Lecture 14: Performance Analysis

CS/ECE 438: Communication Networks

Prof. Matthew Caesar

May 5, 2010

How to evaluate a network design?

- Implementation and testbed/field deployment
  - Pros: high accuracy
  - Cons: costly, difficult to repair/experiment in-field
- Simulations
  - Pros: can be accurate, given realistic models; broad applicability
  - Cons: can be slow, don't always provide intuition behind results
- Analytical results
  - Pros: Quick answers, provides insights
  - Cons: Can be inaccurate or inapplicable

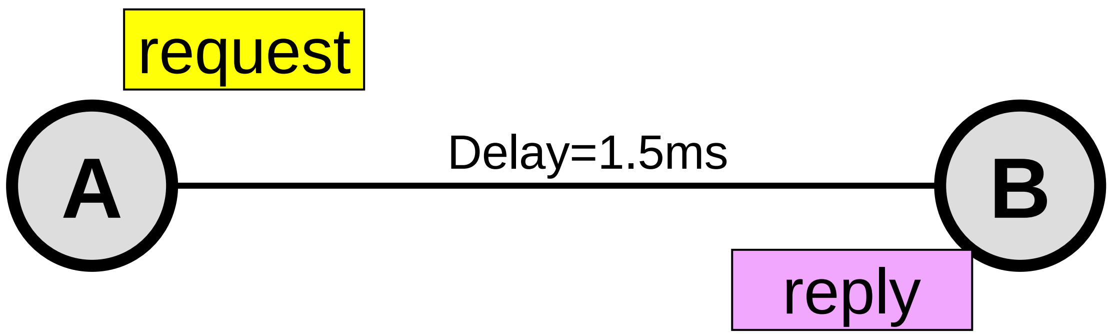
Simulation

- Build an “imitation” of the network that runs on a computer
  - Can be studied to estimate how system would operate in real network
  - Can change variables, replay different workloads perform experiments, to predict and learn behavior of the system
- Useful for situations too complex to analytically model

One approach: Discrete Event Simulation

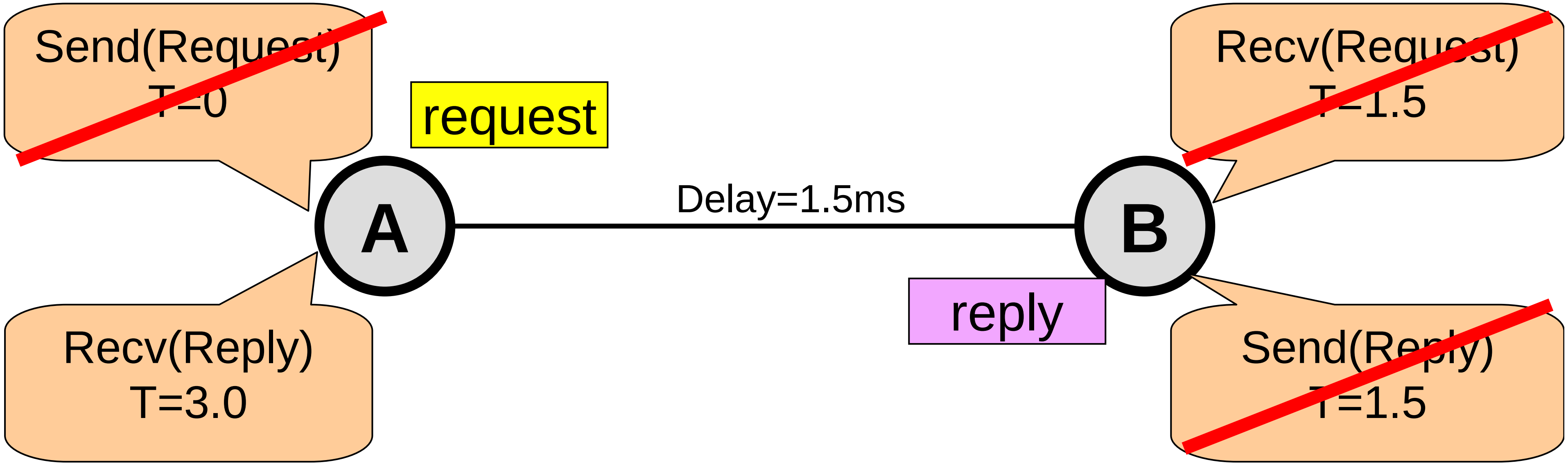
- Operation of the system is represented as a chronological sequence of events
- Each event occurs at an instant of time, can trigger new events to be generated
- Composed of:
  - Clock: current simulation time
  - Event list: list of future events that will occur, sorted by occurrence time
  - Event handlers: function called when event is "executed", may trigger new event to be placed onto list

Discrete Event Simulation: Example



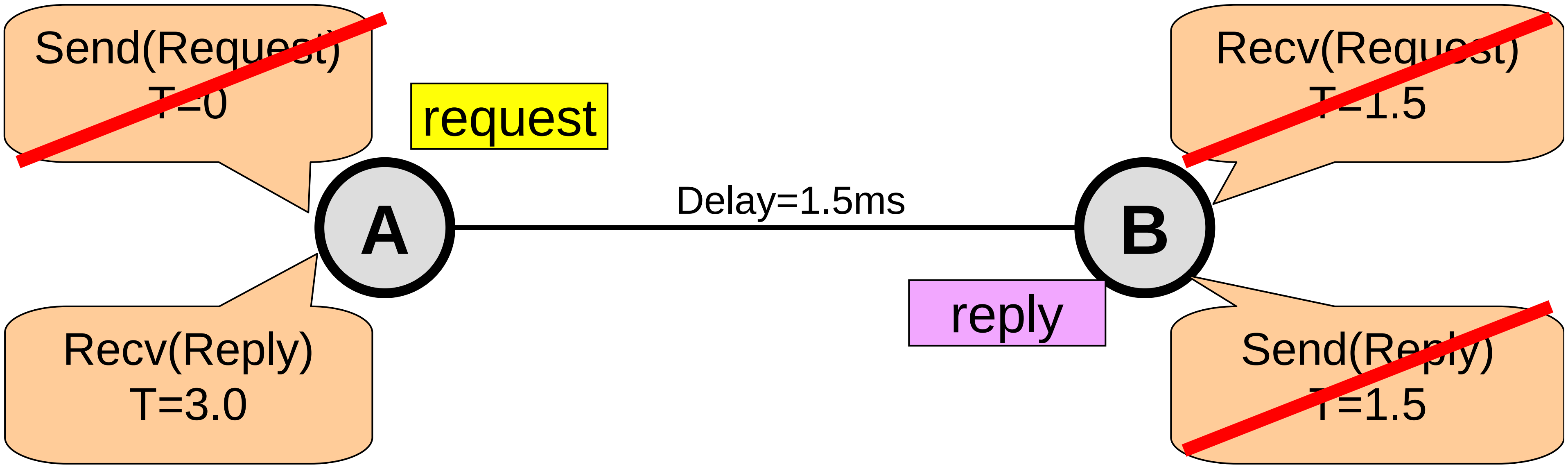
- Example: Simple ping protocol
- Host A sends echo request to Host B, Host B responds with echo reply
- What time does A receive the reply?

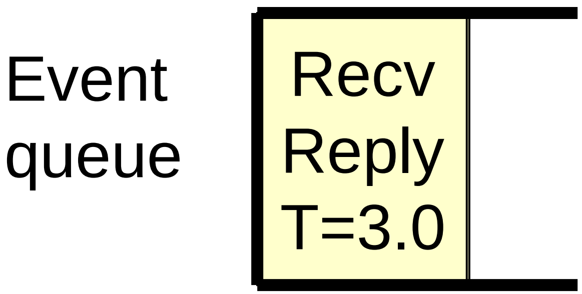
Discrete Event Simulation: Example



- Each event takes place at a certain time
- Algorithm: when processing an event, figure out when the next event will happen, and put it in the queue

Discrete Event Simulation: Example





Event Handler:

```
Recv_Reply(Node A, Time 3.0) {
    printf(“pkt recvd at time %i\n”,Time);
}
```

Analysis

- Write down a set of formulas describing relationships between components
- Plug in numbers to estimate system performance in different settings
- Equations provide insight into underlying characteristics
  - Also, simple/quick to apply
- But, some systems are too complex to analytically model
  - Luckily, a lot of important properties of a lot of important systems can be characterized through analysis

Motivating example

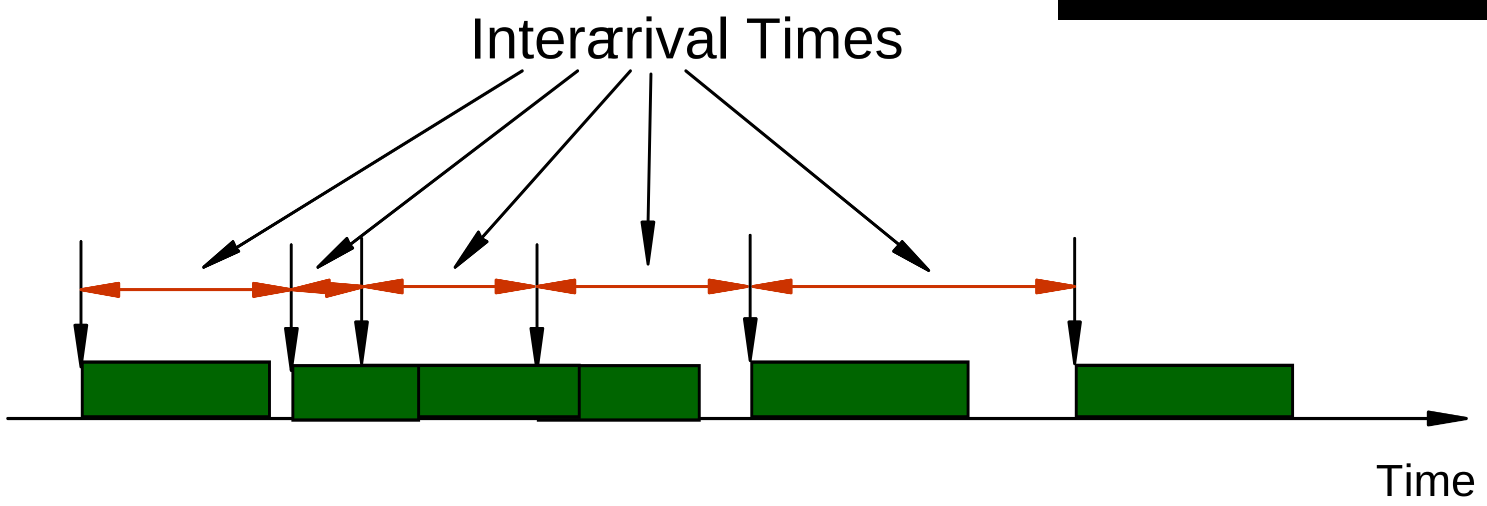
- Suppose you’re sitting on the side of the road watching cars go by
- Suppose you see a big burst of cars come by
- After the burst: does the likelihood new cars will come increase or decrease?

Motivating example

- After the burst: does the likelihood new cars will come increase or decrease?
- Answer: neither!
- Reason: Car arrival times are (reasonably) well modeled by a Poisson Process
- The Poisson distribution is "memoryless" (history gives no information about future events)
- A distribution is memoryless iff:
  - $\Pr(X>m+n \mid X>m) = \Pr(X>n)$

Poisson Process

- Interarrival times are independent and exponentially distributed
- Models well the accumulated traffic [redacted] many independent sources
- The average interarrival time is 1/λ (secs/packet), so l is the arrival rate (packets/sec)



Poisson Process

- A stochastic (random) process, where
  - Events occur continuously and independently of each other
- Composed of a collection of $\{N(t) : t \geq 0\}$ random variables, where $N(t)$ is number of events at time t
  - Number of events between times A and B is $N(B)-N(A)$
- Probability distribution of $N(t)$ is a Poisson distribution

Poisson Process

- Very useful, accurate model for an extremely large class of real events:
  - Arrival of customers in a queue
  - Arrival of HTTP sessions/VoIP calls/etc. at a server
  - Number of raindrops falling in an area
  - Number of photons hitting a photodetector
  - Number of telephone calls at a switchboard
  - Number of particles emitted by radioactive decay of an unstable substance

Poisson Distribution

- Probability distribution of N(t) is a Poisson distribution
- The probability that there are
  - n occurrences,
  - given an arrival rate of λ
- Is:

$$f(n;\lambda) = \frac{\lambda^n e^{-\lambda}}{n!},$$

Poisson Process

- We can use Poisson Process to find expected number of arrivals in an interval

$$P[(N(t+\tau)-N(t))=k]=\frac{e^{-\lambda\tau}(\lambda\tau)^k}{k!} \qquad k=0,1,\ldots,$$

- Where
  - N(t+τ)-N(t) is the number of events in the time interval [t+τ, t]

Poisson Process: Example

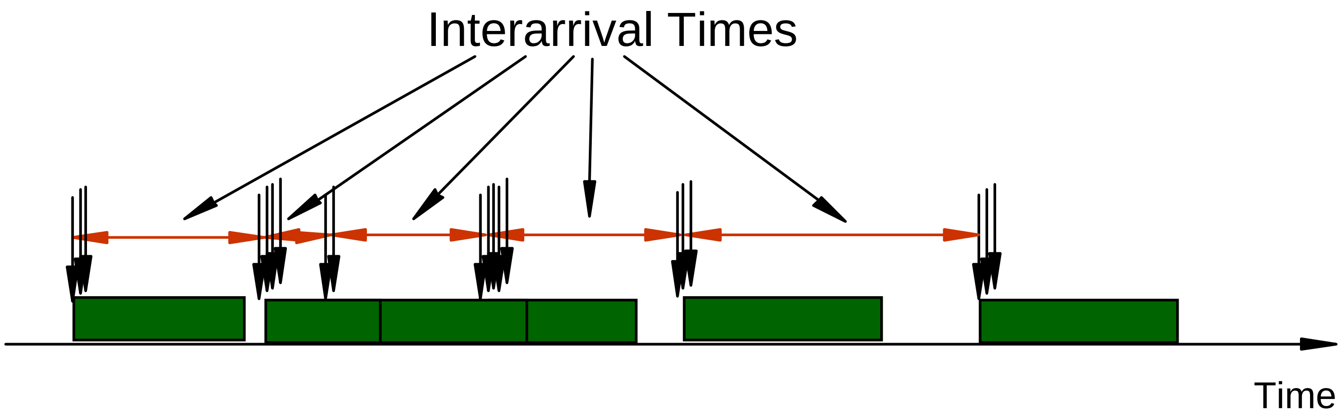
- Example: suppose
  - Cars arrive with rate λ=4 cars/minute
  - Suppose is it Noon on April 14th
  - What is probability that k=7 cars arrive within a 2 minute period?

