Overview of Topics

- Intro to queueing theory
- Discrete-time queues: the Geom/Geom/1 queue
- Continuous-time queues: the M/M/1 queue
- Little’s Law
- Poisson Arrivals See Time Averages (PASTA)
- Corollary: Determinism Minimizes Delay
- References for queueing theory:
  - Data Networks. D. Bertsekas and R. Gallager
  - Stochastic Modelling and the Theory of Queues. R.W. Wolff
- Intro to Stability
Intro to Queueing Theory

• Notation: $A/S/s/k$
  - $A$ stands for the arrival process; e.g. Poisson, Geometric, Deterministic
  - $S$ stands for the service distribution; e.g. Exponential, Geometric, Deterministic
  - $s$ stands for the number of servers
  - $k$ stands for the amount of buffers (if $k$ is absent, then it is understood $k = \infty$)
  - also, there’s the service discipline: FCFS, LCFS, processor sharing, SRPT, etc

• Mainly concerned with the Geom/Geom/1, the M/M/1 and the M/D/1 queues
Discrete-time queues

- Time is slotted
  - packets arrive at the beginning of time slots and depart at the end of time slots
  - there is at most one arrival or departure per time slot

- The Geom/Geom/1 queue: Distributions
  - arrivals are Bernoulli: \( P(\text{arrival at time } t) = p \)
    therefore, interarrival times are geometric with mean \( \frac{1}{p} : P(A = k) = p(1 - p)^{k-1} \)
  - services are Bernoulli: that is, if queue is non-empty at time \( t \)
    the \( P(\text{HoL packet is served at end of time } t) = q \)
    therefore, service times are geometric with mean \( \frac{1}{q} : P(S = k) = q(1 - q)^{k-1} \)
The Geom/Geom/1 queue

- Let queue-size at time $t$, after arrivals and before departures, be $Q(t)$
  - then $Q(t)$ is a discrete-time Markov chain
  - with state space $\{0, 1, 2, \ldots\}$, and transition diagram

- this is called a “birth-death chain”
  - with birth probability $u = p(1-q)$ and death probability $d = q(1-p)$
  - the transition matrix is:

$$
P = \begin{bmatrix}
(1-p) & p & 0 & 0 & \cdots \\
\vdots & d & r & u & 0 & \cdots \\
0 & d & r & u & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & d & r & u & 0 & \cdots \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \ddots
\end{bmatrix}
$$
The Geom/Geom/1 queue

- The equilibrium distribution $\pi$ is obtained by solving $\pi = \pi P$
  - $\pi_0 = (1 - p)\pi_0 + d\pi_1 \Rightarrow \pi_0 = \frac{d}{p}\pi_1$
  - $\pi_1 = p\pi_0 + r\pi_1 + d\pi_2 \Leftrightarrow (d + u)\pi_1 = u\pi_1 + d\pi_2 \Leftrightarrow u\pi_1 = d\pi_2$
  - for $i \geq 2$, $\pi_i = u\pi_{i-1} + r\pi_i + d\pi_{i+1} \Leftrightarrow u\pi_i = d\pi_{i+1}$

- Therefore,
  - $\pi_i = \left(\frac{u}{d}\right)^{i-1}\pi_1, \ i \geq 1, \ and \ \pi_0 = \frac{q(1-p)}{p}\pi_1$
  - since $\sum_{i \geq 0} \pi_i = 1$, we get $\pi_1 = \frac{p(q-p)}{q^2(1-p)}$
  - note that $u < d$ is necessary, else $\sum_{i \geq 0} \pi_i = \infty$
  - thus, for “stability” we need birth probability ($u$) < death probability ($d$)

- And
  - letting $\rho = \frac{u}{d}$, we get $\pi_1 = \frac{p(1-p)}{q}$
  - and $P(Q = i) = \pi_1\rho^{i-1}, \ i \geq 1, \ P(Q = 0) = \pi_1\frac{q(1-p)}{p}$
  - $\rho$ is called the “load factor” and we have seen that $\rho < 1$ for stability
  - this is equivalent to $p < q$ (arrival rate < service rate)
Some queueing quantities

