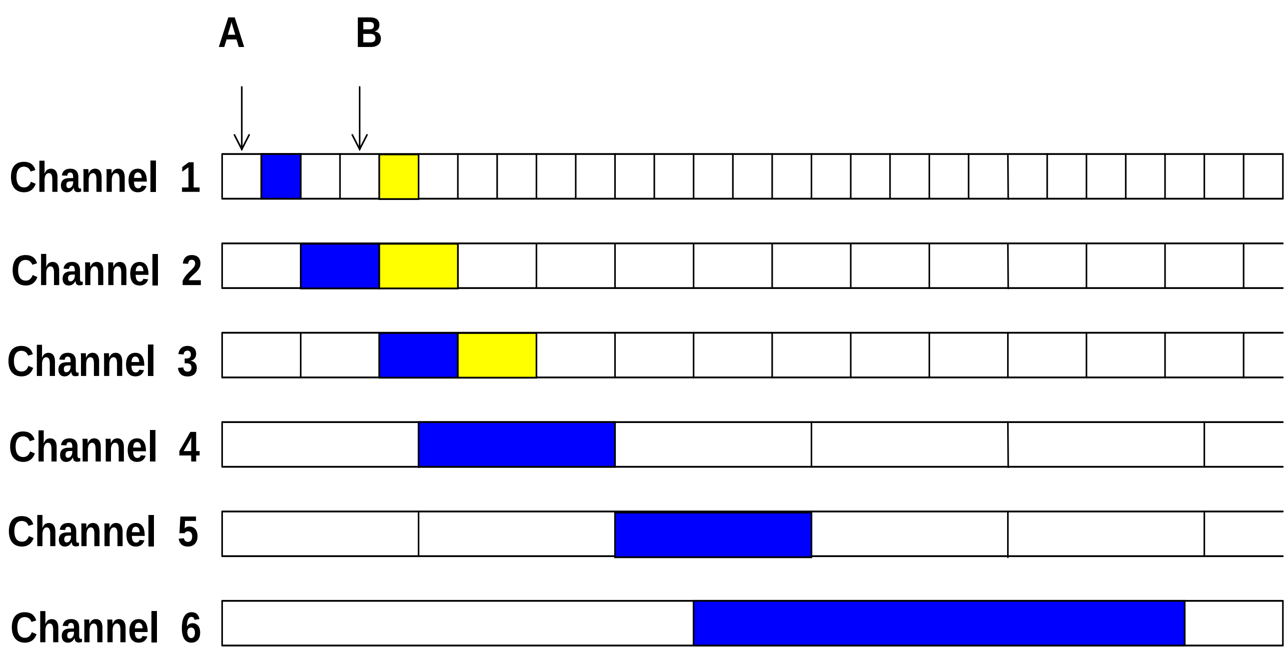
- Partition the media file into 2 segments with relative sizes {1, 2}. For a 90 min. movie:
  - Segment 1 = 30 minutes, Segment 2 = 60 minutes
- Advantage:
  - Max. start-up delay = 30 minutes
  - Bandwidth required = 2 channels = 2 Mbps
- Disadvantage: Requires increased client capabilities



Skyscraper Broadcasts (SB)

[Hua & Sheu 1997]

- Divide the file into *K* segments of increasing size
  - Segment size progression: 1, 2, 2, 5, 5, 12, 12, 25, …
- Multicast each segment on a separate channel at the playback rate
- Aggregate rate to clients: *2 x playback rate*



Comparing Batching and SB

Server Bandwidth	Start-up Delay	
	Batching	SB
1 Mbps	90 minutes	90 minutes
2 Mbps	45 minutes	30 minutes
6 Mbps	15 minutes	3 minutes
10 Mbps	9 minutes	30 seconds

- Playback rate = 1 Mbps, duration = 90 minutes
- Limitations of Skyscraper:
  - Ad hoc segment size progress
  - Does not work for low client data rates

Reliable Periodic Broadcasts (RPB)

[Mahanti *et al*. 2001, 2003, 2004]

- Optimized PB protocols (no packet loss recovery)
  - client fully downloads each segment before playing
  - required server bandwidth near minimal
  - Segment size progression is *not* ad hoc
  - Works for client data rates < 2 x playback rate
- extend for packet loss recovery
- extend for "bursty" packet loss
- extend for client heterogeneity