$$P[(N(t+\tau) - N(t)) = k] = \frac{e^{-\lambda\tau}(\lambda\tau)^k}{k!} \qquad k = 0, 1, \ldots,$$

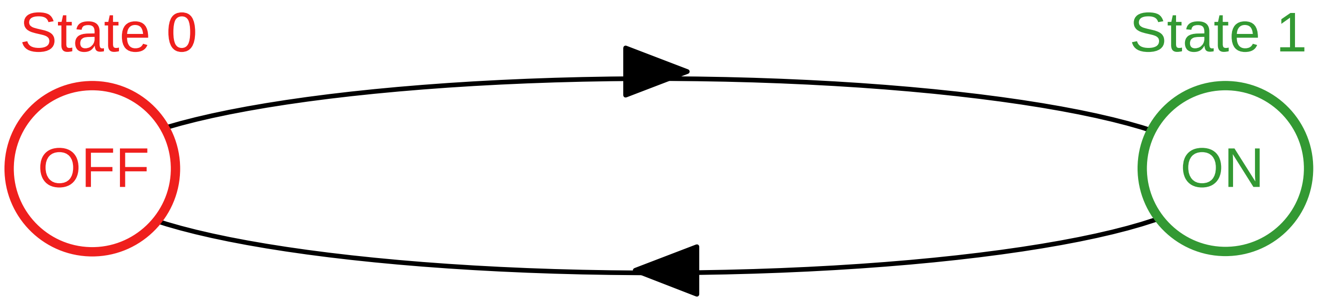
- Pr[N("Noon on Apr 14" + 2)-
  N("Noon on Apr 14" + 0)]
- = Pr[N(2)-N(0)]
- Pr[N(2)-N(0)]=e^(-4*2)*(4*2)^7/(7!)
- =0.139=14%

Variant on Poisson: Batch Arrivals

- Some sources transmit in packet bursts
- May be better modeled by a batch arrival process (e.g., bursts of packets arriving according to a Poisson process)
- The case for a batch model is weaker at queues after the first hop, because of shaping



Markov Modulated Rate Process (MMRP)

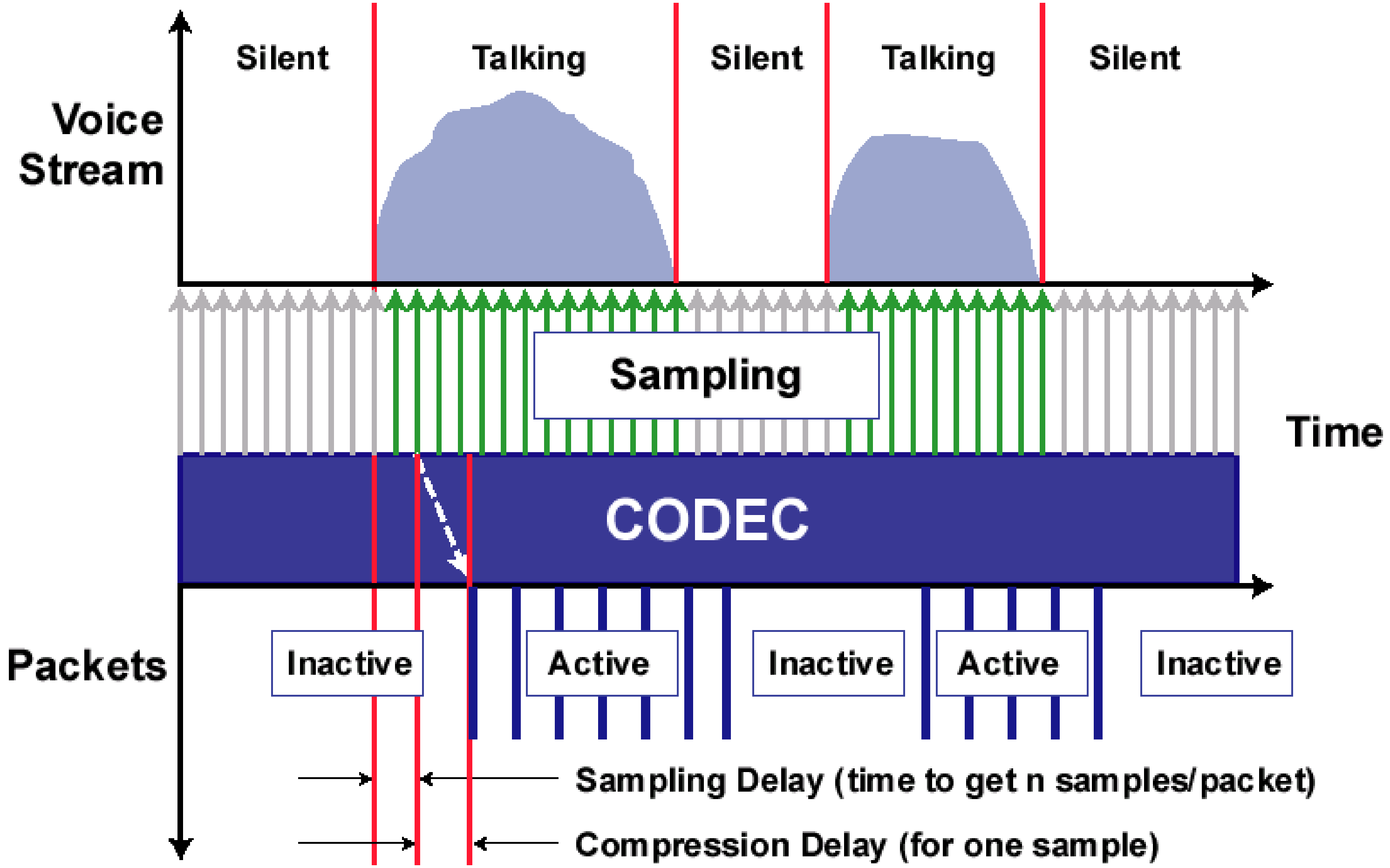


- An "on-off" model for traffic
  - E.g., a VoIP sender with silence suppression
- Stay in each state an exponentially distributed time
  - Transmit according to different model (e.g., Poisson, deterministic, etc) at each state
- Extension: models with more than two states

Source type properties

	Characteristics	QoS Requirements	Model
Voice	* Alternating talk-spurts and silence intervals. * Talk-spurts produce constant packet-rate traffic	**Delay < ~150 ms** Jitter < ~30 ms Packet loss < ~1%	* Two-state (on-off) Markov Modulated Rate Process (MMRP) * Exponentially distributed time at each state
Video	* Highly bursty traffic (when encoded) * Long range dependencies	Delay < ~ 400 ms **Jitter < ~ 30 ms** Packet loss < ~1%	K-state (on-off) Markov Modulated Rate Process (MMRP)
Interactive *BitTorrent* *ssh* *web*	* Poisson type * Sometimes batch-arrivals, or bursty, or sometimes on-off	Zero or near-zero packet loss Delay may be important	Poisson, Poisson with batch arrivals, Two-state MMRP

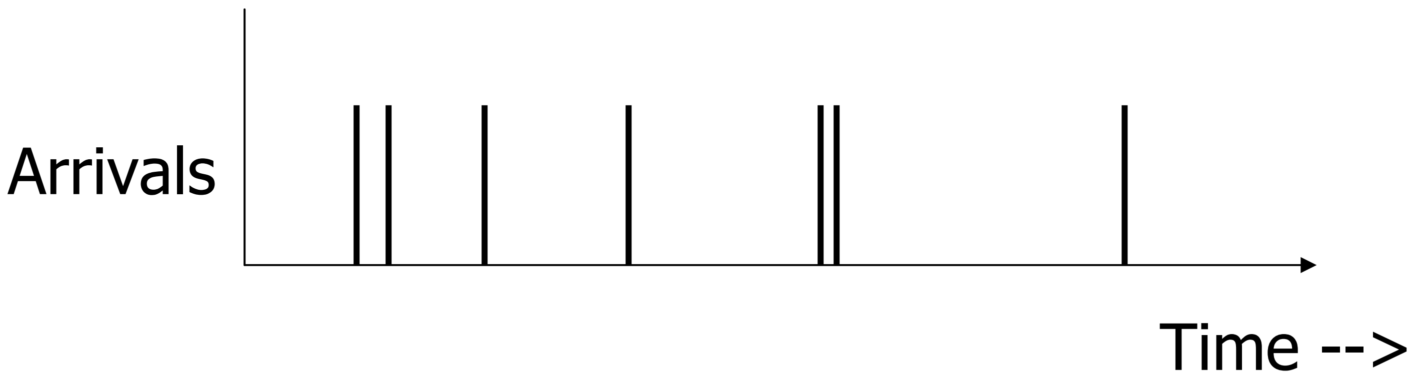
Typical voice source behavior



Motivating example 2

- Suppose we arrive at a bus stop. Suppose we know buses arrive randomly with average interarrival time 10 minutes.
- Suppose you walk up at a random time
- How long will you have to wait, on average, before a bus arrives?

Motivating example 2



- Suppose we arrive at a bus stop. Suppose we know buses arrive randomly with average interarrival time 10 minutes.
- Suppose you walk up at a random time
- How long will you have to wait, on average, before a bus arrives?

Motivating example 2

- How long will you have to wait, on average, before a bus arrives?
- Answer: 10 minutes
- Reason: distribution is memoryless
  - Just because there were 5 minutes without a bus before you got there, has nothing to do with how much longer you'll have to wait
- Related example: Average lifespan is 78 years. If you meet a 77 year old, his expected lifespan is not 78 years.

Motivating example 3

- Suppose you own a bank
  - Customers arrive with rate 30 customers/hour
  - Each customer takes on average 6 minutes to be serviced by the teller
  - You don't know anything about the distribution
- How many customers will be standing in line, on average?

Motivating example 3

- How many customers will be standing in line, on average?
- Answer: 30 customers per hour * 1/10 hours per customer = 3
- Reason:
  - The length of the queue is proportional to the average service time and the average arrival rate
  - In fact, it's equal to the two multiplied together – regardless of arrival distribution!

Analysis: Little's Law

- For a given arrival rate, the time in the system is proportional to packet occupancy
  - $N = \lambda T$
- where
  - N: average # of packets in the system
  - λ : packet arrival rate (packets per unit time)
  - T: average delay (time in the system) per packet

- Examples:
  - On rainy days, streets and highways are more crowded
  - Fast food restaurants need a smaller dining room than regular restaurants with the same customer arrival rate
  - Large buffering together with large arrival rate cause large delays
  - If you see a long line that you're thinking of joining, and you can guess the arrival rate, you can estimate how long you'll wait in that line

Queuing Theory

- What we’ve been discussing so far is known as Queuing Theory
  - Mathematical study of waiting lines (queues)

- Extensions can handle more complex analyses
  - Modeling departure rate from queue
  - Modeling non-Poisson arrival distributions
  - Modeling networks of queues

M/M/1 System

- Nomenclature: M stands for “Memoryless” (a property of the exponential distribution)
  - **M**/M/1 stands for Poisson arrival process (which is memoryless)
  - M/**M**/1 stands for exponentially distributed transmission times
- Assumptions:
  - Arrival process is Poisson with rate λ packets/sec
  - Packet transmission times are exponentially distributed with mean $1/\mu$
  - One server
  - Independent interarrival times and packet transmission times
- Transmission time is proportional to packet length
- Note $1/\mu$ is secs/packet so μ is packets/sec (packet transmission rate of the queue)
- Utilization factor: $\rho = \lambda/\mu$ (stable system if $\rho < 1$)

Delay calculation for M/M/1 system

- Let

  Q = Average time spent waiting in queue

  T = Average packet delay (transmission plus queuing)
- Note that $T = 1/\mu + Q$
- Also by Little's law

  $N = \lambda T$ and $N_q = \lambda Q$

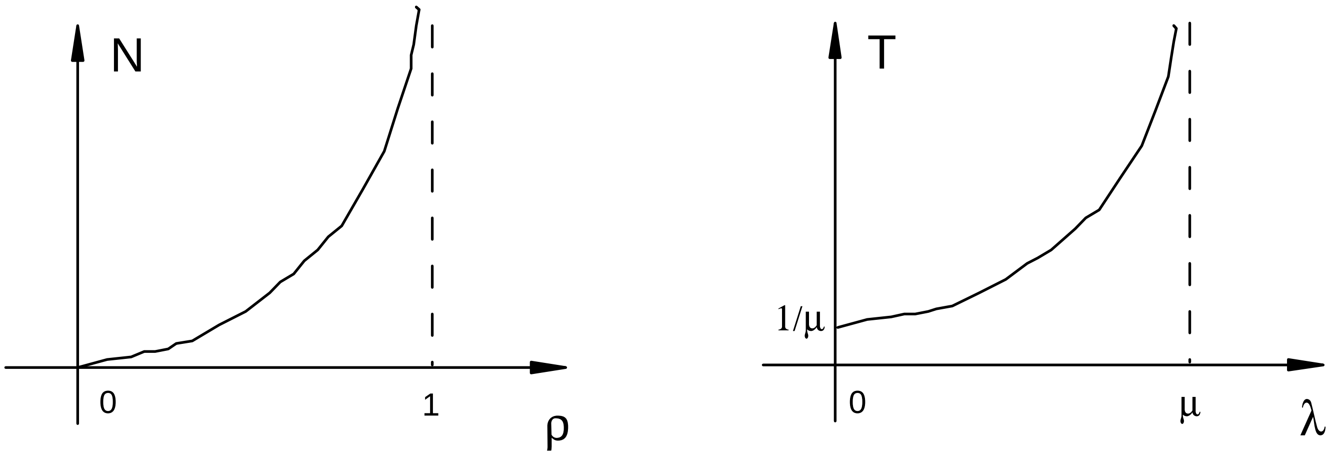
where

  N_q = Average number waiting in queue
- These quantities can be calculated with formulas described previously

M/M/1 Results

- The analysis gives the steady-state probabilities of number of packets in queue or transmission
- P{n packets} = $\rho^n(1-\rho)$ where $\rho = \lambda/\mu$
- From this we can get the averages:

$$N = \rho/(1 - \rho)$$
$$T = N/\lambda = \rho/\lambda(1 - \rho) = 1/(\mu - \lambda)$$



