

# Computer Systems Modelling

R.J. Gibbens

Computer Laboratory  
University of Cambridge

Michaelmas Term 2011/12

*Last revision: 2011-11-25/r-15*

## Twelve lectures covering:

- ▶ **Introduction to modelling**: what is it, why is it useful?
- ▶ **Simulation techniques**: random number generation, Monte Carlo simulation techniques, statistical analysis of results from simulation and measurements;
- ▶ **Queueing theory**: Applications of Markov chains, single/multiple servers, queues with finite/infinite buffers, queueing networks.

## Recommended books



Ross, S.M.

*Probability Models for Computer Science*

Academic Press, 2002



Mitzenmacher, M & Upfal, E.

*Probability and computing: randomized algorithms and probabilistic analysis*

Cambridge University Press, 2005



Jain, A.R.

*The Art of Computer Systems Performance Analysis*

Wiley, 1991



Kleinrock, L.

*Queueing Systems — Volume 1: Theory*

Wiley, 1975

# Introduction to modelling

## Why model?

- ▶ A manufacturer may have a range of compatible systems with different characteristics — which configuration would be best for a particular application?
- ▶ A system is performing poorly — what should be done to improve it? Which problems should be tackled first?
- ▶ Fundamental design decisions may affect performance of a system. A model can be used as part of the design process to avoid bad decisions and to help quantify a cost/benefit analysis.

## A toy problem

system	CPU time	disk time	total
A	4.6	4.0	8.6
B1	5.1	1.9	7.0
B2	3.1	1.9	5.0

- ▶ A database running on Type **A** system is too slow
- ▶ Type **B1** system available immediately, and a type **B2** system in the future
- ▶ Which option is best:
  - ▶ Stick with **A**?
  - ▶ Change to **B1** immediately?
  - ▶ Wait and change to **B2**?
- ▶ What factors affect the correct choice?

# How can modelling help?

Typical performance questions we might ask include:

- ▶ How long will a database request wait before receiving CPU service?
- ▶ What is the utilization of the resource (CPU, disk, ...)? (Utilization is the proportion of time that the resource is busy.)
- ▶ What is the distribution of the number of requests queued at some time  $t$ ? What is its mean, standard deviation, ...

# Techniques

Many different approaches:

- ▶ **Measurement** — if the system already exists then maybe it can be changed and the effects observed and analysed
- ▶ **Simulation** — construct a computer program that emulates the system under study and study that instead
- ▶ **Queueing theory** — analytical models of queueing systems based on stochastic processes
- ▶ **Operational analysis** — analysis based on directly measured quantities and relationships between them: makes few assumptions about the system (not covered in the course in recent years)



## Techniques (2)

Choice of technique depends on ...

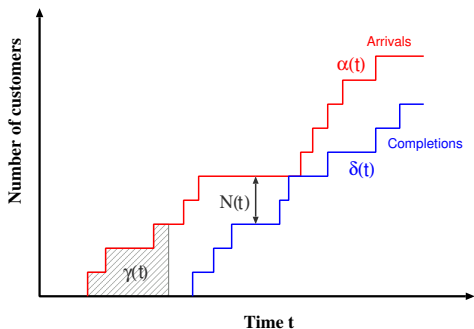
- ▶ **Stage of development:** can only measure an existing system
- ▶ **Time available:** measurements or simulations can take a long time to complete. How easily can different trade-offs be evaluated?
- ▶ **Resources:** systems with which to experiment, people with the relevant skills, cost
- ▶ **Desired accuracy:** how do the assumptions made in analytic techniques effect the result? Are appropriate parameters and workloads used during experimental work?
- ▶ **Credibility:** will people believe (and act on) the results?

## Little's result

Begin with a simple derivation of **Little's Result** — a very general theorem relating the number of jobs in a system with the time they spend there.

**For example:** A disk server takes, on average, 10ms to satisfy an I/O request. If the request rate is 100 per second, then how many requests are queued at the server?

## Little's result (2)



$\alpha(t)$  = number of arrivals in  $(0, t)$

$\delta(t)$  = number of departures in  $(0, t)$

$\alpha(t) - \delta(t)$  = number in the system at  $t$

The area  $\gamma(t)$  between the curves  $\alpha(t)$  and  $\delta(t)$  during the period  $(0, t)$  represents the total time all customers have spent in the system in  $(0, t)$ .

## Little's result (3)

Let

$$\lambda(t) = \alpha(t)/t$$

$$T(t) = \gamma(t)/\alpha(t)$$

$$N(t) = \gamma(t)/t.$$

So that,

- ▶  $\lambda(t)$  — the average arrival rate during  $(0, t)$ ;
- ▶  $T(t)$  — system time per customer averaged over all customers in  $(0, t)$ ;
- ▶  $N(t)$  — average number of customers in system during  $(0, t)$ .

Combining these gives that

$$N(t) = \lambda(t)T(t).$$

## Little's result (4)

Assume the following limits exist

$$\lambda = \lim_{t \rightarrow \infty} \lambda(t)$$
$$T = \lim_{t \rightarrow \infty} T(t).$$

Then we have

$$N = \lim_{t \rightarrow \infty} N(t) = \lim_{t \rightarrow \infty} \lambda(t) T(t) = \lambda T.$$

That is, the average number in the system,  $N$ , equals the average arrival rate  $\times$  average time in system.

This is **Little's result**. The proof makes no assumptions about the way that arrivals or departures are distributed, the queueing discipline or how many servers service the queue.

First proved by John Little in 1961.

## Applications of Little's result

We can re-state this result for any boundary of our queueing system.

Split  $T$  (average time in the system) into  $T_w$  (average time spent waiting) and  $T_s$  (average time spent being served).

Similarly, we can split  $N$  (average number in the system) into  $N_w$  (average number waiting in the queue) and  $N_s$  (average number being served).

Applying Little's result separately to the queue and to the server:

$$N_w = \lambda T_w$$

$$N_s = \lambda T_s$$

# Continuous distributions

We can typically describe the distribution of a **continuous** random variable,  $X$ , in two ways using:

- ▶ either the **cumulative distribution function** (cdf), (or just the **distribution function**)

$$F_X(x) := \mathbb{P}(X \leq x)$$

- ▶ or the **probability density function** (pdf)

$$f_X(x) := \frac{dF_X(x)}{dx}.$$

Note that

$$0 \leq F_X(x) = \int_{-\infty}^x f_X(y) dy$$

and that  $F_X(x)$  increases with  $x$  up to the value

$$F_X(\infty) = \int_{-\infty}^{\infty} f_X(y) dy = 1.$$

## Expected value and moments

The **expected value** of  $X$ , written  $\mathbb{E}(X)$ , is given by

$$\mathbb{E}(X) := \int_{-\infty}^{\infty} x f_X(x) dx.$$

Also called the average, mean or first moment of the distribution and sometimes written as  $\bar{X}$ .

