Introduction to Queuing Theory
and
Mathematical Modelling

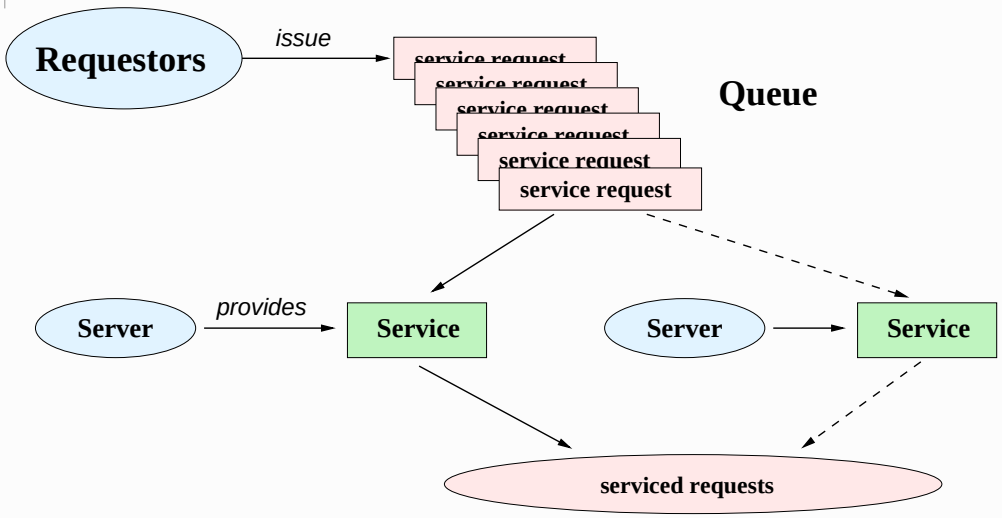
Computer Science 742 S2C, 2014

Nevil Brownlee, *with acknowledgements to*
Peter Fenwick, Ulrich Speidel and Ilze Ziedins

Overview

- Introduction, queuing models
- Mathematics background
  - Random variables
  - Renewal processes
  - Poisson processes
- Queuing theory
  - Kendall notation of queuing *problems*
  - Finding a distibution
  - Little's formula, PASTA

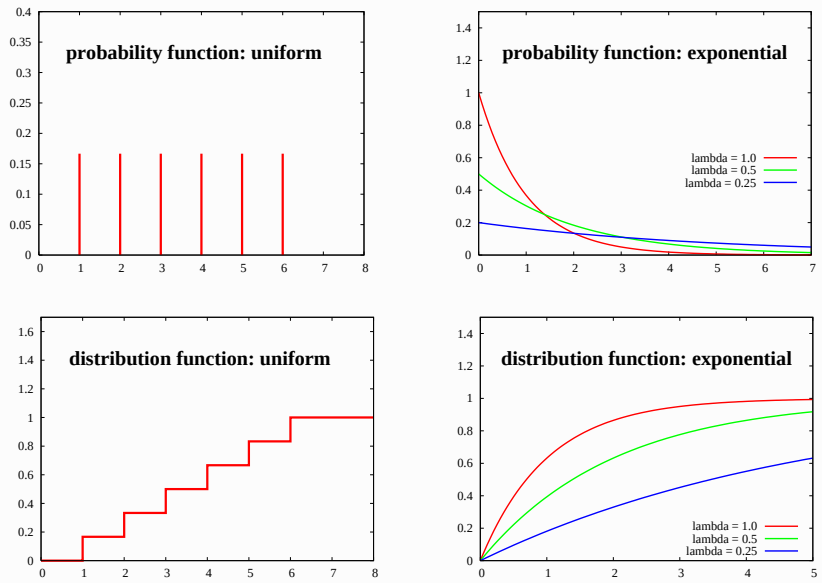
Queuing system diagram



Random variables

- A random variable X is a function that assigns a real-number value to each outcome of an experiment
- Usually described by a *probability density function* $f(x)$ and a *distribution function* $F(x) = \int_{-\infty}^{x} f(u)du$
- $f(x)$ tells how likely it is that X's value will be near x. Note that $f(x)$ can be > 1.
- X has an *exponential distribution* with parameter λ if it has
  - Density function $f(x) = \lambda e^{-\lambda x}; \quad x \geq 0$
  - Distribution function $F(x) = 1 - e^{-\lambda x}; \; x \geq 0$