Example: How Delay Scales with Bandwidth

- Occupancy and delay formulas

  $N = \rho/(1 - \rho)$ $T = 1/(\mu - \lambda)$ $\rho = \lambda/\mu$
- Assume:
  - Traffic arrival rate λ is doubled
  - System transmission capacity μ is doubled
- Then:
  - Queue sizes stay at the same level (ρ stays the same)
  - Packet delay is cut in half (μ and λ are doubled)
- A conclusion: In high speed networks
  - propagation delay increases in importance relative to delay
  - buffer size and packet loss may still be a problem

M/M/m, M/M/∞ System

- Same as M/M/1, but it has m (or ∞) servers
- In M/M/m, the packet at the head of the queue moves to service when a server becomes free
- Qualitative result
  - Delay increases to ∞ as $\rho = \lambda/m\mu$ approaches 1
- There are analytical formulas for the occupancy probabilities and average delay of these systems

Finite Buffer Systems: M/M/m/k

- The M/M/m/k system
  - Same as M/M/m, but there is buffer space for at most k packets. Packets arriving at a full buffer are dropped

- Formulas for average delay, steady-state occupancy probabilities, and loss probability

- The M/M/m/m system is used widely to size telephone or circuit switching systems

Characteristics of M/M/. Systems

- Advantage: Simple analytical formulas
- Disadvantages:
  - The Poisson assumption may be violated
  - The exponential transmission time distribution is an approximation at best
  - Interarrival and packet transmission times may be dependent (particularly in the network core)
  - Head-of-the-line assumption precludes heterogeneous input traffic with priorities (hard or soft)

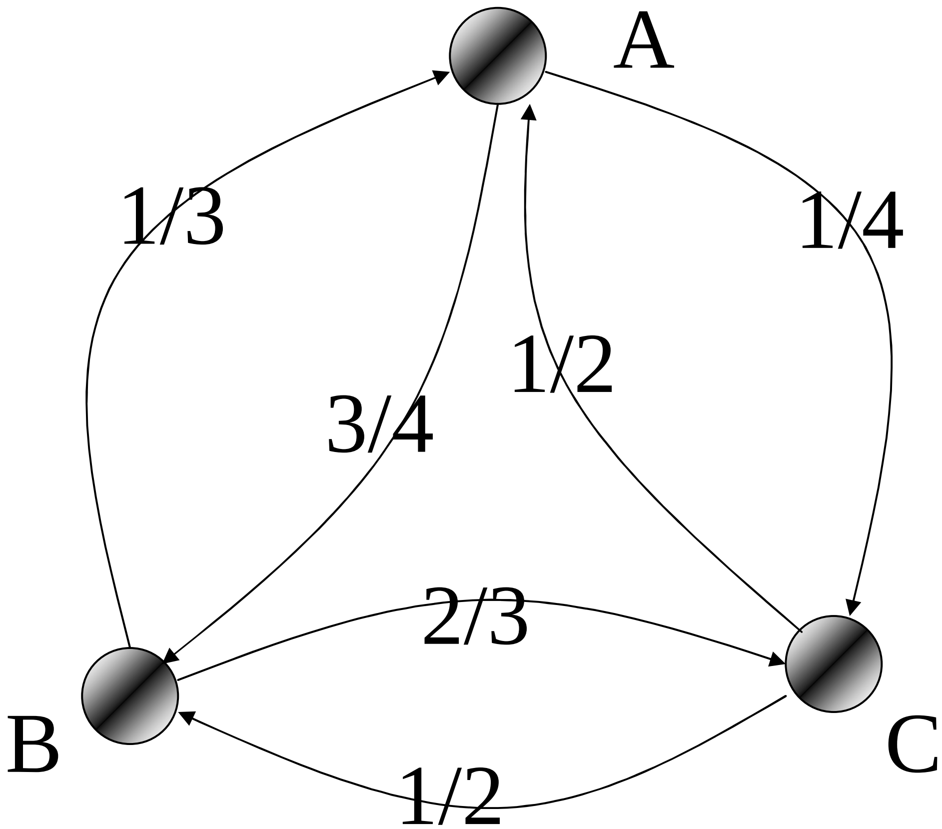
M/G/1 System

- Same as M/M/1 but the packet transmission time distribution is general, with given mean $1/\mu$ and variance σ^2
- Utilization factor $\rho = \lambda / \mu$
- Pollaczek-Kinchine formula for

  Average time in queue = $\lambda(\sigma^2 + 1/\mu^2)/2(1- \rho)$

  Average delay = $1/\mu + \lambda(\sigma^2 + 1/\mu^2)/2(1- \rho)$
- The formulas for the steady-state occupancy probabilities are more complicated
- Insight: As σ^2 increases, delay increases

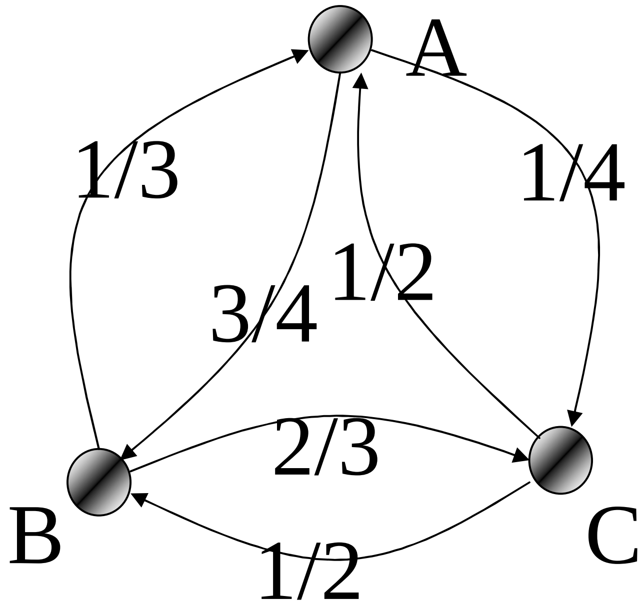
Visualising Markov Chains (the confused hippy hitcher example)



A hitchhiking hippy begins at A town. For some reason he has poor short-term memory and travels at random according to the probabilities shown. What is the chance he is back at A after 2 days? What about after 3 days? Where is he likely to end up?

The Hippy Hitcher (continued)

- After 1 day he will be in B town with probability 3/4 or C town with probability 1/4
- The probability of returning to A via B after 1 day is 3/12 and via C is 1/8 total 3/8
- We can perform similar calculations for 3 or 4 days but it will quickly become tricky and finding which city he is most likely to end up in is impossible.



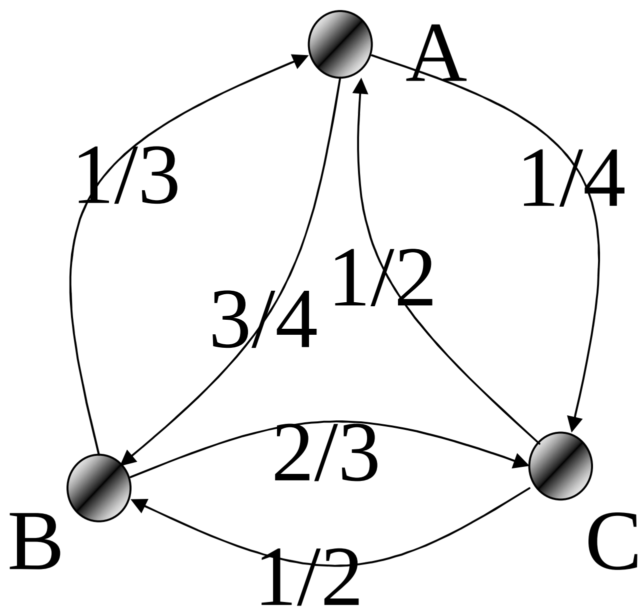
Transition Matrix

- Instead we can represent the transitions as a matrix

Prob of going to B from A

$$P = \begin{bmatrix} 0 & 3/4 & 1/4 \\ 1/3 & 0 & 2/3 \\ 1/2 & 1/2 & 0 \end{bmatrix}$$

Prob of going to A from C



Markov Chain Transition Basics

- p_{ij} are the transition probabilities of a chain. They have the following properties:

$$p_{ij} \geq 0, \sum_{j=0}^{\infty} p_{ij} = 1, \qquad i = 0,1\ldots$$

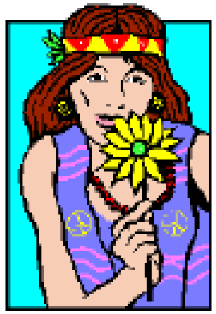
- The corresponding probability matrix is:

$$P = \begin{bmatrix} p_{00} & p_{01} & p_{02} & \cdots & p_{0n} \\ p_{10} & p_{11} & p_{12} & \cdots & p_{1n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{n0} & p_{n1} & p_{n2} & \cdots & p_{nn} \end{bmatrix}$$

Transition Matrix

- Define λ_n as a distribution vector representing the probabilities of each state at time step *n*.
- We can now define 1 step in our chain as: $\lambda_{n+1} = \lambda_n P$
- And clearly, by iterating this, after m steps we have:
  - $\lambda_{n+m} = \lambda_n P^m$

The Return of the Hippy Hitcher



- What does this imply for our hippy?
- We know the initial state vector:
  - λ_0=[1 0 0]
- So we can calculate λ_n with a little drudge work.
- (If you get bored raising P to the power n then you can use a computer)
- But which city is the hippy likely to end up in?
- We want to know $\pi = \lim_{n \to \infty} \lambda_n$

Invariant (or equilibrium) probabilities)

$$\pi = \lim_{n \to \inf} \lambda_n$$

- Assuming the limit exists, the distribution vector π is known as the invariant or equilibrium probabilities.
- We might think of them as being the proportion of the time that the system spends in each state or alternatively, as the probability of finding the system in a given state at a particular time.
- They can be found by finding a distribution which solves the equation $\pi = \pi P$

Example: Weather Prediction

- Suppose the weather, given the preceeding day, is given by the matrix

$$P = \begin{array}{cc} \begin{array}{cc} \textit{sun} & \textit{rain} \end{array} & \\ \begin{pmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{pmatrix} & \begin{array}{l} \textit{sun} \\ \textit{rain} \end{array} \end{array}$$

- Represents a model where
  - A sunny day is followed by another sunny day with probability 90%
  - Rainy day is followed by rain with 50%
  - Etc.

Example: Weather Prediction

- Given a random day, what is its weather?
  - Weather on day 0 is known to be sunny
  - Option #1: "simulate" the weather over time:

$$\mathbf{x}^{(0)} = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} P = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix} = \begin{bmatrix} 0.9 & 0.1 \end{bmatrix}$$

$$\mathbf{x}^{(2)} = \mathbf{x}^{(1)} P = \begin{bmatrix} 0.9 & 0.1 \end{bmatrix} \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix} = \begin{bmatrix} 0.86 & 0.14 \end{bmatrix}$$

- But, this is tedious.

Example: Weather Prediction

- Alternative: note that in steady state, the next day's probabilities won't change from the current day

- If we can find a vector π such that $\pi = \pi P$, then π are the steady-state probabilities we're looking for

Example: Weather Prediction

$$
\begin{aligned}
P &= \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix} \\
\mathbf{q}P &= \mathbf{q} \qquad (\mathbf{q} \text{ is unchanged by } P.)
\end{aligned}
$$

- So $-0.1q_1 + 0.5q_2 = 0$, and since they are a probability vector we know that $q_1 + q_2 = 1$
- Solving this gives:

$$\begin{bmatrix} q_1 & q_2 \end{bmatrix} = \begin{bmatrix} 0.833 & 0.167 \end{bmatrix}$$

Extra slides for review

Where are we?

- Understand
  - How to build a network on one physical medium
  - How to connect networks
  - How to implement an adaptive, reliable byte stream
  - How to address network heterogeneity
  - How to address global scale
  - End-to-end issues and common protocols
  - Congestion control: TCP heuristics, switch/router approaches to fairness

Performance Metrics and Analysis

- Metrics
  - Traditional and extensions
  - Sources of delay
  - Optimizing communication systems
  - Measuring systems
- Basic queueing theory
  - Distributions and processes
  - Single, memoryless queues
- Analysis
  - Prefix problems (good for some Markov chains)
  - Example:
    - Throughput with TCP congestion control
    - Shared medium protocols

Performance Metrics

- Traditional metrics
  - End-to-end latency/RTT
    - Measures time delay
    - Across all layers of network
    - Often abbreviated to "latency" (even for RTT)
  - Bandwidth/throughput
    - Measures data sent per unit time
    - Across all layers of network
- Question: what's missing?

Performance Metrics

- CPU utilization not captured by latency/bandwidth
- Adopt additional metric from parallel computing
  - Distinguish between
    - Latency
      - Propagation delay between hosts
    - Overhead
      - Time spent by processor
  - RTT is twice the sum of
    - One overhead on sending processor
    - Propagation delay
    - One overhead on receiving processor
  - Send/receive overheads can differ

Performance Metrics

- Sources of delay
  - Latency: three main components
    - DMA from sending/to receiving host memory
    - Propagation delay in network
    - Queueing delay in routers
  - Overhead: also three main components
    - Data copy between buffers (e.g., into kernel memory)
    - Protocol (TCP, IP, etc.) processing
    - PIO to write description of frame
  - Note that overhead has fixed and per-byte costs

Performance Metrics

- Optimizing communication systems
  - Optimize the common case
    - Send/receive usually more important than connection setup/teardown
      - TCP header changes little between segments
      - Often only a few connections at end hosts
    - Minimize context switches
    - Minimize copying of data
- Question:
  - what’s hard about having 0 copies?