- let $Q$ be the steady-state number of packets in the queue, including the one in service
- then, $E(Q) = \sum_{i \geq 1} i \pi_i = \frac{\pi_1}{(1-\rho)^2} = \frac{p}{q(1-\rho)}$

- let $N_Q$ be the steady-state number of packets in the queue, excluding the one in service
- then, $N_Q = Q - 1_{\{Q > 0\}} \Rightarrow E(N_Q) = E(Q) - P(Q > 0) = \frac{p}{q(1-\rho)} - \frac{p}{q} = \frac{pq}{q(1-\rho)}$
The M/M/1 Queue

• Specification
  - arrivals occur according to a rate \( \lambda \) Poisson process therefore, interarrival times are \( \exp(\lambda) \): \( P(A > t) = e^{-\lambda t} \)
  - service times are IID rate \( \mu \) exponentials, and independent of arrivals i.e. \( P(S > t) = e^{-\mu t} \)

• Let \( Q(t) \) be the queue-size process at time \( t \)
  - then \( Q(t) \) is a CTMC with state space \( \{0, 1, 2, \ldots\} \)
  - so we need to find out the transition rates \( q_{ij} = \gamma_i p_{ij} \)

• Determining \( \gamma_i \): We know that \( \gamma_i = \frac{1}{E(T_i)} \), where \( T_i \) is the average time in state \( i \). There are two cases to consider

1. for \( i \geq 1 \), \( Q(t) \) leaves \( i \) if there is an arrival or if there is a departure

\[
E(T_i) = E(\min\{\text{arrival time, service time}\}) = \int_0^\infty P(\min\{A, S\} > t) \, dt \\
= \int_0^\infty P(A > t)P(S > t) \, dt = \int_0^\infty e^{-(\lambda+\mu)t} \, dt = \frac{1}{\lambda+\mu} \\
\Rightarrow \gamma_i = \frac{1}{E(T_i)} = \lambda + \mu
\]
2. for $i = 0$, $Q(t)$ leaves $i$ if there is an arrival

$$E(T_i) = E(\text{arrival time}) = \int_0^\infty P(A > t) \, dt = \frac{1}{\lambda}$$

$$\Rightarrow \gamma_i = \frac{1}{E(T_i)} = \lambda$$

- Determining $p_{ij}$ (recall that $p_{ii} = 0$ for CTMCs)

1. for $i \geq 1$, $p_{i,i+1} = P(\text{arrival happens before service}) = P(A < S)$

   thus, $p_{i,i+1} = \int_0^\infty P(A < t, S \in (t, t+dt)) \, dt = \int_0^\infty (1 - e^{-\lambda t}) \mu e^{-\mu t} \, dt = \frac{\lambda}{\lambda + \mu}$

   since $p_{i,i} = 0$, $p_{i,i-1} = 1 - p_{i,i+1} = \frac{\mu}{\lambda + \mu}$

2. for $i = 0$, $p_{i,i+1} = 1$ and $p_{i,i-1} = 0$

- Finally, $q_{i,i+1} = \lambda$ and $q_{i,i-1} = \mu$ (for $i \geq 1$) and $q_{i,i+1} = \lambda$ (for $i = 0$)

- To get the equilibrium distribution, $\pi$

  - we need to solve the global balance equations: $\pi_j \sum_{i=0}^\infty q_{ji} = \sum_{i=0}^\infty \pi_i q_{ij}$

  $$\Rightarrow \text{for every } j: \quad \pi_j \lambda = \pi_{j+1} \mu \quad \Rightarrow \quad \pi_j = \left(\frac{\lambda}{\mu}\right)^j \pi_0$$
The M/M/1 Queue: Equilibrium or steady-state

- If

\[ \lambda > \mu: \quad \pi_j \to \infty \text{ and there is no steady-state} \]
- this is to be expected, since more work is coming in than can be processed
- this is the unstable case

\[ \lambda = \mu: \quad \pi_j = \pi_0 \text{ for every } j \]
- since we require both \( 0 \leq \pi_j \leq 1 \) and \( \sum_{j=0}^{\infty} \pi_j = 1 \), there is again no equilibrium
- this is the critically unstable case

\[ \lambda < \mu: \quad \pi_j = \left( \frac{\lambda}{\mu} \right)^j \pi_0 \text{ is a geometrically decreasing sequence and } \sum_{j=0}^{\infty} \pi_j = 1 \text{ implies} \]
- \( \pi_0 = (1 - \frac{\lambda}{\mu}); \quad \pi_j = \left( 1 - \frac{\lambda}{\mu} \right) \left( \frac{\lambda}{\mu} \right)^j \)
- this is the stable case