The  **$n^{\text{th}}$  moment** is defined as

$$\mathbb{E}(X^n) := \int_{-\infty}^{\infty} x^n f_X(x) dx.$$

The  **$n^{\text{th}}$  central moment** is defined as

$$\mathbb{E}((X - \mathbb{E}(X))^n) := \int_{-\infty}^{\infty} (x - \mathbb{E}(X))^n f_X(x) dx.$$



## Variance, standard deviation & coefficient of variation

The 2<sup>nd</sup> central moment,  $\text{Var}(X)$ , of a random variable  $X$  is known as the **variance**,

$$\text{Var}(X) = \mathbb{E}((X - \mathbb{E}(X))^2) = \mathbb{E}(X^2) - (\mathbb{E}(X))^2.$$

From this, we define the **standard deviation** by  $\sqrt{\text{Var}(X)}$  and the **coefficient of variation** by the dimensionless quantity

$$C_X := \frac{\text{standard deviation}}{\text{mean}} = \frac{\sqrt{\text{Var}(X)}}{\mathbb{E}(X)}.$$

Numerically larger values of  $C_X$  signify “**more variable**” data. For example, a coefficient of variation of 5 might be considered large, while 0.2 might be considered small.

## Exponential distribution $\text{Exp}(\lambda)$

Given a scale parameter,  $\lambda > 0$ , the (positive) random variable  $X$  has the **exponential** distribution  $\text{Exp}(\lambda)$  defined by the density and distribution function

$$f_X(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$F_X(x) = \begin{cases} 1 - e^{-\lambda x} & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases} .$$

**Exercise:** show that the mean of  $X$  is  $\frac{1}{\lambda}$  and the variance is  $\frac{1}{\lambda^2}$ . Hence for this distribution the mean and standard deviation are equal and so

$$C_X = \frac{\sqrt{\text{Var}(X)}}{\mathbb{E}(X)} = \frac{\sqrt{\frac{1}{\lambda^2}}}{\frac{1}{\lambda}} = 1 .$$

# Memoryless property

The exponential distribution is the only continuous distribution with the **Memoryless Property**, namely, that

$$\mathbb{P}(X > t + s \mid X > t) = \mathbb{P}(X > s)$$

Intuitively, it may be used to model the distribution of inter-event times in which the time until the next event does not depend on the time that has already elapsed.

If the inter-event times are independent, identically distributed random variables with the  $\text{Exp}(\lambda)$  distribution then  $\lambda$  is viewed as the **mean event rate**.

# Erlang distribution

A random variable,  $X$ , with pdf

$$f_X(x) = \begin{cases} \lambda e^{-\lambda x} \frac{(\lambda x)^{n-1}}{(n-1)!} & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

where  $n = 1, 2, \dots$  and  $\lambda > 0$  has an **Erlang distribution** with parameters  $n$  and  $\lambda$ . The distribution function can be computed from

$$F_X(x) = \int_0^x f_X(y) dy.$$

The mean and variance of  $X$  can be shown to be  $n/\lambda$  and  $n/\lambda^2$ , respectively.

It is the case that the sum of  $n$  independent  $\text{Exp}(\lambda)$  random variables has an Erlang distribution with parameters  $n$  and  $\lambda$  — see more on this latter.

In fact, it is possible to generalize this distribution to where the parameter  $n$  is any positive real number. This more general case is known as the **Gamma distribution**.

## Normal distribution $N(\mu, \sigma^2)$

A random variable,  $X$ , has a normal distribution, written  $N(\mu, \sigma^2)$ , if its pdf is given by

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}$$

where  $\mu$  is any real number and  $\sigma > 0$ . Recall that the mean of  $X$  is  $\mu$  and the variance is  $\sigma^2$ .

Thus the **standardized** random variable

$$Z = \frac{X - \mu}{\sigma}$$

has a normal distribution with mean 0 and variance 1. The distribution function of  $Z$  is usually written

$$F_Z(x) = \Phi(x).$$

Notice that then

$$F_X(x) = \Phi\left(\frac{x - \mu}{\sigma}\right).$$

# Central Limit Theorem (CLT)

Suppose that  $X_1, X_2, \dots$  is a sequence of independent, identically distributed random variables (each with finite mean  $\mu$  and finite variance  $\sigma^2$ ) then the CLT says that

$$\lim_{n \rightarrow \infty} \mathbb{P} \left( \frac{S_n - n\mu}{\sqrt{n}\sigma} < x \right) = \Phi(x)$$

where  $S_n = X_1 + X_2 + \dots + X_n$  so that the mean is  $\mathbb{E}(S_n) = n\mu$  and the standard deviation is  $\sqrt{\text{Var}(S_n)} = \sqrt{n\sigma^2} = \sqrt{n}\sigma$ .

Notice that the individual random variables  $X_1, X_2, \dots$  are **not** assumed to have normal distributions.

# Discrete distributions

The previous examples have concerned **continuous** random variables whose distributions have been defined by their **density** or, equivalently, their **distribution** functions.

Similar definitions apply to the case of **discrete** random variables,  $X$ , taking values  $x_i$  ( $i \in I$ ), where the distribution is specified by the **probability distribution function** (pdf)

$$0 \leq \mathbb{P}(X = x_i) \leq 1 \quad \forall i \in I$$

and where

$$\sum_{i \in I} \mathbb{P}(X = x_i) = 1.$$

## Expected value and variance

The **expected value** of  $X$  is

$$\mathbb{E}(X) := \sum_{i \in I} x_i \mathbb{P}(X = x_i).$$

Similarly, for the other moments, where the integration for continuous random variables becomes a summation over the set of possible values.

So, for example, we have that

$$\mathbb{E}(X^2) = \sum_{i \in I} x_i^2 \mathbb{P}(X = x_i)$$

and

$$\text{Var}(X) := \mathbb{E}((X - \mathbb{E}(X))^2) = \mathbb{E}(X^2) - (\mathbb{E}(X))^2$$

just as with continuous random variables.



## Bernoulli( $p$ ) distribution

The random variable  $X$  has a **Bernoulli distribution** if it takes just two values either  $X = 0$  or  $X = 1$  with probabilities

$$\mathbb{P}(X = x) = \begin{cases} p & \text{if } x = 1 \\ 1 - p & \text{if } x = 0 \end{cases}$$

We say that  $p = \mathbb{P}(X = 1)$  is the **probability of success** ( $0 \leq p \leq 1$ ) in a Bernoulli trial and  $1 - p = \mathbb{P}(X = 0)$  is the **probability of failure**.  
The mean and variance of  $X$  are  $p$  and  $p(1 - p)$ , respectively.

## Binomial( $n, p$ ) distribution

The (random) number of successes in a fixed length sequence of independent Bernoulli trials has a distribution known as the **Binomial distribution**. The pdf is given by

$$\mathbb{P}(X = x) = \binom{n}{x} p^x (1 - p)^{n-x} \quad x = 0, 1, \dots, n$$

where  $p$  is the probability of success of an individual Bernoulli trial and  $n$  is the fixed number of trials. Thus the parameters satisfy  $0 \leq p \leq 1$  and  $n = 1, 2, 3, \dots$

The mean and variance of  $X$  are  $np$  and  $np(1 - p)$ , respectively. Note that  $\mathbb{P}(X = x)$  is the product of the number of ways that  $x$  successes can occur in  $n$  trials and the probability that exactly that pattern of successes and failures occurs.