Performance Metrics

- Optimizing communication systems
  - General rule of thumb
    - Most (80-90%) messages are short
    - Most data (80-90%) travel in long messages
  - Focus on bottlenecks
    - Reduce overhead to improve short message performance
    - Reduce number of copies to improve long message performance
  - Thus, CPU speed is often more important than network speed

Performance Metrics

- Optimizing communication systems
  - Maximize network utilization
    - Use large packets when possible
    - Fill delay-bandwidth pipe
  - Avoid timeouts
    - Set timers conservatively
    - Use “smarter” receiver (e.g., with selective ACK’s)
  - Avoid congestion rather than recovering from it

Performance Metrics

- Measuring communication systems
  - Latency
    - Measure RTT for 0-byte (or 1-byte) messages
    - Also report variability
  - Bandwidth
    - Measure RTT for range of long messages
    - Divide by number of bytes sent
    - Report as graph or as value in asymptotic limit
  - Overhead
    - Time multiple N-byte message send operations
    - Be careful of flow control and aggregation

Modeling and Analysis

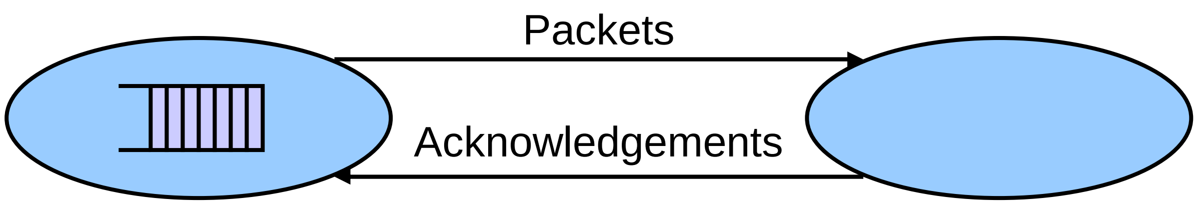
- Problem
  - The inputs to a system (i.e., number of packets and their arrival times) and the exact resource requirements of these packets cannot be predetermined in advance exactly
- But, we can probabilistically characterize these quantities
  - On average, 100 packets arrive per second
  - On average, packets are 500KB
- So, given a probabilistic characterization of these quantities
  - Can we draw some intelligent conclusions about the performance of the system

Delay

- Link delay consists of four components
  - Processing delay
    - From when the packet is correctly received to when it is put on the queue
  - Queueing delay
    - From when the packet is put on the queue to when it is ready to transmit
  - Transmission delay
    - From when the first bit is transmitted to when the last bit is transmitted
  - Propagation delay
    - From when the last bit is transmitted to when the last bit is received

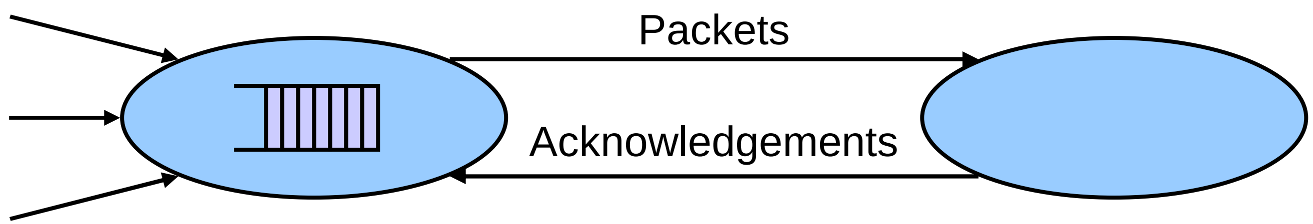
Delay Models

- Consider a data link using stop-and-wait ARQ
  - What is the throughput?
  - Given
    - MSS = packet payload size
    - C = raw link data rate
    - RTT = round trip time (for one bit)
    - p = probability a packet is successful



Delay Models

- Calculate the maximum throughput for stop-and-wait
  - Max throughput = packetlength/(RTT + (packetlength/C))
  - Could also multiply by (payload/packetlength) and p = probability of correct reception
- But what about the delay incurred?
  - There may be multiple bursty data sources



Basic Queueing Theory

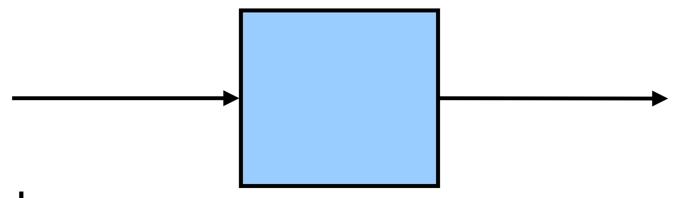
- Elementary notions
  - Things arrive at a queue according to some probability distribution
  - Things leave a queue according to a second probability distribution
  - Averaged over time
    - Things arriving and things leaving must be equal
    - Or the queue length will grow without bound
  - Convenient to express probability distributions as average rates

Little's Law

- Goal
  - Estimate relevant values
    - Average number of customers in the system
      - The number of customers either waiting in queue or receiving service
    - Average delay per customer
      - The time a customer spends waiting plus the service time
  - In terms of known values
    - Customer arrival rate
      - The number of customers entering the system per unit time
    - Customer service rate
      - The number of customers the system serves per unit time

Little's Law

- For any box with something steady flowing through it



$N = \lambda T$

Allows us to express the natural idea that crowded systems (large N) are associated with long customer delays

Little's Law

$$N = \lambda T$$

Mean amount in box → N; Mean throughput → λ; Mean time spent in box → T

- Example
  - Suppose you arrive at a busy restaurant in a major city
  - Some people are waiting in line, while other are already seated (i.e., being served)
  - You want to estimate how long you will have to wait to be seated if you join the end of the line
- Do you apply Little's Law? If so
  - What is the box?
  - What is N?
  - What is λ?
  - What is T?

Little’s Law

$$N = \lambda T$$

Mean amount in box

Mean throughput

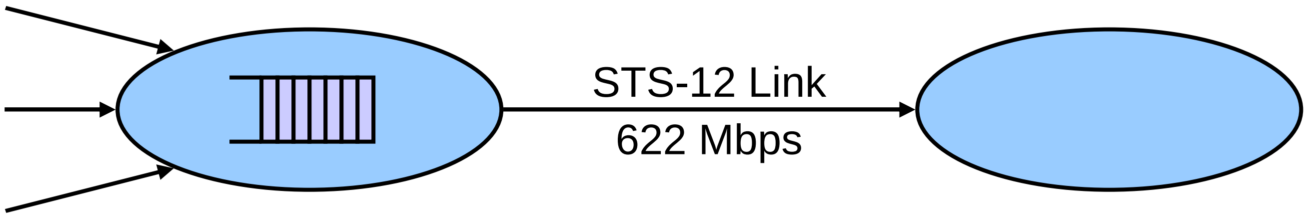
Mean time spent in box

- Box
  - Include the people seated (i.e., being served)
  - Do not include the people waiting in line
- Let N = the number of people seated (say 200)
- Let T = mean amount of time a person stays seated (say 90 min)
- Conclusion
  - Throughput = 200/90 = 2.22 persons per minute
- Wait time
  - If 100 people are waiting, you could estimate that you will need to wait 100/2.22 = 45 min

Little's Law

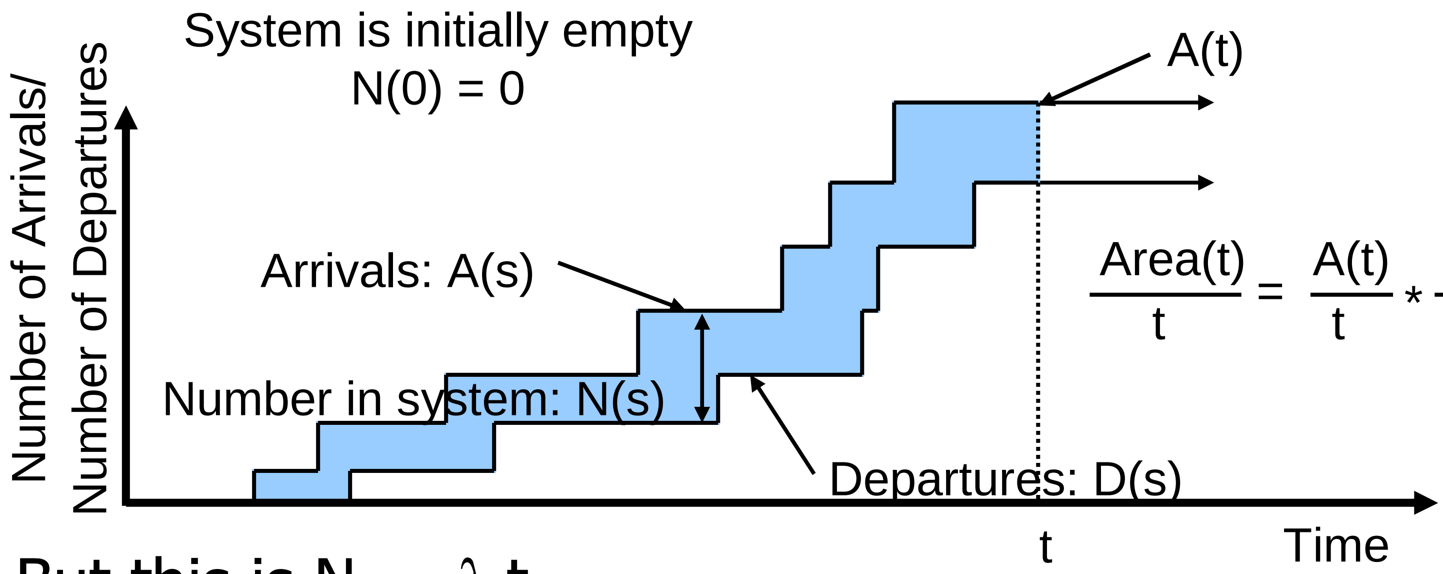
- Variables
  - N(t) = number of customers in the system at time t
  - A(t) = number of customers who arrived in the interval [0,t]
  - T_i = time spent in the system by the i^{th} customer
  - λ_t = average arrival rate over the interval [0,t]

Little's Law



- Suppose ATM streams are multiplexed at an output link with speed 622 Mbps
- Question
  - If 200 cells are queued on average, what is the average time in queue?
- Answer
  - $T = N/\lambda$
  - $T = 200 * 53 * 8 / 622M$
  - $T = 0.136$ ms

Proof of Little's Law



$$\frac{Area(t)}{t} = \frac{A(t)}{t} * \frac{Area(t)}{A(t)}$$

- But this is $N_t = \lambda_t t_t$
  - With time averaging over [0,t]
- Let t tend to infinity: $N = \lambda t$

Memoryless Distributions/ Poisson Arrivals

- Goal for easy analysis
  - Want processes (arrival, departure) to be independent of time
  - i.e., likelihood of arrival should depend neither on earlier nor on later arrivals
- In terms of probability distribution in time (defined for t > 0),

$$f(t) = \frac{f(t+\Delta t)}{\int_{\Delta t}^{\infty} f(t')\,dt'} \qquad \text{for all } \Delta t \geq 0$$

Memoryless Distributions/ Poisson Arrivals

solution is:

$$f(t) = \lambda e^{-\lambda t}$$

what is λ?

- it's the rate of events
- note that the average time until the next event is

$$\int_0^\infty f(t)\,t\,dt = \left(te^{-\lambda t}\right]_0^\infty + \int_0^\infty e^{-\lambda t}dt$$

$$= \left(-\frac{1}{\lambda}e^{-\lambda t}\right]_0^\infty$$

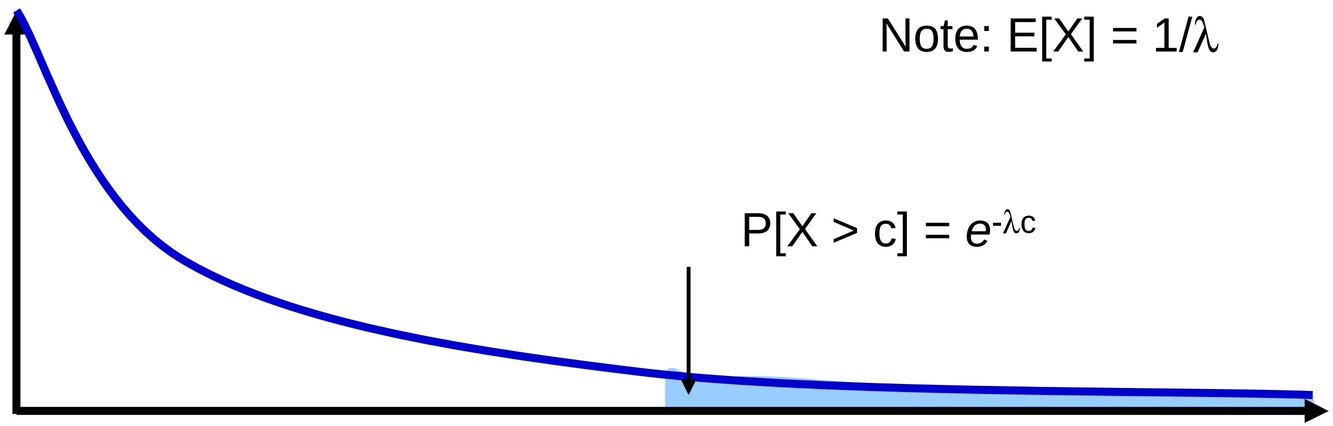
$$= \frac{1}{\lambda}$$

Plan

- Review exponential and Poisson probability distributions
- Discuss Poisson point processes and the M/M/1 queue model

Exponential Distribution

- A random variable X has an exponential distribution with parameter λ if it has a probability density function
  - $f(x) = \lambda\, e^{-\lambda x}$, for $x \geq 0$



Exponential Distribution

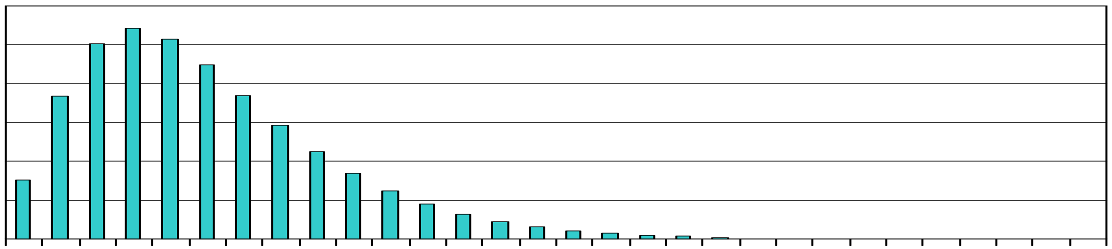
- Suppose a waiting time X is exponentially distributed with parameter λ = 2/sec
  - Mean wait time is ½ sec
- What is
  - P[X>2]?
  - P[X>6]?
  - P[X>6 | X>4]?