- Quantities

- \( \rho = \frac{\lambda}{\mu} \) is called the traffic intensity; if \( \rho < 1 \), then \( \pi_j = (1 - \rho)\rho^j \)
- for \( \rho < 1 \), \( E(Q) = \sum_j j\pi_j = \frac{\rho}{1-\rho} \)
- \( E(N_Q) = E(Q - 1_{\{Q>0\}}) = E(Q) - P(Q > 0) = \frac{\rho}{1-\rho} - \rho = \frac{\rho^2}{1-\rho} \)
Little’s Law

- This relates average queue-sizes and average waiting times
  - holds quite generally - no distributional assumptions on arrivals and services

- Suppose that a stable queue
  - is empty at time 0
  - $A(t)$ be the number of arrivals in $[0, t]$, let $\lambda = E(A(1))$ be the arrival rate
  - $D_i$ be the “delay” of the $i^{th}$ packet (delay = waiting time + service time)

- from figure, we see that

$$\int_{t=0}^{T} Q(t) \, dt \quad \frac{T}{T} = \frac{\sum_{k=1}^{A(T)} D_k}{T} = \frac{A(T)}{T} \sum_{k=1}^{A(T)} D_k.$$

- letting $T \to \infty$ we get: $E(Q) = \lambda E(D)$ this is Little’s Law
Some consequences

- For the Geom/Geom/1 queue
  - we have seen that \( E(Q) = \frac{p}{q(1-\rho)} \)
  - and arrival rate \( \lambda = p \)
  - therefore, \( E(D) = E(Q)/\lambda = \frac{1}{q(1-\rho)} \)
  - since waiting time = delay - service time,
    - \( E(W) = E(D) - E(S) = \frac{1}{q(1-\rho)} - \frac{1}{q} = \frac{\rho}{q(1-\rho)} \)
    - by Little’s Law, \( E(N_Q) = \lambda E(W) = p\frac{\rho}{q(1-\rho)} \)

- Similarly for the M/M/1 queue
  - we have seen that \( E(Q) = \frac{\rho}{1-\rho} \)
  - therefore, \( E(D) = E(Q)/\lambda = \frac{1}{\mu(1-\rho)} = \frac{1}{\mu(\mu-\lambda)} \)
    - since waiting time = delay - service time,
      - \( E(W) = E(D) - E(S) = \frac{\lambda}{\mu(\mu-\lambda)} \)
    - again, by Little’s Law, \( E(N_Q) = \lambda E(W) = \frac{\lambda^2}{\mu(\mu-\lambda)} = \frac{\rho^2}{1-\rho} \)
Other consequences: Back to delays and algorithms

- We have seen that bad scheduling algorithms can cause bad delay
  - let’s understand this better

- Suppose we have
  - $N$ queues at which packets arrive (distribution of arrival process doesn’t matter)
  - and there is a single server who serves
    (1) from the longest queue (LQF)
    (2) from the shortest queue (SQF)
  - each packet takes one unit of time to serve (or transmit)

- Let
  - $(X^l_1(t), ..., X^l_N(t))$ be the queue sizes under LQF, and let $S^l(t) = \sum_{i=1}^{N} X^l_i(t)$
  - $(X^s_1(t), ..., X^s_N(t))$ be queue sizes under SQF, let $S^s(t) = \sum_{i=1}^{N} X^s_i(t)$
Under identical inputs we know that

- $S^l(t) = S^s(t)$ for every $t$
- therefore, $E(S^l(t)) = E(S^s(t)) \Rightarrow \sum_{i=1}^{N} E(X^l_i(t)) = \sum_{i=1}^{N} E(X^s_i(t))$
- by symmetry (identical distribution of the arrivals to the $N$ queues) we get
  
  \[ E(X^l_i(t)) = E(X^l_1(t)) \text{ and } E(X^s_i(t)) = E(X^s_1(t)) \]
- therefore, $E(S^l(t)) = E(S^s(t)) \Rightarrow E(X^l_i(t)) = E(X^s_i(t))$

$\rightarrow$ that is, the average occupancies under LQF and SQF are the same!