## Poisson( $\lambda$ ) distribution

The random variable  $X$  has a **Poisson** distribution if it takes values  $0, 1, 2, \dots$  with probabilities

$$\mathbb{P}(X = i) = e^{-\lambda} \frac{\lambda^i}{i!} \quad i = 0, 1, 2, \dots$$

where  $\lambda > 0$  is a scale parameter.

**Exercise:** show that both the mean and variance of  $X$  are both equal to  $\lambda$ .

The Poisson( $\lambda$ ) distribution is a good approximation to a Binomial( $n, p$ ) distribution when  $\lambda = np$  and when the number of trials,  $n$ , is large and the probability of success,  $p$ , is small.

## Geometric( $p$ ) distribution

Given a sequence of independent Bernoulli trials, each with probability of success  $p$  how long do we wait till the first successful trial?

The number of trials,  $X$ , up to and including the first successful trial has a distribution called the **Geometric** distribution given by

$$\mathbb{P}(X = n) = p(1 - p)^{n-1} \quad n = 1, 2, \dots$$

The mean of  $X$  is given by  $1/p$  and the variance by  $(1 - p)/p^2$ .

# The Poisson process

Consider a process of events occurring at random points of time and let  $N(t)$  be the number of events that occur in the interval  $[0, t]$ . A **Poisson process** at rate  $\lambda$  ( $\lambda > 0$ ) is defined by the following conditions:

- ▶  $N(0) = 0$ ;
- ▶ The numbers of events in **disjoint** time intervals are independent and the distribution of the number of events in a interval depends only on its length (and not its location);
- ▶

$$\mathbb{P}(N(h) = i) = \begin{cases} 1 - \lambda h + o(h) & i = 0 \\ \lambda h + o(h) & i = 1 \\ o(h) & \text{otherwise.} \end{cases}$$

A quantity  $g(h) = o(h)$  if  $\lim_{h \rightarrow 0} g(h)/h = 0$ .

## The Poisson process (2)

Consider the number of events,  $N(t)$ , occurring in an interval of length  $t$ . Divide the interval into  $n$  nonoverlapping subintervals each of length  $h = t/n$ .

A subinterval contains a single event with probability approximately  $\lambda(t/n)$  and so it follows that the number of such subintervals is approximately a Binomial random variable with parameters  $n$  and  $p = \lambda t/n$ .

Letting  $n \rightarrow \infty$ , shows that  $N(t)$ , the number of events in  $[0, t]$ , is a Poisson random variable with parameter  $\lambda t$ .

## The Poisson process (3)

Given a Poisson process of rate  $\lambda$  let  $X_1$  be the time of the first event and for  $n > 1$  let  $X_n$  denote the time between the  $(n-1)^{st}$  and  $n^{th}$  events.

The sequence  $X_1, X_2, \dots$  gives us the sequence of **inter-event times** between the events in a Poisson process.

## The Poisson process (4)

To determine the distribution of  $X_1$  note that

$$\mathbb{P}(X_1 > t) = \mathbb{P}(N(t) = 0) = e^{-\lambda t}$$

since  $N(t)$  is a Poisson random variable with parameter  $\lambda t$ . Thus,  $X_1$  has an  $\text{Exp}(\lambda)$  distribution.

Now consider,  $X_2$  then

$$\begin{aligned}\mathbb{P}(X_2 > t | X_1 = s) &= \mathbb{P}(0 \text{ events in } (s, s+t] | X_1 = s) \\ &= \mathbb{P}(0 \text{ events in } (s, s+t]) \\ &= e^{-\lambda t}.\end{aligned}$$

The inter-event times  $X_1, X_2, \dots$  are independent, identically distributed random variables each with distribution  $\text{Exp}(\lambda)$ .



## Time for first $n$ events

Let  $S_n = \sum_{i=1}^n X_i$  be the (random) time for the first  $n$  events in a Poisson process. Then

$$\begin{aligned}\mathbb{P}(S_n \leq t) &= \mathbb{P}(N(t) \geq n) \\ &= \sum_{j=n}^{\infty} e^{-\lambda t} \frac{(\lambda t)^j}{j!}\end{aligned}$$

So, the pdf of  $S_n$  is given by differentiating

$$\begin{aligned}f_{S_n}(t) &= \sum_{j=n}^{\infty} j\lambda e^{-\lambda t} \frac{(\lambda t)^{j-1}}{j!} - \sum_{j=n}^{\infty} \lambda e^{-\lambda t} \frac{(\lambda t)^j}{j!} \\ &= \sum_{j=n}^{\infty} \lambda e^{-\lambda t} \frac{(\lambda t)^{j-1}}{(j-1)!} - \sum_{j=n}^{\infty} \lambda e^{-\lambda t} \frac{(\lambda t)^j}{j!} \\ &= \lambda e^{-\lambda t} \frac{(\lambda t)^{n-1}}{(n-1)!}.\end{aligned}$$

Thus,  $S_n = \sum_{j=1}^n X_i$  has an Erlang distribution with parameters  $n$  and  $\lambda$ .

# Non-homogeneous Poisson processes

The Poisson process has a constant rate of events,  $\lambda$ , but we can relax this assumption to use a **time-dependent** rate function  $\lambda(t)$  to produce a non-homogeneous Poisson process as follows.

- ▶  $N(0) = 0$ ;
- ▶ The numbers of events in disjoint time intervals are independent;
- ▶

$\mathbb{P}(\text{exactly } i \text{ events occur in } (t, t+h])$

$$= \begin{cases} 1 - \lambda(t)h + o(h) & i = 0 \\ \lambda(t)h + o(h) & i = 1 \\ o(h) & \text{otherwise.} \end{cases}$$

# Simulation techniques

# Introduction

The main building block of a simulation study is a source of **random variables**.

We will begin by looking at the generation of sources of  $U(0, 1)$  continuous random variables and then show how this leads to the generation of random variables with arbitrary distributions, both continuous and discrete.

## Random number generation

It is important **not** to generate random numbers with an ad hoc method — complex algorithms do not necessarily generate random outputs.

Some operating systems include support for generating random numbers based on (e.g.) key strokes or network inter-arrival times — however, these mechanisms are not usually suited to generating large volumes of random data.

How can we algorithmically generate a long **random** sequence?

The answer is that the sequence is not random, but appears random as far as can be determined from statistical tests. The sequence is termed **pseudo-random**.

## Random number generation (2)

Important requirements include

- ▶ The algorithm should be fast;
- ▶ The storage requirements should be low;
- ▶ The random sequence should only repeat after a very long period;
- ▶ The sequence generated should possess two important statistical properties: uniformity and independence.

For ease of implementation and speed, most random number generators use integer representation over an interval  $m$

Given a value in this range, a desired value in the range  $(0, 1)$  can be obtained by dividing by  $m$

## Multiplicative congruential method

A popular method, known as the **multiplicative congruential** method, starts with a **seed** value,  $X_0$ , and then recursively constructs the successive values  $X_n$  by the equation

$$X_n = (aX_{n-1}) \text{ modulo } m$$

where  $a$  and  $m$  are positive integers. The values  $X_n$  lie in the range  $0, 1, 2, \dots, m-1$ .

The sequence of values  $X_n$  has period at most  $m$  and so we should wish to choose  $a$  and  $m$  such that the period remains large whatever the seed value  $X_0$ .