Exponential Distribution

- Remember: $\lambda = 2$
- P[X>2]
  - $= e^{-2\lambda} = 0.183$
- P[X>6]
  - $= e^{-6\lambda} = 6.14 \times 10^{-6}$
- P[X>6|X>4]
  - = P[X>6,X>4]/P[X>4]
  - = P[X>6]/P[X>4]
  - $= e^{-6\lambda}/e^{-4\lambda}$
  - $= e^{-2\lambda}$
  - = 0.183!
- Note: this demonstrates the memoryless property of exponential distributions

Poisson Distribution

- The random variable X has a Poisson distribution with mean λ, if for non-negative integers i:
  - $P[X = i] = (\lambda^i e^{-\lambda})/i!$
- Facts
  - $E[X] = \lambda$
  - If there are many independent events,
    - The k^{th} of which has probability p_k (which is small) and
    - λ = the sum of the p_k is moderate
    - Then the number of events that occur has approximately the Poisson distribution with mean λ

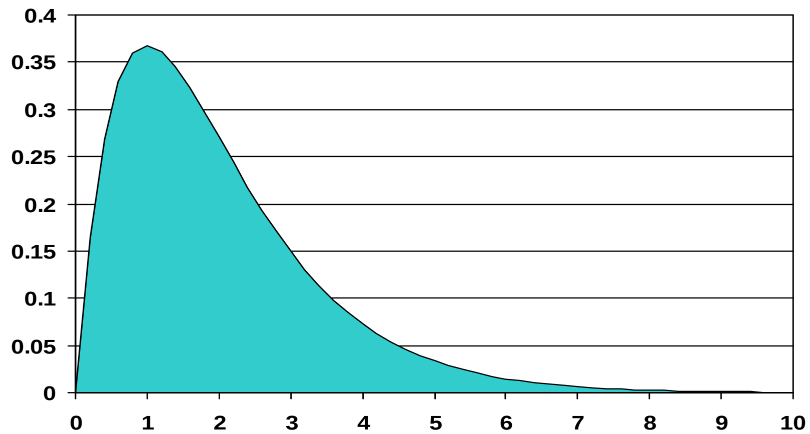


Poisson Distribution

- Example
  - Consider a CSMA/CD like scenario
  - There are 20 stations, each of which transmits in a slot with probability 0.03. What is the probability that exactly one transmits?

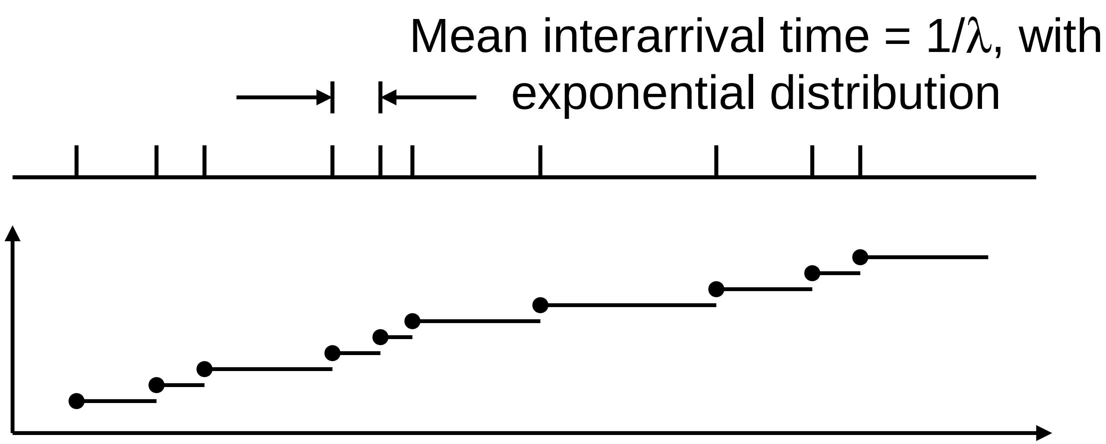
Poisson Distribution

- Exact answer
  - $20 * (0.03) * (1 – 0.03)^{19} = 0.3364$
- Poisson approximation
  - Use $P[X = i] = (\lambda^i e^{-\lambda})/i!$
  - With $i = 1$ and $\lambda = 20 * (0.03) = 0.6$
  - Approximate answer $= \lambda e^{\lambda} = 0.3393$



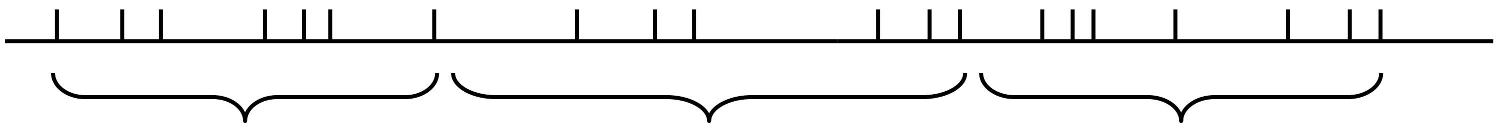
Poisson Point Process

- Definition
  - A Poisson point process with parameter λ
    - A point process with interpoint times that are independent and exponentially distributed with parameter λ.



Poisson Point Process

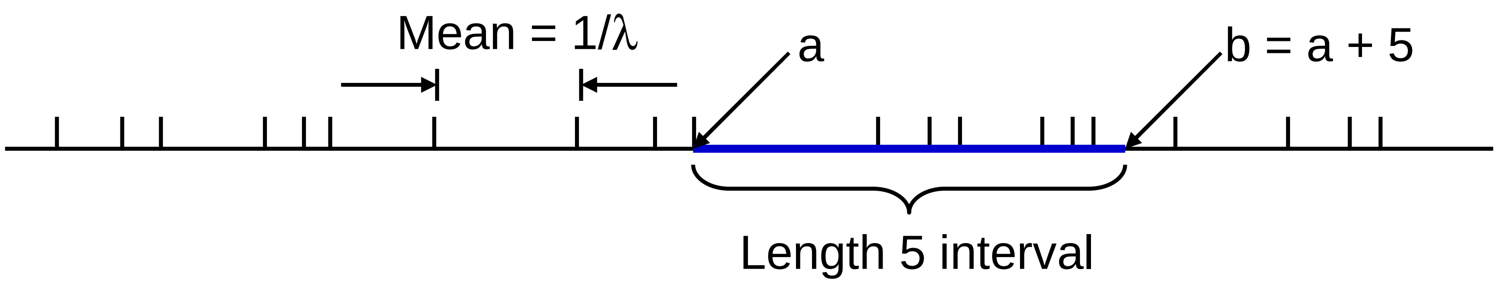
- Equivalently
  - The number of points in disjoint intervals are independent, and the number of points in an interval of length t has a Poisson distribution with mean λt



Shown are three disjoint intervals. For a Poisson point process, the number of points in each interval has a Poisson distribution.

Poisson Point Process

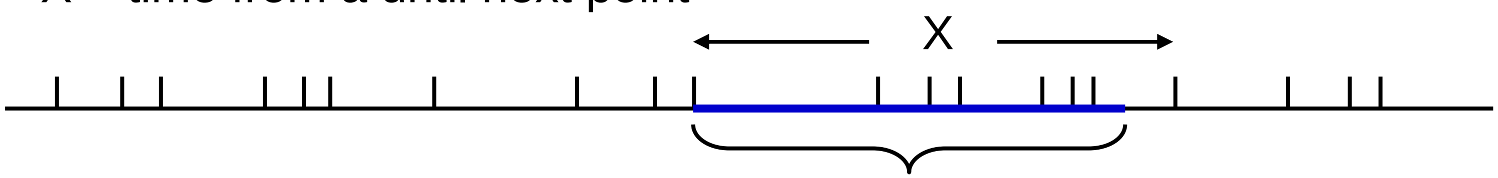
- Exercise
  - Given a Poisson point process with rate $\lambda = 0.4$, what is the probability of NO arrivals in an interval of length 5?



Try to answer two ways, using two equivalent descriptions of a Poisson process

Poisson Point Process

X = time from a until next point



N = number of points in interval

(Poisson with mean 5λ)

Solution 1: $P[X > 5] = e^{-5\lambda} = 0.1353$

Solution 2: $P[N = 0] = e^{-5\lambda} = 0.1353$

(remember: $P[N = i] = (5\lambda)^i * (e^{-5\lambda}) / i!$, for $i = 0$)

Simple Queueing Systems

- Classify by
  - "arrival pattern/service pattern/number of servers"
    - Interarrival time probability density function
    - The service time probability density function
    - The number of servers
    - The queueing system
    - The amount of buffer space in the queues
  - Assumptions
    - Infinite number of customers

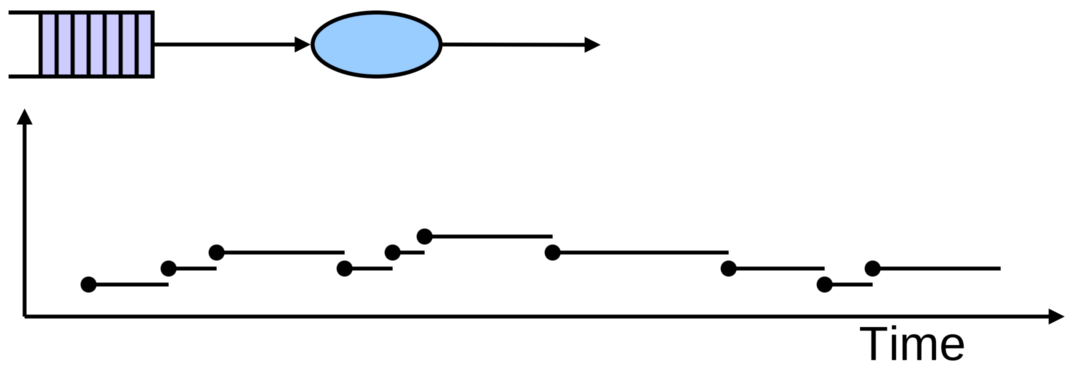
Simple Queueing Systems

- Terminology
  - M = Markov (exponential probability density)
  - D = deterministic (all have same value)
  - G = general (arbitrary probability density)
- Example
  - M/D/4
    - Markov arrival process
    - Deterministic service times
    - 4 servers

M/M/1 System

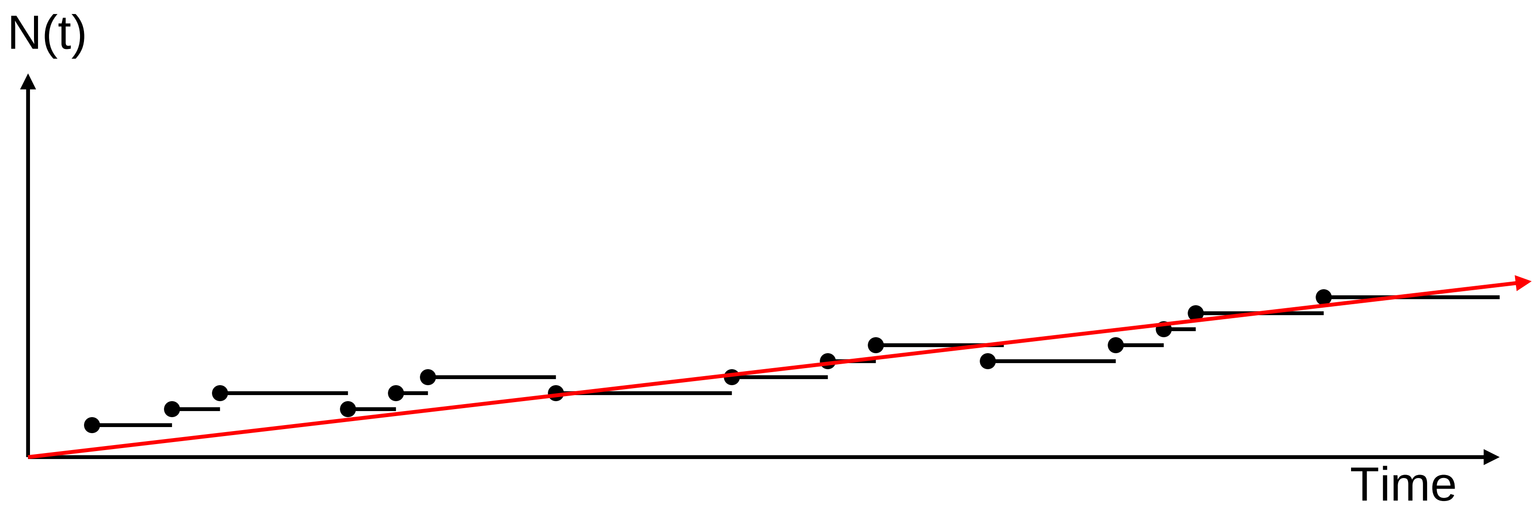
- Goal
  - Describe how the queue evolves over time as customers arrive and depart
- An M/M/1 system with arrival rate λ and departure rate μ has
  - Poisson arrival process, rate λ
  - Exponentially distributed service times, parameter μ
  - One server

N(t) = number in system (system = queue + server)



M/M/1 System

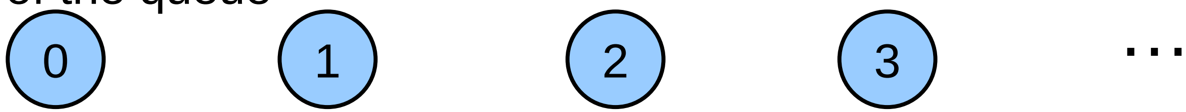
- If the arrival rate λ is greater then the departure rate μ
  - N(t) drifts up at rate λ - μ



M/M/1 System

- On the other hand,
  - if $\lambda < \mu$, expect an equilibrium distribution.
- The state of the queue is completely described by the number of customers in the queue
  - Due to the memoryless property of exponential distributions, N is described by a single state transition diagram
  - N is a Markov process, meaning past and future are independent given present

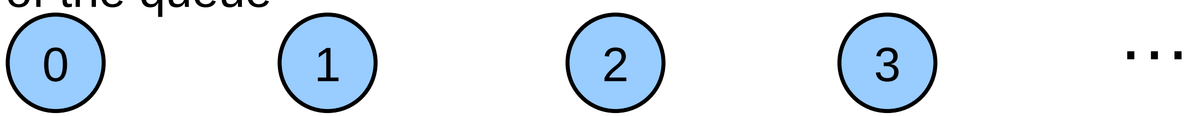
States of the queue



M/M/1 System

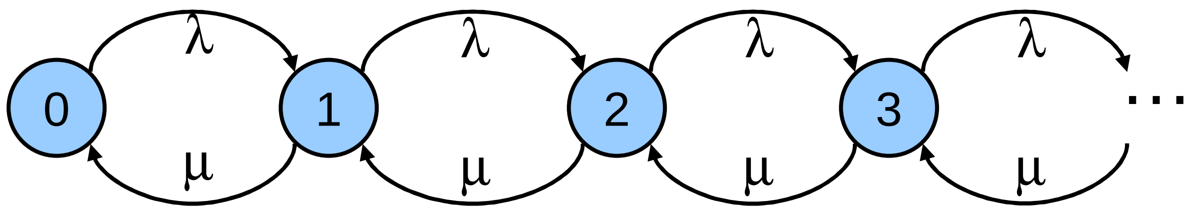
- N is a discrete random variable
  - p_k = probability that there are k customers in the queue
  - Equivalently,
    - p_k = probability that queue is in state k