Probabilities: 6-sided die, exponential



Renewal Processes

- Consider a sequence of events which happen first at time $T_0 = 0$, then keep happening at random intervals
- The events occur at times $T_n (n = 0, 1, ...;\ T_0 = 0)$, and $S_n = T_n - T_{n-1} (n = 1, 2, ...)$ are the times between them, often called *renewal periods*
- If the random variables S_n are independent and identically distributed *(iid)*, then the sequence $\{T_n; n = 0, 1, ...\}$ is a *renewal process*
- Renewal processes are useful for modelling streams of packets on a wire, jobs to be processed, etc.
- A renewal process is completely characterised by the common distribution function, $F(x)$, or the density function, $f(x)$ (if it exists), of its renewal periods

Poisson processes (1)

- A renewal process with exponentially distributed renewal periods S, i.e. $F(x) = 1 - e^{-\lambda x}$, is called a *Poisson process*
- Poisson processes are often used in modelling. They derive several useful properties from the exponential distribution, as follows ..
- Lack of memory:
  $$P(S \leq t + \Delta t \mid S > t) = P(S \leq t);\ s, t \geq 0$$
  - Knowing that the process has been running for time t doesn't affect its distribution for the remaining time (i.e. the time until the next event)
  - The process *forgets its past*

Poisson processes (2)

- Uniform arrival rate:
  $$P(S \leq t + \Delta t \mid S > t) \approx \lambda\, \Delta t, for\ small\ \Delta t$$
  - With a Poisson process, arrivals occur at an average rate λ
  - This implies PASTA:
    Poisson Arrivals See Time Averages
  - A Poisson arrival acts as a random observer and sees the queue in equilibrium
- Superposition:
  - If $A_1, A_2, ...A_n$ are independent Poisson processes with rates $\lambda_1, \lambda_2, ...\lambda_n$, their superposition is also a Poisson process, with rate $\lambda_1 + \lambda_2 ... + \lambda_n$
  - A Poisson process can also be decomposed into a set of Poisson processes

Poisson distribution

- For a Poisson process, i.e. exponential distribution of interarrival times (renewal periods):
  - Probability that n arrivals occur in interval of length t is $P_n(t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t}$
  - This formula is the *Poisson distribution* with parameter λt, for which $Mean = \lambda t$, and $Var = \lambda t$
  - λ is the mean *rate,* i.e. λ events occur per unit time
- More generally . . .
  - *Mean* of random variable X: $Mean(X) = E[X] = \int_{-\infty}^{\infty} x\, f(x) dx$
  - *Variance* of X: $Var(X) = \sigma^2 = E[(X - E[X])^2]$

Kendall notation of a queuing *problem*

- *ArrDist/ServDist/Servers/Buffers/Population/ServDisc*
- A queuing problem can be described by its
  - arrival distribution (arrival times of service requests)
  - service distribution (time server takes to service a request)
  - number of available servers
  - buffers (total number of possible service requests in the system)
  - population (total number of possible requests)
  - service discipline (in which order do we deal with requests?)
- We often only give first three, e.g. $M/M/1$
  other parameters take default values, i.e. $\infty/\infty/FIFO$

Arrival time distributions

- Need to model the arrival process (customers coming through bank door, packets arriving at router)
- Some processes have highly predictable arrival processes (e.g., a plane lands and 100 passengers get off), others have a less deterministic nature (e.g., customers arriving at a bank)
- Arrival processes are renewal processes
- Can often model arrivals using statistical distributions e.g. (Kendall notation in parentheses),Exponential (M), deterministic (D), Erlang with parameter k (E_k)
- The thing we're usually interested in is the interarrival time; average arrival rate (arrivals per time unit) is denoted as λ where applicable

Deterministic arrival distributions (D)

- Very simple: All inter-arrival times τ are constant!
- $\tau = 1/\lambda$

Exponential arrival distributions (M)

- Exponential arrival distributions are perhaps the most common apart from deterministic ones
- Approximately exponentially distributed, e.g. the time until you next meet a friend you haven't heard from in 2 years, the time the next customer walks through the door at your local supermarket, etc.
- Exponential distribution: $P(S \leq t) = \int_0^t \lambda\, e^{-\lambda u} du$
- S = time between two arrivals (random variable)
- Probability next arrival occurs in t seconds is $P(S \leq t)$.
- Mean of S is $1/\lambda$, standard deviation is also $1/\lambda$.
  (Remember: λ *is the mean arrival rate*)
- Memoryless

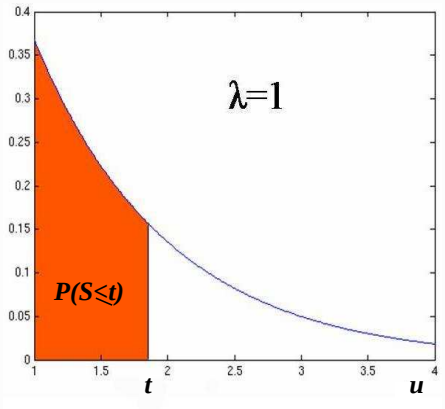
Erlang arrival distributions (E_k)