A common choice is  $m = 2^{31} - 1$  and  $a = 7^5 = 16,807$ .

## Mixed congruential method

An extension to the multiplicative congruential method is the **mixed congruential** method which includes an additive term to the recursion

$$X_n = (aX_{n-1} + c) \text{ modulo } m$$

We will not investigate such methods further here. Instead we will assume that we have an efficiently generated supply of random numbers distributed as independent  $U(0, 1)$  random variables.



## Random variable generation

We now have a sequence of pseudo-random uniform variables. How do we generate variables from other distributions?

We will find that with a suitable transformation random behaviour can be programmed so that the resulting random variables appear to have been drawn from any desired probability distribution.

## Discrete distributions

Suppose we are given a distribution  $p_i$  with  $0 \leq p_i \leq 1$  for  $i \in I = \{0, 1, \dots\}$  and  $\sum_{i \in I} p_i = 1$  and that we wish to generate a discrete random variable,  $X$ , whose probability distribution function is

$$\mathbb{P}(X = x_i) = p_i \quad \forall i \in I.$$

This may be done by generating a (pseudo-)random variable  $U$  with distribution  $U(0, 1)$  and setting

$$X = \begin{cases} x_0 & \text{if } U < p_0 \\ x_1 & \text{if } p_0 \leq U < p_0 + p_1 \\ x_2 & \text{if } p_0 + p_1 \leq U < p_0 + p_1 + p_2 \\ \vdots & \\ x_i & \text{if } \sum_{j=0}^{i-1} p_j \leq U < \sum_{j=0}^i p_j \\ \vdots & \end{cases}$$

# The inverse transform method

For now since  $U$  is  $U(0, 1)$

$$\mathbb{P}(X = x_i) = \mathbb{P}\left(\sum_{j=0}^{i-1} p_j \leq U < \sum_{j=0}^i p_j\right) = p_i$$

If we write  $F(x_k) = \sum_{i=0}^k p_i$  then the process of generating  $X$  is given by

- ▶ Generate a random number  $U$
- ▶ If  $U < F(x_0)$  set  $X = x_0$  and stop
- ▶ If  $U < F(x_1)$  set  $X = x_1$  and stop
- ▶  $\vdots$

If the  $x_i$  are ordered so that  $x_0 < x_1 < \dots$  then this is equivalent to choosing  $X = x_i$  if  $F(x_{i-1}) \leq U < F(x_i)$  and for this reason the method is known as the **inverse transform method**.

## Geometric random variables

Here

$$p_i = \mathbb{P}(X = i) = p(1-p)^{i-1} \quad i = 1, 2, \dots$$

and so

$$\begin{aligned} \sum_{j=1}^{i-1} p_j &= 1 - \mathbb{P}(X > i-1) \\ &= 1 - (1-p)^{i-1}. \end{aligned}$$

Thus, we can use the inverse transform method by setting  $X$  to the value of  $i$  such that

$$1 - (1-p)^{i-1} \leq U < 1 - (1-p)^i.$$

A little algebra shows that we can write this as

$$X = \left\lfloor \frac{\log(U)}{\log(1-p)} \right\rfloor + 1.$$

## Poisson random variables

Here we have for  $\lambda > 0$

$$p_i = \mathbb{P}(X = i) = e^{-\lambda} \frac{\lambda^i}{i!} \quad i = 0, 1, \dots$$

Hence, it follows that

$$p_{i+1} = \frac{\lambda}{i+1} p_i \quad i = 0, 1, \dots$$

and an algorithm to generate a  $\text{Poisson}(\lambda)$  random variable is as follows.

- 1 Generate a random number  $U$
- 2 Set  $i = 0, p = e^{-\lambda}, F = p$
- 3 If  $U < F$ , set  $X = i$  and stop
- 4 Set  $p = \lambda p / (i + 1), F = F + p, i = i + 1$
- 5 Go to step 3

Clearly, similar algorithms can be formulated for other discrete distributions.

# Generating continuous random variables

Let  $U$  be a random variable with distribution  $U(0, 1)$  then

$$X = F_X^{-1}(U)$$

is a random variable with distribution function  $F_X(x)$ .

**Proof:**

$$\begin{aligned}\mathbb{P}(X \leq x) &= \mathbb{P}(F_X^{-1}(U) \leq x) \\ &= \mathbb{P}(F_X(F_X^{-1}(U)) \leq F_X(x)) \\ &= \mathbb{P}(U \leq F_X(x)) \\ &= F_X(x).\end{aligned}$$

So, the inverse transform method for continuous random variables with distribution function  $F_X(x)$  generates  $X = F_X^{-1}(U)$  where  $U$  is  $U(0, 1)$ .

## Uniform distribution $(a, b)$

Consider the **uniform** random variable on the interval  $(a, b)$  with distribution function

$$F_X(x) = (x - a)/(b - a)$$

for  $x$  in the interval  $(a, b)$ .

Given a pseudo-random uniform value  $U$  in the interval  $(0, 1)$  we set

$$X = F_X^{-1}(U)$$

and so

$$U = F_X(X) = (X - a)/(b - a)$$

so that

$$X = (b - a)U + a.$$

# Exponential distribution

For the exponential distribution with parameter  $\lambda$  we have

$$F_X(x) = \begin{cases} 1 - e^{-\lambda x} & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

and so given  $U$ , a pseudo-random variable with distribution  $U(0, 1)$ , we set

$$X = F_X^{-1}(U).$$

Thus

$$U = F_X(X) = 1 - e^{-\lambda X}$$

so that

$$X = -\frac{1}{\lambda} \log(1 - U).$$

Note that both  $U$  and  $1 - U$  are distributed as  $U(0, 1)$  so we might as well set

$$X = -\frac{1}{\lambda} \log(U).$$



## Alternative method for Poisson random variables

Recall that for a Poisson process of rate  $\lambda$ , the number of events in  $[0, 1]$ ,  $N(1)$ , is  $\text{Poisson}(\lambda)$ . Moreover, the inter-arrival times of events  $X_i$  are independent  $\text{Exp}(\lambda)$ . Hence,

$$N(1) = \max \left\{ n : \sum_{i=1}^n X_i \leq 1 \right\}.$$

Thus  $N = N(1)$  is a  $\text{Poisson}(\lambda)$  random variable where putting  $X_i = -\log(U_i)/\lambda$

$$\begin{aligned} N &= \max \left\{ n : \sum_{i=1}^n -\frac{1}{\lambda} \log U_i \leq 1 \right\} \\ &= \max \{ n : \log(U_1 U_2 \cdots U_n) \geq -\lambda \} \\ &= \max \{ n : U_1 U_2 \cdots U_n \geq e^{-\lambda} \} \\ &= \min \{ n : U_1 U_2 \cdots U_n < e^{-\lambda} \} - 1 \end{aligned}$$

## Simulating a Poisson process

Consider the problem of generating the first  $n$  event times of a Poisson process of rate  $\lambda$ . One way is to generate  $U_1, U_2, \dots, U_n$  random numbers each from a  $U(0, 1)$  distribution and then set  $X_j = -\frac{1}{\lambda} \log(U_j)$ . Then the first  $n$  event times are  $\sum_{i=1}^j X_i$  for  $j = 1, 2, \dots, n$ .