States of the queue



M/M/1 System

- Goal
  - Find the steady state (long run) probabilities of the queue being in state i, i = 0, 1, 2, 3, …
- Transitions occur only when
  - A customer finishes service
  - A customer arrives
- Birth-death process
  - Transition from state i to state i+1 on arrival
  - Transition from state i to state i-1 on departure

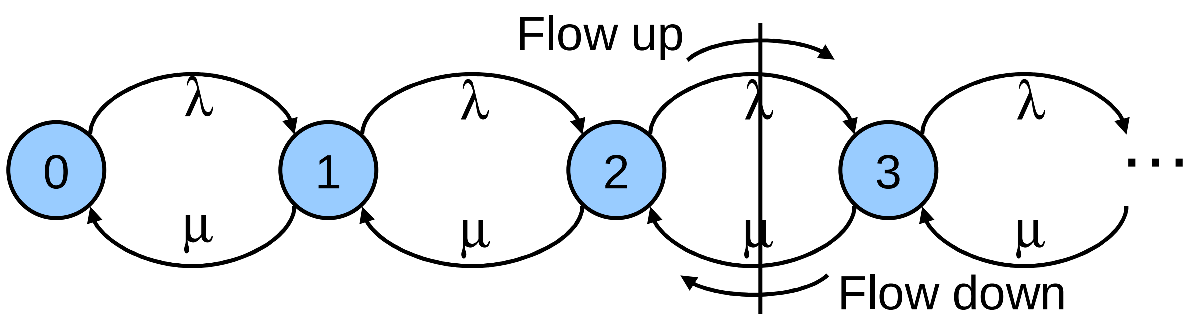


M/M/1: Transition rates

- If the queue is in state i with probability p_i
  - Then equivalently , the queue is in state i a fraction of p_i of the time
- The number of transitions/second out of state i onto state i+1 is given by
  - (fraction of time queue is in state i) * (arrival rate)
  - $p_i * \lambda$
- The number of transitions/second out of state i onto state i-1 is given by
  - (fraction of time queue is in state i) * (departure rate)
  - $p_i * \mu$

M/M/1: Steady State

- Claim
  - For the steady state to exist, # of transitions/sec from state i to state i+1 must equal # of transitions/sec from state i+1 to state i
- Result
  - Net flow across boundary between states must be zero
- Basic idea (not a real proof)
  - Otherwise, in the long run, the net flow of the system would always drift to the higher state with probability 1



M/M/1 System

- Given that we must balance flow across all boundaries,
  - $\lambda p_i = \mu p_{i+1}$ for all $i \geq 0$
- Balance Equations

$$
\begin{array}{llllll}
\lambda p_0 = \mu p_1 & \Rightarrow & p_1 = (\lambda/\mu)\, p_0 & & \\
\lambda p_1 = \mu p_2 & \Rightarrow & p_2 = (\lambda/\mu)\, p_1 & \Rightarrow & p_2 = (\lambda/\mu)^2\, p_0 \\
\lambda p_2 = \mu p_3 & \Rightarrow & p_3 = (\lambda/\mu)\, p_2 & \Rightarrow & p_3 = (\lambda/\mu)^3\, p_0 \\
\ldots & & \ldots & & \ldots \\
\lambda p_i = \mu p_{i+1} & \Rightarrow & p_{i+1} = (\lambda/\mu)\, p_i & \Rightarrow & p_{i+1} = (\lambda/\mu)^{i+1}\, p_0
\end{array}
$$

M/M/1 System

- Problem
  - To solve the balance equations, we need one more equation:
    - $\sum_{i=0}^{\infty} p_i = 1$
- Thus
  - $p_k = (\lambda/\mu)^k p_0$ (1)
  - $\sum_{i=0}^{\infty} p_i = 1$ (2)
- Plugging 1 into 2, we get
  - $\sum_{i=0}^{\infty} p_0 * (\lambda/\mu)^i = 1$
- Result (for $\lambda < \mu$)
  - $p_0 = 1 / (\sum (\lambda/\mu)^i) = \ldots = 1 - \lambda/\mu$
  - $p_k = (\lambda/\mu)^k * (1 - \lambda/\mu)$

M/M/1 System

- So What?
  - We now know the probability that there are 0, 1, 2, 3, … customers in the queue (p_i)
- Define N_{avg}
  - = average # of customers in queue
  - = expected value of the # of customers in the queue
- N_{avg}
  - $= \sum_{\text{all possible \# of cust}} i * P[i \text{ customers}]$
  - $= \sum_{i=0}^{\infty} i * p_i = \sum_{i=0}^{\infty} (1 - \lambda/\mu) * (\lambda/\mu)^i * i$
  - $= (\lambda/\mu)/(1 - \lambda/\mu)$

M/M/1 System

- Define Q_{avg}
  - = average # of customers in waiting area of the queue
- Q_{avg}
  - $= \sum_{\text{all possible \# of cust in waiting area}} i * P[\text{i customers in waiting area}]$
  - $= \sum_{i=0}^{\infty} i * P[\text{i+1 customers in queue}]$
  - $= \sum_{i=0}^{\infty} (1 - \lambda/\mu) * (\lambda/\mu)^{i+1} * i$
  - $= (\lambda/\mu)/(1 - \lambda/\mu) - \lambda/\mu$
  - $= N_{avg} - \lambda/\mu$

M/M/1 System - Utilization

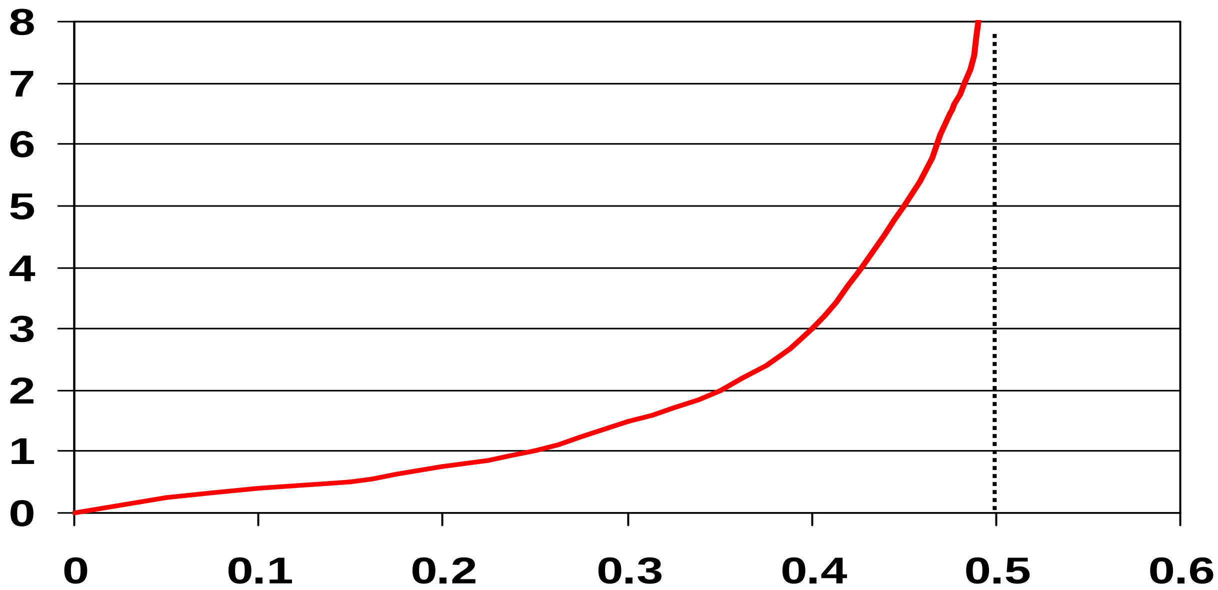
- Utilization
  - The fraction of time the server is busy
  - = P[server is busy]
  - = 1 – P[server is NOT busy]
  - = 1 – P[zero customers in queue]
  - = $1 - p_0$
  - = $1 - (1 - \lambda/\mu)$
  - = λ/μ
- Since utilization cannot be greater then 1,
  - Utilization = $\min(1.0, \lambda/\mu)$

M/M/1 System - Utilization

- Utilization example
  - Packets arrive for transmission at an average (Poisson) rate of 0.1 packets/sec
  - Each packet requires 2 seconds to transmit on average (exponentially distributed)
  - $N_{avg} = (\lambda/\mu)/(1 - \lambda/\mu) = 0.1*2 /(1 – 0.1*2) = 0.25$
  - $Q_{avg} = N_{avg} - \lambda/\mu = 0.25 – 0.2*2 = 0.005$
  - $\rho = \lambda/\mu = 0.2$

M/M/1 System - Utilization

- Intuitively, as the number of packets arriving per second (λ) increases, the number of packets in the queue should increase



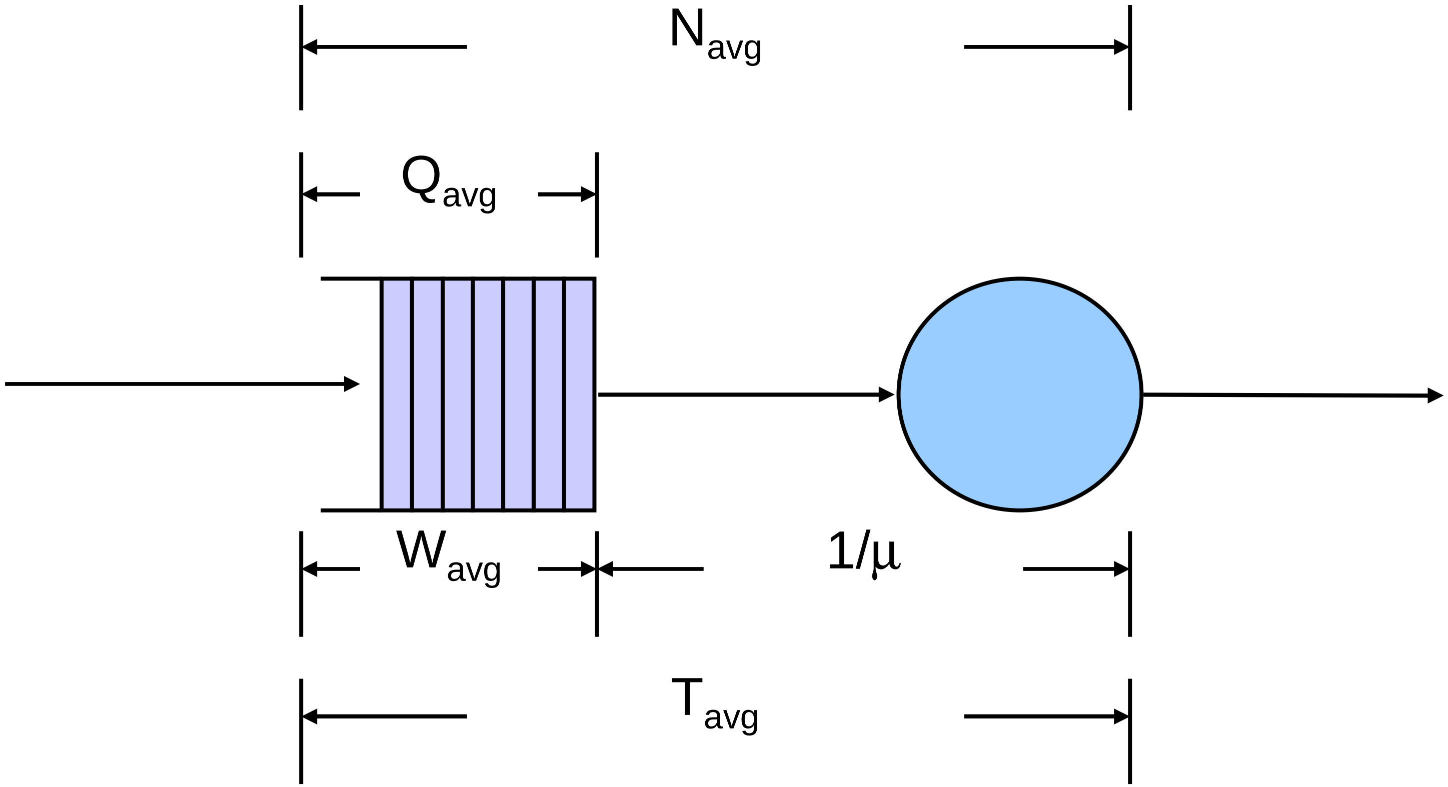
M/M/1 System - Utilization

- Normalized Traffic Parameter (ρ)
  - Note that N_{avg} and Q_{avg} only depend on the ratio λ/μ
  - Define ρ
    - = (avg arrival rate * avg service time)
    - = $\lambda * 1/\mu = \lambda/\mu$
  - Intuitively, if we scale both arrival rate and service time by a constant factor, N_{avg} and Q_{avg} should remain the same
  - Note
    - If $\lambda > \mu$ (i.e. $\lambda/\mu > 1$), then more packets are arriving per second than can be serviced
    - Thus, N_{avg} and Q_{avg} are unbounded when $\rho \geq 1$!