To summarize

- throughputs under both LQF and SQF are the same
  (because the server works just as often under both policies)
- furthermore, we have seen the average occupancies are the same
- by Little’s Law, this means the average delays are also the same
  (because arrival rates are equal under both policies)
- what gives? i.e. does this mean the two policies are the same
  so far as throughput and delay are concerned?
Let’s look at an example

- Take a 16 queue example
  
  - arrivals at each queue are Bernoulli(\(\frac{1}{17}\)) IID
  - therefore, total load equals \(16/17 = 0.9411\)
  - the simulation results:
    
    LQF: Average queue = 0.49959, average delay = 8.49206
    SQF: Average queue = 0.498332, average delay = 8.47078
  
  - sanity check: apply Little’s Law
    
    \(\frac{1}{17} \times 8.49206 = 0.49995\) (Little miracles!)
— LQF keeps queue-sizes better balanced. Hence, queue-sizes are tightly distributed around the mean value. Under SQF a small queue remains small for a long time, and a long queue will remain long for a long time. The load distribution under SQF is very uneven, even though the average load under SQF is the same as in LQF.
Poisson Arrivals See Time Averages (PASTA)

- Time and arrival averages. Consider a queue in equilibrium
  - time average: $\pi_n = P(Q(t) = n)$, where $t$ is an arbitrary instance of time
  - arrival average: $a_n = P(\text{arriving packet sees } n \text{ packets in queue})$
  - Question: when is $a_n = \pi_n$?

- Let’s begin with a “bad” example
  - consider a queue where services are deterministic, equal to 3
  - packets arrive at 1, 2, 3, 11, 12, 13, 21, 22, 23, ... 
  - the evolution of the queue looks like ...
• Computing the probabilities

- \( \pi_0 = 1/10, \pi_1 = 4/10, \pi_2 = 4/10, \pi_3 = 1/10 \)
- \( a_0 = 1/3, a_1 = 1/3, a_2 = 1/3 \)

→ these are totally different!

- therefore, in general, time averages are not equal to arrival averages

• But

- if arrivals are Poisson (or Bernoulli, in discrete-time)
- then (we shall see that) time averages equal arrival averages
- this is PASTA - Poisson arrivals see time averages

• Implications

- to avoid falling victim to sampling bias, choose sampling instants completely randomly
- this makes sense (i.e. like random inspection)
- the beauty of the Poisson (or Bernoulli) process is that future arrivals occur completely independently of past arrivals, and past arrivals are what influence the present queue size!
Consider a queue

- at which packets arrive as a Poisson process of rate $\lambda$
- the service distribution is arbitrary
- let $\pi_n$ be the prob that queue-size equals $n$ at an arbitrary time $t$, say
- let $a_n$ be the prob that an arrival sees a queue of size $n$
- then we get

$$a_n = P(Q(t) = n | A(t, t + \delta) = 1)$$

$$\overset{(a)}{=} \frac{P(Q(t) = n, A(t, t + \delta) = 1)}{P(A(t, t + \delta) = 1)}$$

$$\overset{(b)}{=} \frac{P(Q(t) = n) P(A(t, t + dt) = 1)}{P(A(t, t + dt) = 1)}$$

$$= P(Q(t) = n) = \pi_n,$$

- where (a) is due to Bayes’ law
- and (b) uses the fact that arrivals after time $t$ are independent of the queue-size at time $t$
→ therefore arrival averages equal time averages
The M/GI/1 Queue and the Pollaczek-Khinchine Formula

• Specification

- arrivals occur according to a rate \( \lambda \) Poisson process
  therefore, interarrival times are \( \exp(\lambda) \): \( P(A > t) = e^{-\lambda t} \)
- service times are IID and \emph{arbitrarily} distributed with
  i.e. mean \( E(S) = \frac{1}{\mu} \) and second moment \( E(S^2) \)