To generate the first  $T$  time units we could proceed as above and stop when the sum first exceeds  $T$ . Algorithmically,

- 1 Set  $t = 0, l = 0$
- 2 Generate a random number  $U$
- 3 Set  $t = t - \frac{1}{\lambda} \log(U)$ . If  $t > T$  stop
- 4 Set  $l = l + 1, S(l) = t$
- 5 Go to step 2

will build the sequence of event times in  $S(\cdot)$ .

# A simple queueing system



We characterise queueing systems by:

- ▶ Arrival process  $A(t) = \mathbb{P}(\text{inter-arrival time} \leq t)$
- ▶ Service process  $B(x) = \mathbb{P}(\text{service time} \leq x)$
- ▶ Storage capacity available for waiting customers
- ▶ The number of servers/customers available
- ▶ The different classes of arriving customers (big jobs, small jobs, ...)
- ▶ Queueing discipline used: FIFO, FCFS, LCFS, priority, ...
- ▶ Defections, balking, bribing, ...

## Queueing systems notation

The Kendall notation describes a single queueing system using the notation  $A/B/m/k/l$  where:

- ▶  $A$  is the inter-arrival time distribution of customers
- ▶  $B$  is the service time distribution
- ▶  $m$  is the number of parallel servers
- ▶  $k$  is the limit on the customers in this system
- ▶  $l$  is the population size

If the population size or the limit on the queue length are not specified then they are assumed to be infinite.

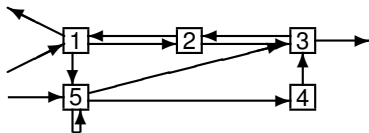
## Queueing notation (2)

- ▶  $M$  – exponential distribution (ie, memoryless)
- ▶  $E_r$  –  $r$ -stage Erlang distribution
- ▶  $D$  – Deterministic
- ▶  $G$  – General

### Examples:

- ▶  $M/M/1$ : exponential inter-arrival, exponential service, single server
- ▶  $M/E_r/1$ : exponential inter-arrival,  $r$ -stage Erlang service, single server
- ▶  $M/G/1$ : exponential inter-arrival, general service time, single server
- ▶  $M/M/K/K$ : exponential inter-arrival, exponential service,  $K$  servers and at most  $K$  customers present

# Queueing networks



More generally, consider systems comprising multiple inter-connected servers, forming a **queueing network**. Consider:

- ▶ the properties of each server
  - e.g. using Kendall notation
- ▶ the way in which jobs move between the servers
  - e.g. the links between servers and the ways jobs move between them
- ▶ the workload being analyzed
  - e.g. disk-server workload comprises a 20 : 1 mix of small/large requests

## Queueing networks (2)

We can classify queueing networks as either

- ▶ **closed** networks in which a fixed set of jobs circulate between servers, but no new jobs are introduced and no jobs leave the system
  - e.g. a computer system with a fixed number of terminals attached to it.
- ▶ **open** networks in which jobs may enter and leave the system
  - e.g. the network on the previous slide: jobs arrive at 1 or 5 and leave from 1 or 3.

Open networks may be further classified as **feed-forward** if a single job visits each server *at most* once.

# Simulation

Simulation allows arbitrarily complex systems to be evaluated.

- ▶ Able to capture the dynamic behaviour of systems
- ▶ The dynamics of complex systems are obtained by tracking the evolution of the system over time
- ▶ Examples include communication network design, road traffic modelling, etc.



## Simulation (2)

Execution of a simulation model consists of a series of state space changes.

We **always** follow the **dynamic** evolution of the system, even if we only want a mean value.

As well as techniques for implementing simulators it is necessary to know how to analyse their results.

Simulation is of particular use when we are studying systems that are not in steady state.

# Types of simulation

- ▶ **Discrete state/event** simulation in which the state of the system is described by discrete variables
  - e.g. the number of jobs at different stages on a production line
- ▶ **Continuous state/event** simulation in which the state is described by continuous variables
  - e.g. the quantities of various chemical reagents

A similar distinction may be drawn between **discrete time** and **continuous time** simulations depending on whether the system state is defined at certain discrete times or at all times.

## Types of simulation (2)

We will be concerned with **discrete event** simulation because it applies most naturally to computer systems in which state variables are generally discrete, e.g.

- ▶ the state of jobs in the system;
- ▶ the number of jobs of different kinds;
- ▶ the number or availability of devices.

## Pros and cons

The principal advantage of simulation is its extreme generality. However, . . .

- ▶ The design, coding and debugging of a simulation program are often time consuming and difficult to understand — the task may even approach that of implementing the system and measuring it directly!
- ▶ Generality can lead to complexity which can obscure understanding of the model — fine details may be irrelevant if the simulated workload is already a poor approximation.
- ▶ Execution of the simulation can be computationally expensive.
- ▶ Statistical analysis of the output can be problematic — e.g. how long to run the simulation before averaging the results?

# Events

Each event contains a time stamp identifying 'when it occurs' and denotes some change in the state of the system to be simulated e.g. 'IP packet arrived'

Events are ordered in time in an **event list**

Initialize the clock to 0
Initialize the event list
WHILE termination criterion is not met
remove earliest tuple $(t, m)$ from the event list
update the clock to $t$
simulate the effect of transmitting $m$ at time $t$

It is **crucial** that the simulator always selects the event with the **earliest** time stamp

Frequently most of the simulation time is spent maintaining the chronological order of the event list.

# Simulation variables

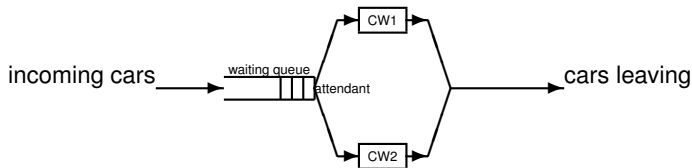
As we have seen, discrete event simulations involve both **events** and **variables** and where we keep track of variables as they change during a simulation.

There are several types of variables to consider.

- ▶ **Time variable**,  $t$ , to record the passage of time during the simulation;
- ▶ **Counter variables** which keep count of the number of times that certain events have occurred by time  $t$ ;
- ▶ **System state variables** which define the state of the system at time  $t$ .

As simulation events occur we change these variables and gather any output of interest.

## Example



An automatic car wash which is able to service one car at a time — is it viable to install a second car wash?

Model: attendant (**att**), car washes, (**cw1**, **cw2**), entrance (**source**) and exit (**sink**).

- ▶ The inter-arrival time of cars that need to be washed is randomly distributed
- ▶ If both car washes are busy, an arriving car joins the queue
- ▶ When a car wash becomes idle then the car at the head of the queue is sent to it.

## Example (2)

Events:

- ▶ **source**→**att**: car arrives in the system
- ▶ **att**→**cw**: car sent from the queue to the car wash
- ▶ **cw**→**att**: car wash becomes idle
- ▶ **cw**→**sink**: car departs the system

Note how the departure of a car is modelled by **two** events: one representing the car leaving the system and the other that signals to the attendant that the car wash is now idle. Observe the similarity with object-oriented styles of programming in which objects communicate solely by method invocations.