M/M/1 System – Time Delays

- Given $\{p_0, p_1, p_2, \ldots\}$, we can derive N_{avg} and Q_{avg}
- We may also want to know the following
  - T_{avg} = average time from when a packet arrives until it completes transmission
  - W_{avg} = average time from when a packet arrives until it starts transmission

M/M/1 System – Time Delays



M/M/1 System – Little's Law

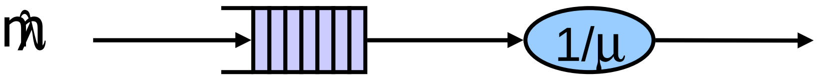
- Now we can use Little's Law to relate N_{avg} and Q_{avg} to T_{avg} and W_{avg}
  - $N_{avg} = \lambda T_{avg}$ $\Rightarrow T_{avg} = N_{avg}/\lambda$
  - $Q_{avg} = \lambda W_{avg}$ $\Rightarrow W_{avg} = Q_{avg}/\lambda$
  - Also note: $W_{avg} + 1/\mu = T_{avg}$

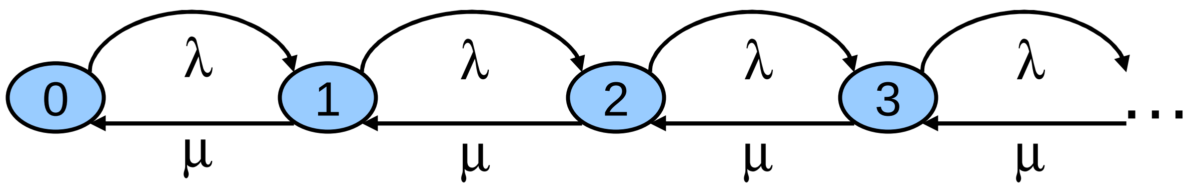
M/M/1 System

- Packets arrive with the following parameters
  - λ = 2 packets per second
  - $1/\mu$ = ¼ sec per packets
  - ρ = 0.5
- Utilization = $\rho = \lambda/\mu$ = 2/4 = 0.5
- $N_{avg} = \rho/(1 - \rho)$ = 0.5/1-0.5 = 1 packet
  - $\Rightarrow T_{avg} = N_{avg}/\lambda$ = ½ = 0.5 sec
- $Q_{avg} = N_{avg} - \rho = 1 – 0.5 = 0.5$
  - $\Rightarrow W_{avg} = Q_{avg}/\lambda$ = 0.5/2 = 0.25 sec

M/M/1 System - Summary

1. Draw state diagram





2. Write down balance equations

   flow "up" = flow "down"

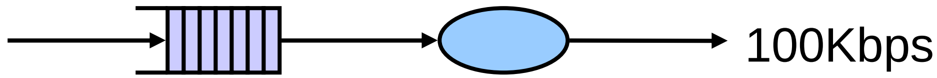
3. Solve balance equations using

   $\sum_{i=0}^{\infty} p_i = 1$ for $\{p_0, p_1, p_2, \ldots\}$

4. Compute N_{avg} and Q_{avg} from $\{p_i\}$

5. Compute T_{avg} and W_{avg} using Little's Theorem

M/M/1 System - Example



- Packets arrive at an output link according to a Poisson process
  - The mean total data rate is 80Kbps (including headers)
  - The mean packet length is 1500
  - The link speed is 100Kbps
- Questions
  - What assumptions can we make to fit this situation to the M/M/1 model?
  - Under these assumptions, what is the mean time needed for queueing and transmission of a packet?

M/M/1 System - Example

- Answer Part 1:
  - "Customers"
    - Packets
  - "Server"
    - The transmitter
  - Service times
    - The transmission times
  - Packets have variable lengths, with a exponential distribution
  - Packet lengths are independent of each other and independent of arrival time

M/M/1 System - Example

- Remember
  - The mean total data rate is 80Kbps
  - The mean packet length is 1500
  - The link speed is 100Kbps
- Answer Part 2: Find λ, μ and T
  - Need to convert from bit rates to packet rates
    - λ = 80Kbps/12Kb = 6.66 packets/sec
    - μ = 100 Kbps/12Kb = 8.33 packets/sec
  - So, T = mean time for queueing and transmission
    - $T = 1/(\mu - \lambda) = 1/1.67 = 0.6$ sec

M/M/1 System - Example

- Also
  - The mean transmission time is
    - $1/\mu = 0.12$ sec,
  - So the mean time spent in queue is
    - $W = T - 1/\mu = 0.6 – 0.12 = 0.48$sec
  - The mean number of packets is
    - $N = \rho/(1 - \rho) = 0.8/(1 – 0.8) = 4$ packets

M/M/1 System in Practice

- The assumptions we made are often not realistic
- We still get the correct qualitative behavior
- Simple formulas for predictive delay are useful for provisioning resources in a network and setting controls
- Real traffic seems to have bursty behavior on multiple time scales
  – This is not true for Poisson processes

Analysis: Tools and Examples

- Cycle analysis for discrete Markov processes
  - Start with a discrete Markov process
    - Transitions happen periodically (every Δt)
    - Probabilities independent of past/future behavior
  - Form all possible cyclic sequences (cycles)
    - Pick a "start" state
    - List all cycles from that state
    - Calculate probability per cycle
    - Calculate average cycle length
  - Can calculate expected values of cycle-dependent properties with average length and cycle probabilities

Network Example

- Slotted CSMA/CD access
- 10 transmitters
- Each with 1/20 probability to transmit in an idle slot
- A transmission
  - Lasts 5 slots,
  - Transmits 5 data units, and
  - Suppresses other transmissions.
- What is average throughput per slot?

Network Example

- What is average throughput per slot?
  - Find the number of successful transmissions
- Two types of slots
  - Non-suppressed
    - Chance of success in non-suppressed slot is:

      $10 \bullet (p) \bullet (1 - p)^9 = 10 \bullet (1/20) \bullet (19/20)^9 = 0.315$
  - Suppressed
    - Chance of success in suppressed slot is:

      1

Network Example

- Use cycle analysis

cycle	probability
I	0.685
1234I	0.315

- Average cycle length
  = 1•0.685+5•0.315 = 2.260 slots
- Average throughput
  = 5•0.315
  = 1.575 data units/cycle
- Throughput per slot
  = 1.575/2.260
  = 0.697 data units/slot

(compare with 0.315 data units/slot using 1-slot packets)

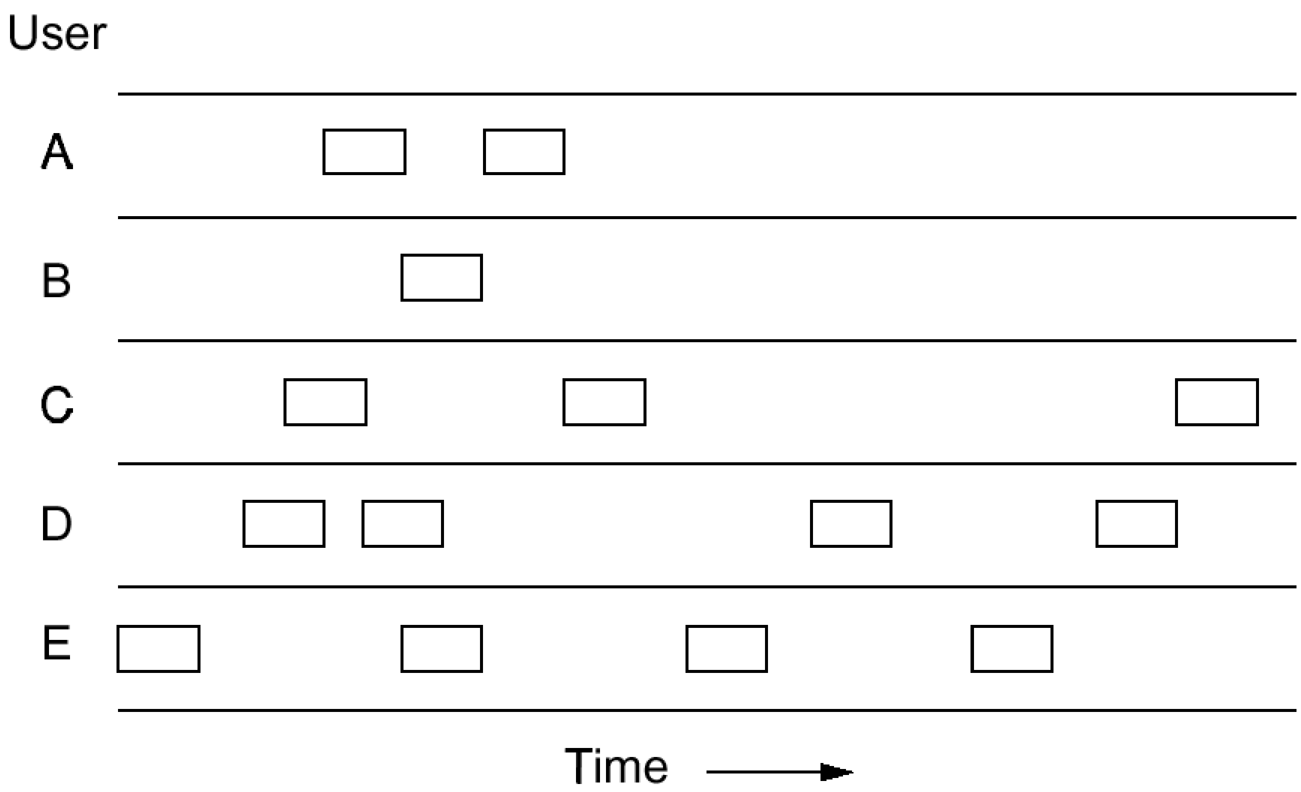
Analysis of Shared Medium Protocols

- ALOHA
  - Packet radio system on Hawaiian Islands
  - Two forms
    - Pure
      - No global synchronization
      - Low utilization
    - Slotted
      - Global synchronization (to define time slots)
      - Larger (but still fairly low) utilization

Pure ALOHA

- User model
  - Each transmitter hooked to one terminal
  - One person at each terminal
  - Person types a line, presses return
  - Transmitter sends line
  - Checks for success (no interference)
  - If collision occurred, wait random time and resend

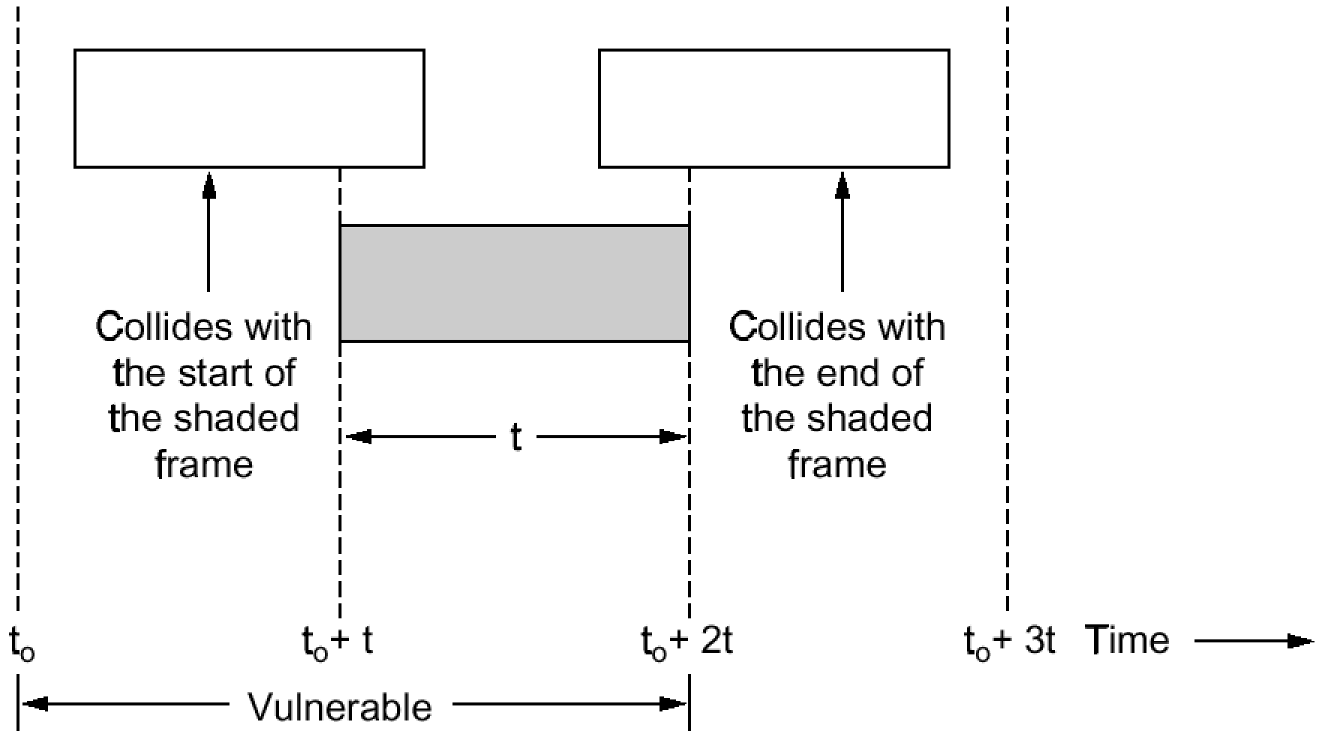
Pure ALOHA



Pure ALOHA

- Collisions
  - A frame not will suffer a collision if no other frames are sent within one frame time of its start
  - Let t = time to send a frame
  - If any other user has generated a frame between time t_0 and time $t_0 + t$, the end of that frame will collide with the beginning of our frame
  - Similarly, any other frame started between time $t_0 + t$ and time $t_0 + 2t$ will collide with the end of our frame

Pure ALOHA



Collides with the start of the shaded frame
t
Collides with the end of the shaded frame
t_o
t_o+ t
t_o+ 2t
t_o+ 3t
Time
Vulnerable

Pure ALOHA

- Also assume fixed packet sizes (maximizes throughput)
- Arrival and success rates
  - Frames generated at rate S
  - In steady state, must leave at S as well
  - Some frames retransmitted
  - Assume also Poisson with rate G, G > S

Pure ALOHA

- Question:
  - How does G (retransmission rate) relate to S (frame rate)?
- $S = G\ P_0$
  - P_0 is the probability of successful transmission

Pure ALOHA

- Simplifying assumptions
  - Poisson arrival process
  - Infinite pool of users (want to ignore blocked user effects)
- Frame Arrival
  - The probability that k frames will be generated during a given frame time is governed by a Poisson distribution

$$\Pr[k] = \frac{G^k e^{-G}}{k!}$$

Pure ALOHA

- Empty slot
  - The probability of no frames being transmitted is e-G
- How many frames in our transmission period?
  - In an interval two frames long, the mean number of frames generated is 2G
- Collision?
  - The probability of no other traffic being generated during the entire vulnerable period is
  - $P0 = e^{-2G}$
- Remember
  - $S = GP_0$
  - $S = Ge^{-2G}$

Pure ALOHA

- What is the relationship between offered traffic and throughput?
  - Maximum throughput occurs
    - G = 0.5
    - S = 1/2*e*
- Utilization
  - Maximum of 0.184!