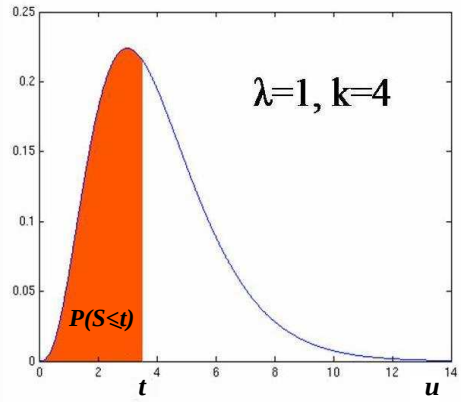
- Interarrival time probability given by

$$P(S \leq t) = \int_0^t \lambda \frac{(\lambda u)^{k-1}}{(k-1)!}\, e^{-\lambda u} du$$

- Values ≥ 0, two parameters: $Mean = 1/\lambda$, shape k,
  – often more realistic than plain exponential
- Applies to a cascade of servers with exponential distribution times, such that a customer can't be started until the previous one has been completely processed
- When k is integer, Erlang distribution is sum of k independent exponential distributions (*gamma* function)
- Note that the exponential distribution results for $k = 1$

Exponential and Erlang plots





Service time distributions

- Service time: time that a server needs to deal with a service request. For example, the time it takes to re-fuel a car or the time it takes to route a packet at a router
- Average service time is often denoted as $1/\mu$, where μ is the average service rate (number of requests serviced per time unit) per server
- Can model these service times via distributions – basically the same as for arrival time distributions.
  Just use μ instead of λ

Number of servers

- The number of available servers, n is obviously a very important parameter of a queuing system, e.g., number of pumps at the service station
- A queuing system can have either a separate queue for each server, or a common queue for all servers
- Kendall notation says nothing about common or separate queues
- Total service rate is $n\mu$

Buffer size

- The maximum number of service requests that can be in the system (queued or being serviced) at any one time
- This number may be unlimited (the queue just gets longer), in which case the Kendall notation omits it . . .
- . . . or limited (e.g., limited number of buffers in a packet switch)
- Note that if $\lambda < n\mu$, i.e. *arrival rate* $<$ *service rate,* we can often assume – even for relatively small actual population sizes – that the buffer size is unlimited

Population size

- Total number of possible service requests at any instant in time, may be limited or unlimited
- Example: Computer Science has N students. The maximum number of students needing a lab computer at any one time is therefore limited to N. They are not going to be able to do their assignments any faster if we give them more than N lab computers to work on
- Can often assume that the number is unlimited. For example, assume that we have an unlimited number of CS students as it is unlikely that all of them will ever turn up in the lab at once
- Default in Kendall notation is unlimited (value is omitted)

Queuing/service discipline

- How do we process requests in the queue?
- First-come-first-served (FCFS)?
- Last-come-first-served (LCFS)?
- Request-dependent, e.g., quick/easy jobs first/last?
- According to request priority?
- Default in Kendall notation is FCFS (value is omitted in this case)

Kendall notation again

- So what's an *M/D/2/12/200/FCFS* queue then?
- Answer:
  - exponentially distributed interarrival times *(Markov)*
  - fixed service time *(Deterministic)*
  - two servers
  - up to 10 places in the queue (plus two being served)
  - at most 200 possible requests at any one time, i.e., not all requests may make the queue
  - requests that arrive first get serviced first *(FCFS)*

Queue Occupation Rate, some simple insights

- Consider a $G/G/n$ queue (with G being any possible distribution)
- If λ is the arrival rate and μ is the service rate of a single server, then
  - The queue will grow to infinity if $\lambda > n\mu$
  - The queue will also grow to infinity if $\lambda = n\mu$ *(random walk),* except if it's a $D/D/n$ queue
  - The quantity $\rho = \lambda/n\mu$ is called the "occupation rate." It states the average portion of time that each server is busy
- Reference: *Introduction to Operations Research,* Hillier & Lieberman, McGraw Hill

Little's formula

- Sometimes referred to as *Little's law* or *result*
- States that the expected number of requests in the queuing system N (in queue and being processed) is given by

$$N = \lambda T$$

  where T is the expected time that a request will spend in the system
- Alternatively $T = N/\lambda$ or
  $time\ in\ system = number\ in\ system / av\ interarrival\ time$