Given that it takes **cw1** 8 minutes and **cw2** 10 minutes to wash a car, a **possible** sequence of events is . . .



## Sequence of events

message	event	time	sender	receiver	content
1	-	0	cw1	att	idle
2	-	0	cw2	att	idle
3	1	6	source	att	car 1
4	2	6	att	cw1	car 1
5	3	11	source	att	car 2
6	4	11	att	cw2	car 2
7	5	12	source	att	car 3
8	6	14	cw1	sink	car 1
9	-	14	cw1	att	idle
10	7	14	att	cw1	car 3
11	8	17	source	att	car 4
12	9	19	source	att	car 5
13	10	21	cw2	sink	car 2
14	-	21	cw2	att	idle

## Simulating a single server queue

As a more detailed example consider the simulation of a single server queue to which customers arrive according to a (homogeneous) Poisson process of rate  $\lambda$ .

On arrival a customer either enters service immediately (if the server is free) or waits in a queue of customers (if the server is busy). When a customer departs the server the customer who has been waiting longest (assuming FIFO discipline) enters service or, if the queue is empty, the server remains idle until the next customer arrives.

The times taken to serve each customer are independent, identically distributed random variables with probability distribution function  $G(\cdot)$ .

## Simulating a single server queue (2)

Let  $T$  be a fixed time beyond which customers are no longer allowed to enter the system.

Beyond time  $T$  the server continues to serve all remaining customers until the system is empty.

We use the simulation to estimate

- ▶ the average time a customer spends in the system;

## Simulating a single server queue (3)

Variable	Description
Time	$t$
Counter	$N_A$ , the No of arrivals $N_D$ , the No of departures
System state	$n$ , the No of customers

The event list will consist of two elements:  $t_A$ ,  $t_D$ , the times of the next arrival and next departure, respectively. If there is no customer in service then put  $t_D = \infty$ .

We will output  $A(i)$  and  $D(i)$  the times of arrival and departure of customer  $i$  together with  $T_f$  the time past  $T$  that the last customer departs the system.

## Simulating a single server queue (4)

Within the simulation we use the random variable  $T_s$  as the time of the next arrival after time  $s$  and  $Y$  as a random service time chosen from the given distribution  $G(\cdot)$ .

The initialization of the variables is as follows.

- ▶ Set  $t = N_A = N_D = 0$ ;
- ▶ Set  $n = 0$ ;
- ▶ Generate  $T_0$  and set  $t_A = T_0$  and  $t_D = \infty$ .

The simulation advances event by event updating the variables according to one of the following cases.

## Simulating a single server queue (5)

Case I:  $t_A \leq t_D$ ,  $t_A \leq T$

- ▶ Reset  $t = t_A$ ,  $N_A = N_A + 1$ ,  $n = n + 1$ ;
- ▶ Generate  $T_t$ , reset  $t_A = T_t$ ;
- ▶ If  $n = 1$ , generate  $Y$  and reset  $t_D = t + Y$ ;
- ▶ Collect output data  $A(N_A) = t$ .

Case II:  $t_D \leq t_A$ ,  $t_D \leq T$

- ▶ Reset  $t = t_D$ ,  $n = n - 1$ ,  $N_D = N_D + 1$ ;
- ▶ If  $n = 0$  reset  $t_D = \infty$  else generate  $Y$  and reset  $t_D = t + Y$ ;
- ▶ Collect output data  $D(N_D) = t$ .

## Simulating a single server queue (6)

Case III:  $\min(t_A, t_D) > T, n > 0$

- ▶ Reset  $t = t_D, n = n - 1, N_D = N_D + 1$ ;
- ▶ If  $n > 0$  generate  $Y$  and reset  $t_D = t + Y$ ;
- ▶ Collect output data  $D(N_D) = t$ .

Case IV:  $\min(t_A, t_D) > T, n = 0$

- ▶ Stop.

Each simulation run will produce the quantities  $D(i) - A(i)$  giving the amount of time that the  $i^{\text{th}}$  customer spent in the system.

We then use the numerical average value of these quantities as our estimate of the mean time that a customer spends in the system.

Recall that this result depends on the use of the **weak law of large numbers**.

## Simulation performance measures

As the simulation itself is stochastic, so too are the observed outputs. It is critical to realize that a simulation can only yield **estimates** for performance measures and there will **always** be some statistical error in the estimates.

We can attempt to reduce the error by

- ▶ running the simulation for longer until sufficient samples have been taken;
- ▶ running the same simulation with a different pseudo-random number sequence, and combining the results from multiple runs. We will say more on this later.



# Performance metrics

- ▶ **Utilization**: The utilization is the proportion of time that a server is busy.  
An estimate can therefore be obtained by taking the sum of the busy times of the server and dividing by  $T$ , the simulation length. In the case of a  $k$ -server, the busy times can be estimated together and divided by  $kT$ .
- ▶ **Throughput**: mean number of customers receiving service per unit time
- ▶ **Mean queueing time**
- ▶ **Mean waiting time**

## Statistical analysis of results

Suppose that  $X_1, X_2, \dots, X_n$  are independent, identically distributed random variables and let  $\mu$  and  $\sigma^2$  be their common mean and variance, respectively.

Hence,

$$\mathbb{E}(X_i) = \mu$$

and

$$\text{Var}(X_i) = \mathbb{E}((X_i - \mathbb{E}(X_i))^2) = \sigma^2.$$

Given a sample of values of the random variables  $X_1, X_2, \dots, X_n$  how might we **estimate** the values of the true but unknown parameters  $\mu$  and  $\sigma^2$  and how does the sample size,  $n$ , effect the accuracy of our estimates?

# Sample mean

The quantity

$$\bar{X} := \frac{1}{n} \sum_{i=1}^n X_i$$

is called the **sample mean**. Note that the sample mean is a random variable and its mean is given by

$$\begin{aligned} \mathbb{E}(\bar{X}) &= \mathbb{E}\left(\sum_{i=1}^n \frac{X_i}{n}\right) \\ &= \sum_{i=1}^n \frac{\mathbb{E}(X_i)}{n} \\ &= \frac{n\mu}{n} \\ &= \mu. \end{aligned}$$

## Sample mean (2)

Similarly, the variance of  $\bar{X}$  is given by

$$\begin{aligned}\text{Var}(\bar{X}) &= \mathbb{E}((\bar{X} - \mu)^2) \\ &= \text{Var}\left(\sum_{i=1}^n \frac{X_i}{n}\right) \\ &= \frac{1}{n^2} \sum_{i=1}^n \text{Var}(X_i) \\ &= \frac{\sigma^2}{n}.\end{aligned}$$

Thus,  $\bar{X}$  can be used to estimate  $\mu$  from the sample. It is a good estimator of  $\mu$  when  $\sigma/\sqrt{n}$  is small.

We can view  $\text{Var}(\bar{X})$  as the **mean squared error** in our estimation of  $\mu$  by  $\bar{X}$ .

## Sample variance

Call the random variable

$$S^2 := \frac{\sum_{i=1}^n (X_i - \bar{X})^2}{n-1}$$

the **sample variance**.

It is possible to show that

$$\mathbb{E}(S^2) = \sigma^2$$

and so  $S$  is a suitable estimator for the true standard deviation  $\sqrt{\text{Var}(X_i)} = \sigma$ .