Slotted ALOHA

- Hosts wait for next slot to transmit
- Vulnerable period is now cut in half
- How many frames in our transmission period?
  - In an interval one frame long, the mean number of frames generated is G
- Collision?
  - The probability of no other traffic being generated during the entire vulnerable period is
  - $P0 = e^{-G}$
  - $S = Ge^{-G}$

Slotted ALOHA

- What is the relationship between offered traffic and throughput?
  - Maximum throughput occurs
    - G = 1
    - S = 1/*e*
- Utilization
  - Maximum of 0.368!
  - 37% empty slots
  - 37% successes
  - 26% collisions

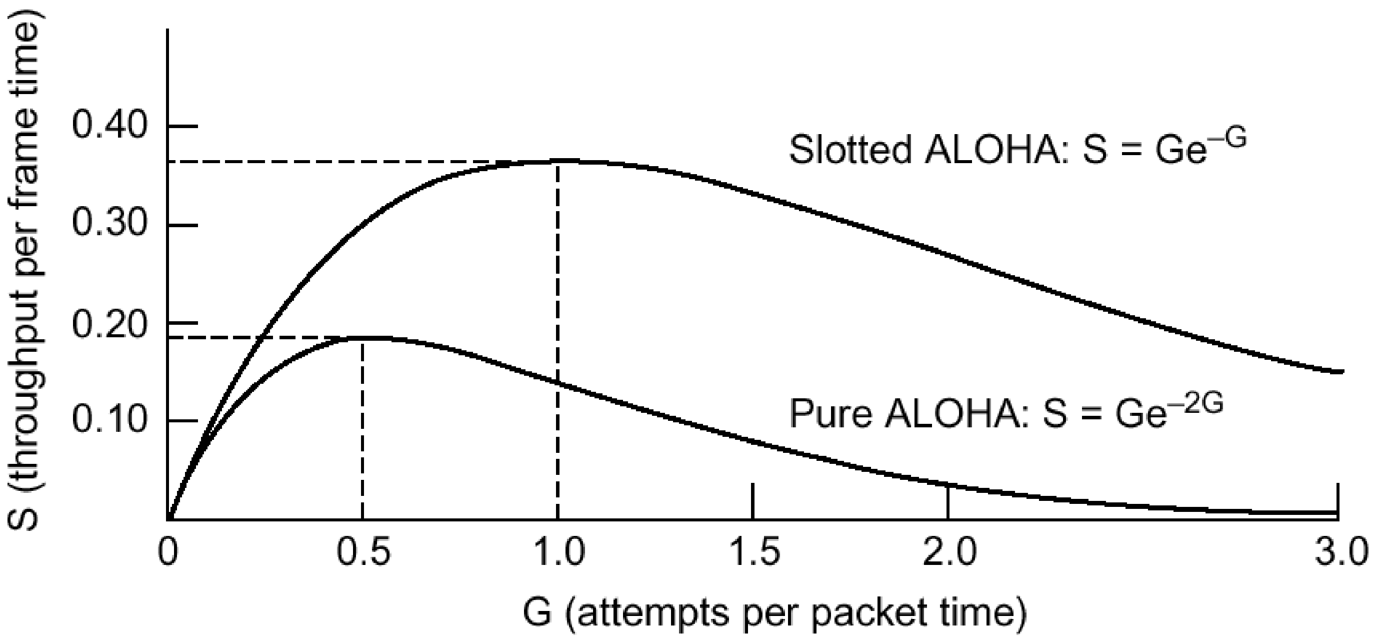
Slotted ALOHA

- Higher values of G
  - Reduces the number of empty slots
  - Increases the number of collisions exponentially
- Consider the transmission of a test frame
  - P[collision] = $1 - e^{-G}$
  - P[transmit in k attempts] = $e^{-G} (1 - e^{-G})^{k-1}$
    - (k – 1 collisions followed by one success)
  - E[# of transmissions] $= \Sigma_{k-1}^{\infty} kP_k$

    $= \Sigma_{k-1}^{\infty} ke^{-G} (1 - e^{-G})^{k-1}$

    $= e^{G}$
- Small increases in channel load can drastically reduce its performance

Aloha Analysis



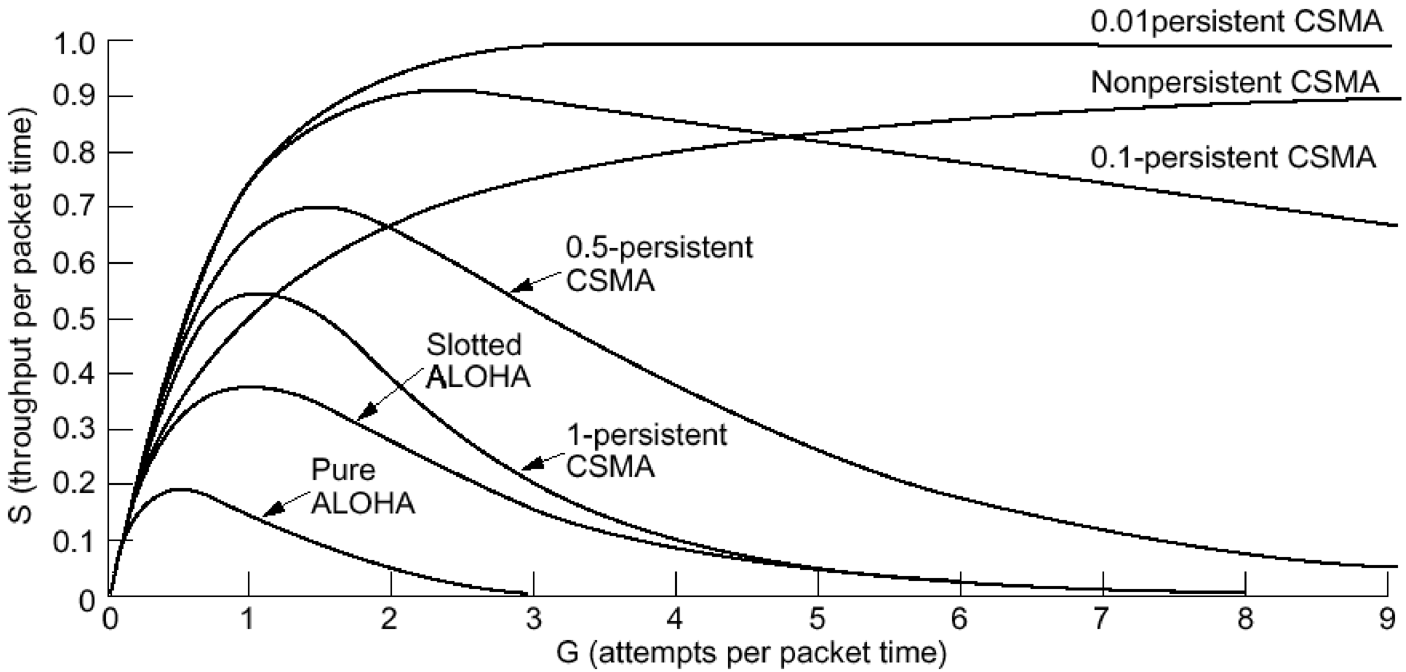
ALOHA Analysis

- Tradeoff
  - Pure ALOHA provides smaller delays
  - Slotted ALOHA provides higher throughput

Carrier Sense Protocols

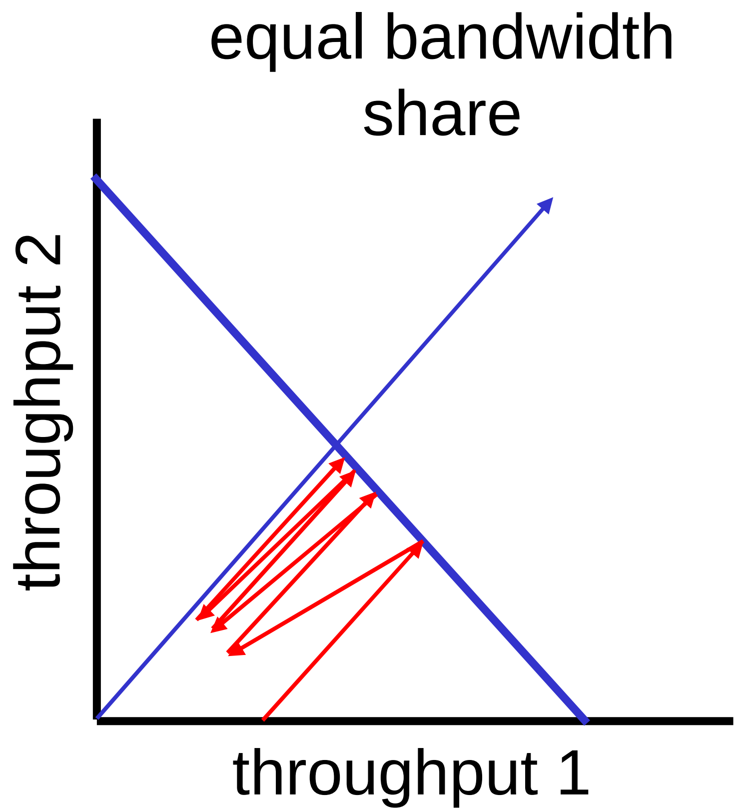
- Unlike ALOHA, listen for other transmissions before sending
- Two classes divided by action taken when another host is transmitting
  - Persistent:
    - listen until transmission completes
  - Non-persistent:
    - back off randomly, then try again
- Persistent protocols vary by chance of transmission
  - p-persistent gives p chance of transmission per idle slot
  - Ethernet is special case: 1-persistent, always transmits when idle slot perceived

CSMA Analysis



TCP Throughput on a Congested Link

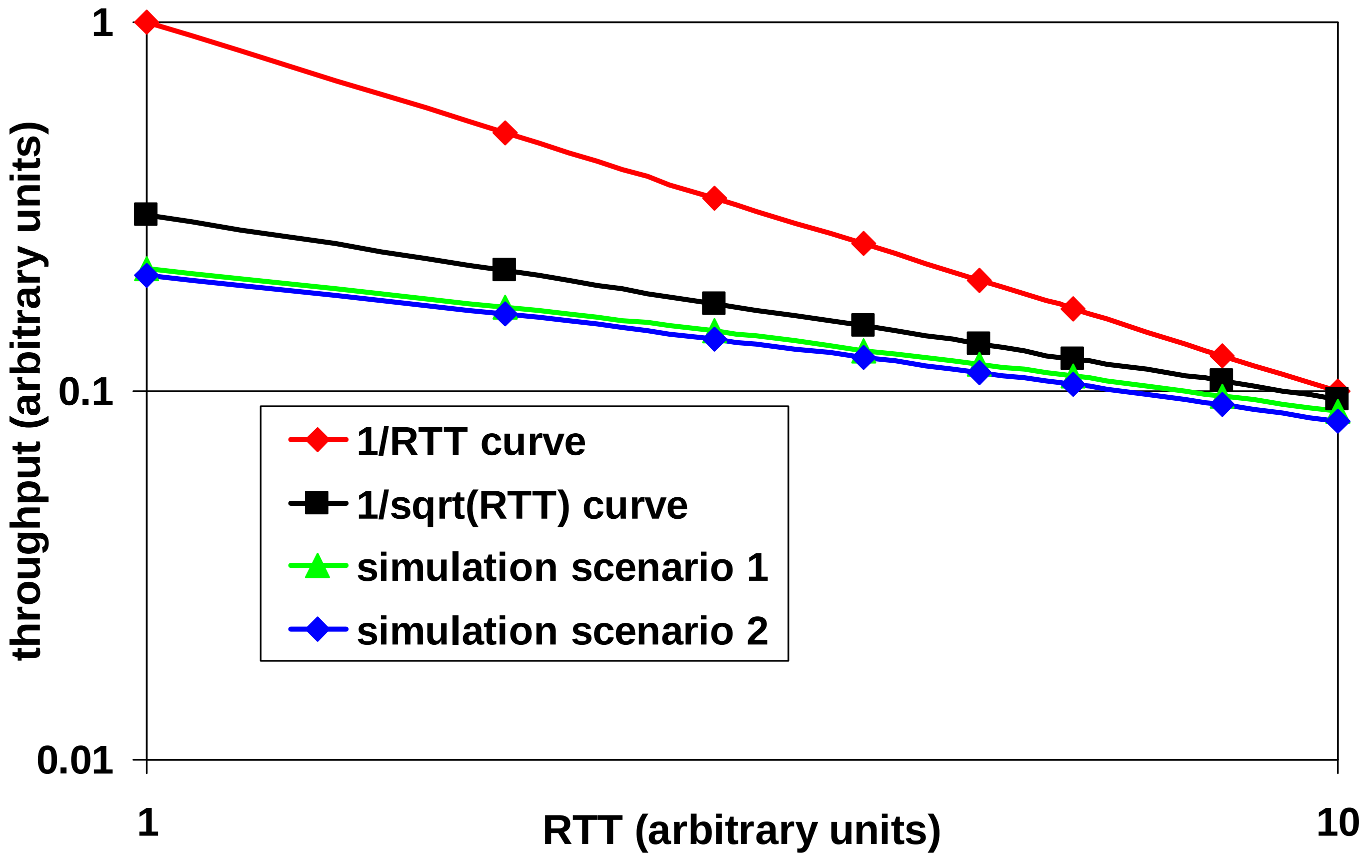
- What assumption was made for fairness?
  - At equilibrium, AIMD growth and backoff go in opposite directions
  - Backoff always goes toward origin
- What about growth (i.e., does it always have slope 1)?



Expected TCP Throughput

- NO!
  - Additive increase adds fixed amount per RTT
  - Throughput growth is proportional to 1/RTT
  - For two-flow case, slope is RTT_1/RTT_2
- Analysis with many flows
  - Bottleneck capacity C
  - Rates grow to bottleneck, then all back off at once
  - Total rate of throughput growth is fixed, so time Δt between backoffs is also fixed
  - Growth for each flow is $\Delta t/RTT$, and throughput is proportional to this growth

Throughput Dependence on RTT



Throughput Dependence on RTT

- What's going on?
  - Assumed all flows back off under contention
    - (arguably) more likely that only one flow backs off
  - Probability of congestion packet loss is proportional to throughput
  - Intuition
    - Low-RTT flow is more likely to back off
    - Reduces throughput advantage of low-RTT flows

"Analysis"

- Consider a flow F among many, varied flows
  - Backoffs happen very frequently
  - Probability to back off proportional to rate
  - Could happen any time
  - Approximate by Poisson process
- Let flow F have expected throughput C
  - Exp. time between backoffs proportional to 1/C
  - Between backoffs, throughput changes from 2/3 C to 4/3 C (average is C)
  - Rate of change proportional to C^2
  - Rate of change also proportional to 1/RTT
  - Thus C proportional to 1/sqrt(RTT)

Lessons from this Example

- Analysis
  - Only as good as your understanding
  - Easy to shortcut steps when you know the answer (non-rigorous math is not uncommon)
- Simulation
  - No better than analysis with regard to understanding
    - e.g., a simulator that backs off all flows achieves throughput proportional to 1/RTT
- Experiments are necessary!
  (but can be hard to set up)