Reliable Periodic Broadcasts (RPB)

[Mahanti *et al*. 2001, 2003, 2004]

- Optimized PB protocols (no packet loss recovery)
  - client fully downloads each segment before playing
  - required server bandwidth near minimal
  - Segment size progression is *not* ad hoc
  - Works for client data rates < 2 x playback rate
- extend for packet loss recovery
- extend for "bursty" packet loss
- extend for client heterogeneity

Optimized Periodic Broadcasts

- Playback rate assumed equal to 1
- r = segment streaming rate
- s = maximum # streams client listens to concurrently
- b = client data rate = s x r

Optimized Periodic Broadcasts

- Playback rate assumed equal to 1
- r = segment streaming rate
- s = maximum # streams client listens to concurrently
- b = client data rate = s x r

Optimized Periodic Broadcasts

- Playback rate assumed equal to 1
- r = segment streaming rate = 1
- s = maximum # streams client listens to concurrently = 2
- b = client data rate = s x r = 2
- length of first s segments: $\frac{1}{r} l_k = \frac{1}{r} l_1 + \sum_{j=1}^{k-1} l_j$
- length of segment k > s: $\frac{1}{r} l_k = \sum_{j=k-s}^{k-1} l_j$

Comparison with Skyscraper

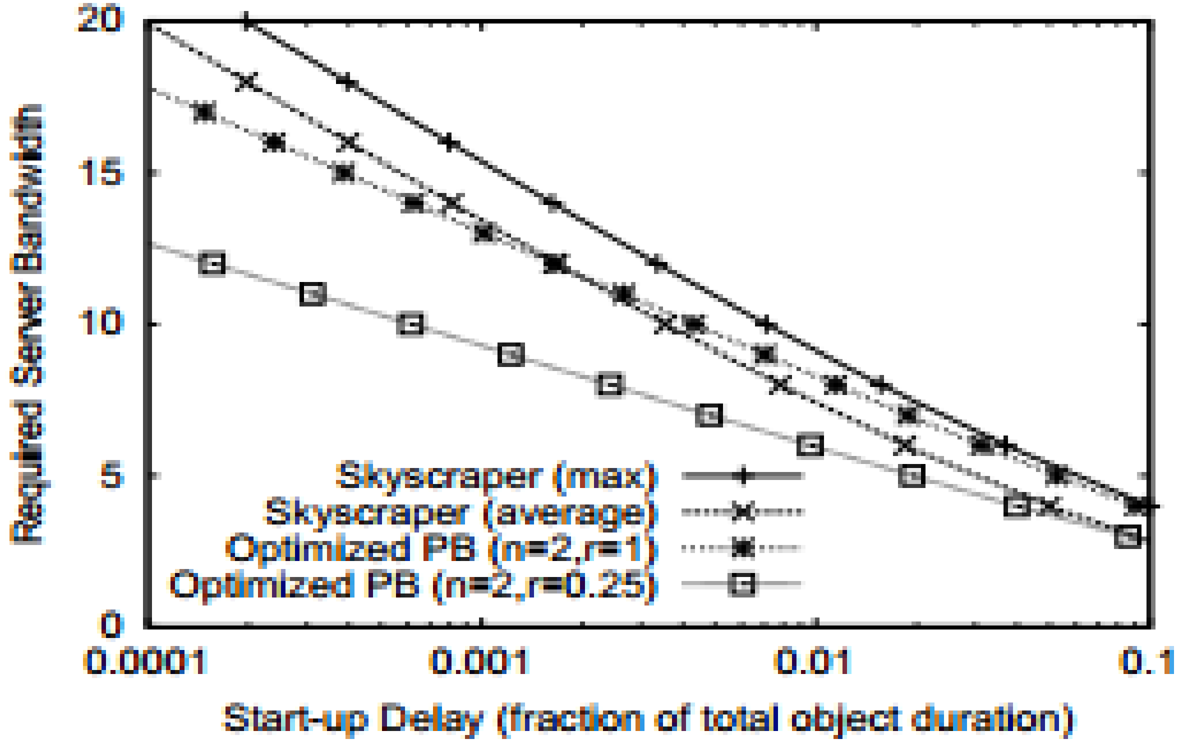


Figure 5: Performance Comparison of Optimized PB and Skyscraper Broadcasts

Immediate service: Hierarchical Stream Merging

D. Eager, M. Vernon, and J. Zahorjan, "Minimizing Bandwidth Requirements for On-Demand Data Delivery", IEEE TKDE, 2001.

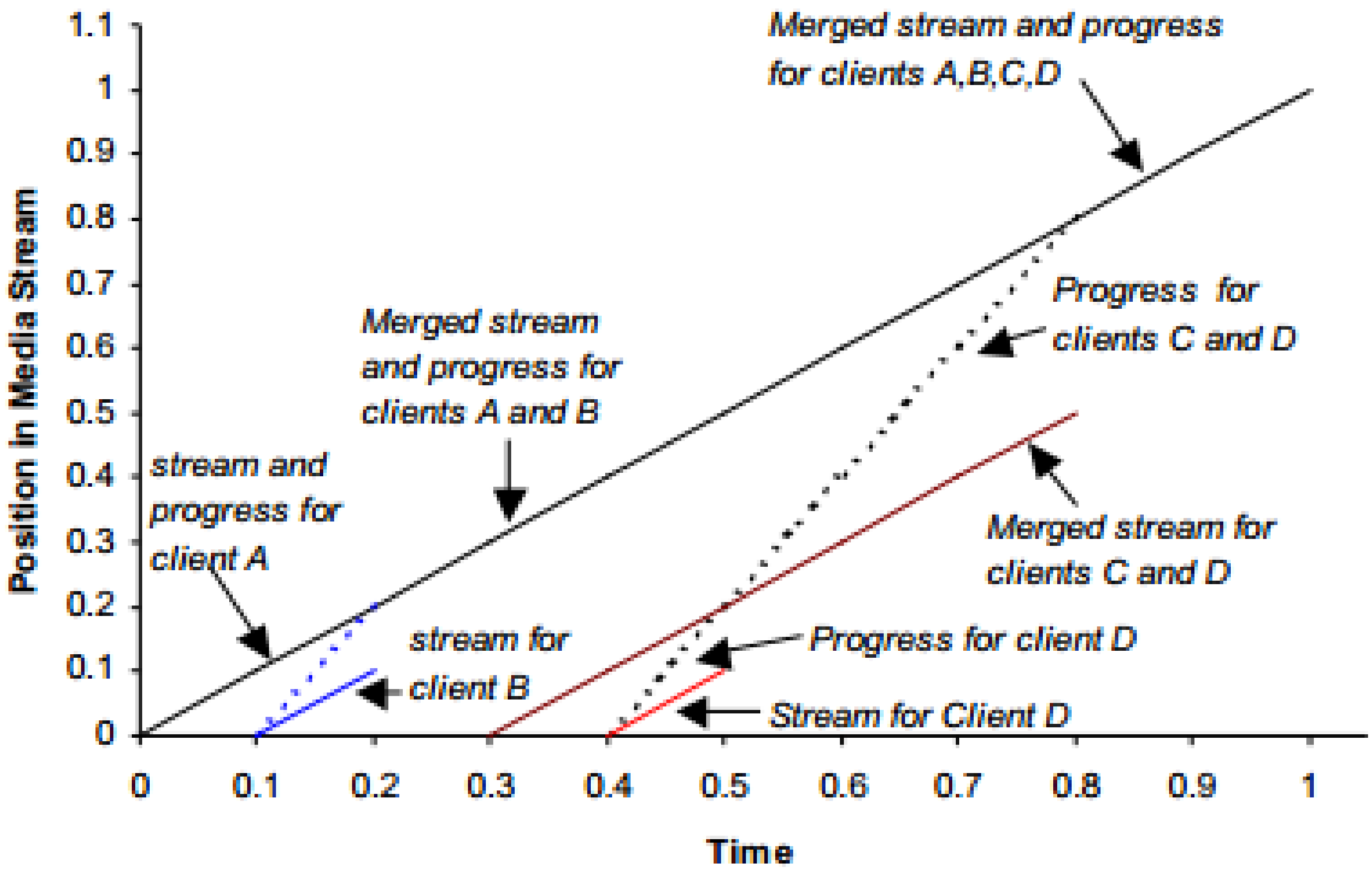


Figure 5: Example of Hierarchical Multicast Stream Merging