It is important to have an estimator for  $\sigma$  since we have seen that the accuracy of our estimator ( $\bar{X}$ ) for the true mean  $\mu$  depends on the variance as well as on the sample size,  $n$ .

# Confidence intervals

We can use the **Central Limit Theorem** to see that for large sample sizes,  $n$ , the random variable

$$\frac{(\bar{X} - \mu)}{\sigma/\sqrt{n}} = \sqrt{n} \frac{(\bar{X} - \mu)}{\sigma}$$

is approximately distributed  $N(0, 1)$ .

Additionally, we usually do not know the true variance  $\sigma^2$  but instead need to estimate it by  $S^2$  so that

$$\sqrt{n} \frac{(\bar{X} - \mu)}{S}$$

is approximately distributed  $N(0, 1)$ .

## Confidence intervals (2)

Write  $z_\alpha$  for the value such that  $\mathbb{P}(Z > z_\alpha) = \alpha$  where  $Z$  is a standard normal random variable  $N(0, 1)$  then it follows that

$$\mathbb{P}(-z_{\alpha/2} < Z < z_{\alpha/2}) = 1 - \alpha$$

and so by the CLT for large  $n$

$$\mathbb{P}\left(-z_{\alpha/2} < \sqrt{n} \frac{(\bar{X} - \mu)}{S} < z_{\alpha/2}\right) \approx 1 - \alpha$$

or, equivalently,

$$\mathbb{P}\left(\bar{X} - z_{\alpha/2} \frac{S}{\sqrt{n}} < \mu < \bar{X} + z_{\alpha/2} \frac{S}{\sqrt{n}}\right) \approx 1 - \alpha.$$

The interval  $\bar{X} \pm z_{\alpha/2} S / \sqrt{n}$  is an (approximate)  $100(1 - \alpha)$  percent confidence interval for  $\mu$ .

## Student's $t$ -distribution

If it is known that the common distribution of the variables  $X_i$  is also Normal then

$$\sqrt{n} \frac{(\bar{X} - \mu)}{S}$$

has (exactly) a distribution called the **Student's  $t$ -distribution** with  $n - 1$  degrees of freedom.

Thus, in this case, an alternative confidence interval for  $\mu$  is to take  $t_\alpha$  in place of  $z_\alpha$  where  $t_\alpha$  is defined analogously as

$$\mathbb{P}(T > t_\alpha) = \alpha$$

where  $T$  is a random variable with the Student  $t$ -distribution with  $n - 1$  degrees of freedom.

Values of  $z_\alpha$  and  $t_\alpha$  are readily available in statistical tables or in suitable libraries or packages.



## Stopping rules

How do we know when a system has been run 'long enough' for performance measures to be accurate?

We can repeat the simulation several times with different random seed values to obtain many samples. These multiple runs are called **replications**.

Having repeated the experiment  $n$  times, we can construct a confidence interval on our measure  $L$ , say.

Although large numbers of replications reduce the variance, each replication requires re-stabilizing the simulation.

Can we avoid this?

We may be able to use a single, long run, and break up our samples into  $n$  blocks, each of these can form a sample  $L_j$ .

## Stopping rules (2)

What could go wrong with this?

Correlation between successive blocks could mean that we have biased samples.

If the block size is large then the correlation should be small.

Explicit techniques exist to estimate the correlation and obtain the block size.

The simulation can be stopped once the estimate of  $L$  becomes stable. For example, once the confidence interval around  $L$  becomes sufficiently narrow.

## Stopping rules (3)

Suppose we wish to run the simulation until our  $100(1 - \alpha)$  confidence interval for the true mean value is of at most of width  $\ell$ , say.

We can guarantee this by means of the following algorithm

- ▶ Generate at least 100 data values, say;
- ▶ Repeatedly, generate additional data values, stopping when the number of values generated,  $n$ , is such that  $2z_{\alpha/2}S/\sqrt{n} < \ell$

The initial 100 data values is for illustration, a suitable value will depend on the precise simulation experiment under consideration. The intention is to suppress the effects of the initial transient phase.

## Goodness of fit tests

How might we **validate** our simulation model?

One area of concern is the assumption of various probabilistic distributions in the model. For example, how sure are we in the use of a Poisson distribution for the numbers of events in a given interval or the choice of the distribution  $G(\cdot)$  for the service times of customers in a queue.

Statistical procedures have been developed to help with these questions. The hypothesis of a particular distribution can be tested by observing the system and then asking whether the assumption of a particular distribution is 'consistent' with the data. Such procedures are known as statistical **goodness of fit** tests.

## Chi-Squared test for discrete data

Suppose we have  $n$  independent random variables  $Y_1, Y_2, \dots, Y_n$  and we wish to test the **null hypothesis** that they have the distribution

$$\mathbb{P}(Y_j = i) = p_i \quad i = 1, 2, \dots, k.$$

If we let  $N_i$  be the number of  $Y_j$  equal to  $i$  then we expect under the null hypothesis that

$$\mathbb{E}(N_i) = np_i \quad i = 1, 2, \dots, k.$$

Thus we should consider rejecting the null hypothesis when

$$T := \sum_{i=1}^k \frac{(N_i - np_i)^2}{np_i}$$

is large. How large is too large?

## Chi-Squared test (2)

It can be shown that under the null hypothesis and when  $n$  is large that the distribution of  $T$  is approximately a **chi-squared** random variable with  $k - 1$  degrees of freedom.

Thus, we can assess the value of

$$\mathbb{P}(T > t_{\text{obs}})$$

where  $t_{\text{obs}}$  is the observed value from standard tables of the chi-squared distribution.

Typically, we would reject the null hypothesis when

$$\mathbb{P}(T > t_{\text{obs}})$$

has a value less than 0.05 or, more conservatively, as low as 0.01. Otherwise, we say that the observed data appears consistent with the null hypothesis.

## Kolmogorov-Smirnov test for continuous data

Suppose now that we wish to test whether  $n$  independent random variables  $Y_1, Y_2, \dots, Y_n$  arise from a common continuous distribution  $F(x)$ .

First we observe the  $n$  random variables and construct the **empirical distribution** defined by

$$F_e(x) := \frac{\text{No of } i \text{ such that } Y_i \leq x}{n}.$$

This will measure the proportion of observed values less than or equal to  $x$  and so should be 'close' to the function  $F(x)$  under the null hypothesis.

## Kolmogorov-Smirnov test for continuous data (2)

Consequently, we would expect the quantity

$$D = \max_x |F_e(x) - F(x)|$$

to be small and we should reject the null hypothesis if  $D$  is too large. The quantity  $D$  is called the **Kolmogorov-Smirnov test statistic**. The distribution will depend on the sample size  $n$  and has been tabulated.



## Other tests for randomness

Various statistical tests are available.

**Runs tests**, which examine the arrangement of numbers in a sequence (a run) to test the hypothesis of independence.

These tests frequently check for the number of “up runs”, the number of “down runs”, and the runs above and below the mean.