• Let the queue be in equilibrium and let

1. \( W(t) \) be the unfinished work in the system at time \( t \)
   - i.e. \( W(t) \) is the amount of time from time \( t \) the server will be busy,
     if no new work was allowed into the system after time \( t \)
   - it is also the amount of time that a packet arriving at time \( t \) will have
     to wait before being served
   - \( W(t) \) is called the “work-load process” or the “waiting time process”

2. \( S_i \) and \( W_i \) be the service and waiting times, respectively, of packet \( i \).
Now

- notice that

\[
\frac{\int_0^T W(t) \, dt}{T} = \sum_{k=1}^{A(T)} \frac{A_k}{T},
\]

where \( A_k \) is the area of each of the triangles and the adjoining parallelogram.

- thus

\[
\frac{\int_0^T W(t) \, dt}{T} = \frac{1}{T} \sum_{k=1}^{A(T)} \left( \frac{S_k^2}{2} + W_k S_k \right) = \frac{A(T)}{T} \frac{1}{A(T)} \sum_{k=1}^{A(T)} \left( \frac{S_k^2}{2} + W_k S_k \right)
\]

- letting \( T \to \infty \), we get

\[
E(W_{time}) = \lambda \left( \frac{ES^2}{2} + E(W_1 S_1) \right) = \lambda \left( \frac{ES^2}{2} + E(W_1) E(S_1) \right)
\]
- but since the arrivals are Poisson, by PASTA, $E(W_{time}) = E(W_1)!
- therefore, letting $E(W_{time}) = E(W_1) = E(W)$, we get that

$$E(W) = \frac{\lambda ES^2}{2(1 - \lambda E(S))} = \frac{\lambda ES^2}{2(1 - \rho)}$$

- this is the famous Pollaczek-Khinchine formula

**Corollary: Determinism minimizes waiting**

- consider an M/GI/1 queue
- allow the service time to be arbitrary, except that $E(S) = \frac{1}{\mu}$
- since $Var(S) = ES^2 - (ES)^2 \geq 0$, we get that $ES^2 \geq (ES)^2 = \frac{1}{\mu^2}$.
- equality is achieved iff $Var(S) = 0$; i.e. iff $S$ is deterministic!
- this implies $E(W) = \frac{\lambda ES^2}{2(1-\rho)}$ is smallest for deterministic services
  (since the numerator is smallest in this case)
- or, determinism minimizes waiting
Stability

- Stability pertains to the question:
  1. When does a system in which work arrives according to certain inputs and in which work is removed in some fashion (i.e. service algorithms) not allow work to become unbounded?
  2. For example, if the system is describable as a Markov chain, when is there an equilibrium such that the expected size is finite?
  3. For us, it is equivalent to the question when do switches or networks achieve 100% throughput?

- In the next few lectures
  - we will understand what stability means
  - find methods to establish the stability of algorithms
  - know how to devise and use Lyapunov functions
An example

- Take 32 queues with dedicated servers
  - service rate of each server = 1 packet/sec, service distributions are identical
  - let packets arrive at rate $\lambda$
  - packets are assigned upon arrival to:
    1. the shortest queue
    2. the longest queue

- What happens?
  - under join the shortest queue
    - we get a maximum throughput of upto 32: $\lambda$ can go upto (but less than) 32
  - under join the longest queue
    - the maximum throughput is only 1: $\lambda$ cannot increase beyond 1
    → the LQF policy completely underutilizes the capacity of the system
    → note: different stability regions under different schedules for same architecture
Stability

• In the previous example it was pretty obvious what was wrong with LQF

• In general, in switch scheduling algorithms
  - it is far from obvious why a certain algorithm is not giving 100% throughput
  - often, we know why an algorithm doesn’t do well by looking at counterexamples
    - i.e. traffic patterns we construct that expose weaknesses in the algorithm
  - likewise, we often know why an algorithm does well both based on intuition and when we construct a proof to show that it gives 100% throughput
    - the proofs usually involve the construction of Lyapunov (or potential) functions
  → we will learn more about counterexamples and about Lyapunov functions in the next few lectures