**Autocorrelation tests** check the correlation structure of the sequence of observations.

## Variance reduction techniques

So far we have used the sample mean,  $\bar{X}$ , as our estimator for  $\mu$ , the mean value of our distribution. We know that

$$\mathbb{E}(\bar{X}) = \mu \quad \text{and} \quad \text{Var}(\bar{X}) = \frac{\sigma^2}{n}$$

where  $n$  is the sample size.

Might we be able to find an alternative estimator for  $\mu$  which has smaller variance?

Such **variance reduction techniques** can sometimes produce significant speed-ups in the simulation.

## Antithetic variables

Suppose that  $X_1$  and  $X_2$  are two identically distributed random variables with common mean  $\mathbb{E}(X_1) = \mathbb{E}(X_2) = \mu$ . Then

$$\text{Var}\left(\frac{X_1 + X_2}{2}\right) = \frac{1}{4}(\text{Var}(X_1) + \text{Var}(X_2) + 2\text{Cov}(X_1, X_2)).$$

Hence, we would get a reduced variance by using  $(X_1 + X_2)/2$  when  $X_1$  and  $X_2$  are **negatively correlated**.

Recall that in the inverse transform method we generate (pseudo) random numbers by first generating pseudo random numbers  $U_1, U_2, \dots$  with a  $U(0, 1)$  distribution. But then the numbers  $1 - U_1, 1 - U_2, \dots$  are also random numbers with a  $U(0, 1)$  distribution and these two series of numbers are negatively correlated.

## Antithetic variables (2)

It often happens in practice that dividing the simulation runs into two groups and using  $1 - U$  for the second group in place of  $U$  in the first group yields two random variables  $X_1$  and  $X_2$  which are negatively correlated.

In this case we say that  $X_1$  and  $X_2$  are **antithetic variables**.

## Example

Consider a queueing system and let  $D_i$  be the delay in the queue of the  $i$ th customer and suppose we wish to estimate

$$\mathbb{E}(D_1 + D_2 + \cdots + D_n)$$

the sum of the delays of the first  $n$  customers.

We should expect to require a collection of  $2n$  random variables  $U_j$  (one for each arrival and departure event per customer).

Repeating the simulation using the  $2n$  random numbers given by  $1 - U_j$  will then give an improved estimator compared to using a 'fresh' set of  $2n$  random numbers.

## Control variates

Suppose we run a simulation and gather from the output a random variable  $X$  for estimating  $\mu$  with  $\mathbb{E}(X) = \mu$ .

Now, suppose that we also gather another random variable,  $Y$ , from the same output with a known mean value  $\mathbb{E}(Y) = \mu_Y$ .

Hence, for any number  $c$

$$Z = X + c(Y - \mu_Y)$$

is also an estimator for  $\mu$  since clearly  $\mathbb{E}(Z) = \mu$ .

What is the best choice of  $c$ ?

## Control variates (2)

Note that

$$\begin{aligned}\text{Var}(Z) &= \text{Var}(X + c(Y - \mu_Y)) = \text{Var}(X + cY) \\ &= \text{Var}(X) + c^2\text{Var}(Y) + 2c\text{Cov}(X, Y)\end{aligned}$$

and so, using calculus, the variance is minimized by taking  $c = c^*$  where

$$c^* = -\frac{\text{Cov}(X, Y)}{\text{Var}(Y)}$$

and then

$$\text{Var}(X + c^*(Y - \mu_Y)) = \text{Var}(X) - \frac{(\text{Cov}(X, Y))^2}{\text{Var}(Y)} \leq \text{Var}(X).$$

The variable,  $Y$ , is called the **control variate** for  $X$ .

## Example

Suppose we are simulating a queueing system and we are interested in estimating the total time spent in the system by all customers arriving before time  $t$ . If  $W_i$  is the amount of time spent in the system by the  $i$ th customer then we wish to estimate  $\mu = \mathbb{E}(X)$  where

$$X = \sum_{i=1}^{N(t)} W_i$$

where  $N(t)$  is the number of arrivals by time  $t$ .



## Example (2)

If the service time of the  $i$ th customer is  $S_i$  then

$$Y = \sum_{i=1}^{N(t)} S_i$$

can act as a control variate and we have from known quantities that

$$\mathbb{E}(Y) = \mu_Y = \mathbb{E}(S)\mathbb{E}(N(t)).$$

Note that in order to compute the optimal choice  $c^*$  for  $c$  we would have to estimate the variance  $\text{Var}(Y)$  and covariance  $\text{Cov}(X, Y)$  from the simulated data.

# Queueing theory

# Stochastic processes

A **stochastic process** is a collection of random variables  $X(t)$  taking values in a state space  $S$  indexed by times in a set  $T$ .

The values  $X(t)$  are said to denote the state of the process at time  $t$ .

An observed set of values  $X(t)$  for  $t \in T$  is said to be a **sample path** or **realization** of the process.

## Discrete and continuous

A stochastic process is a **discrete-state** process when  $S$  is countable, e.g integer-valued. Otherwise, the process is **continuous-state**.

A stochastic process is a **discrete-time** process when  $T$  is countable, e.g  $T = \{0, 1, 2, \dots\}$ . Otherwise, the process is **continuous-time**.

Markov chains are discrete-state, discrete-time stochastic processes.

# Markov processes

A **Markov process** is a stochastic process such that for all  $t_1 < \dots < t_n < t$  and for events  $A_1, \dots, A_n, A$

$$\mathbb{P}(X(t) \in A | X(t_1) \in A_1, \dots, X(t_n) \in A_n) = \mathbb{P}(X(t) \in A | X(t_n) \in A_n)$$

This is known as the **Markov property** and it says that the choice of next state depends only on the current state (and not on earlier states).

Thus, more precisely, Markov chains are discrete-state, discrete-time Markov processes. In this case, we usually denote the time values,  $T$ , as  $T = \{0, 1, 2, \dots\}$  and also use the notation  $X_t$  for  $X(t)$ ,  $t \in T = \{0, 1, 2, \dots\}$ .

## Birth death processes

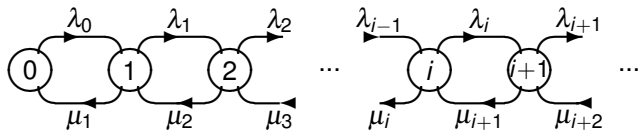
A **birth death process** is a discrete-state continuous-time Markov process with states given by the non-negative integers and for which we allow transitions only between neighbouring states.

A jump up by one is said to be a **birth** event and a jump down by one a **death** event.

The Markov property implies that if the process is observed in some state  $i$  then the time of the next jump is independent of how long it has already spent in state  $i$ . This is the memoryless property and so the time spent in a state is exponentially distributed.

## State space diagram

We use  $\lambda_i$  to represent the instantaneous birth rate in state  $i$  and  $\mu_i$  to represent the instantaneous death rate in state  $i$ .



## Time dependent solution of BDP

We denote by  $P_i(t)$  the probability of being in state  $i$  at time  $t$ .

The probability of a birth in an interval  $\Delta t$  when the system starts in state  $i$  is assumed to be  $\lambda_i \Delta t + o(\Delta t)$ .

The probability of a death in  $\Delta t$  when the system starts in state  $i$  is  $\mu_i \Delta t + o(\Delta t)$ .

The probability of more than one event in  $\Delta t$  is  $o(\Delta t)$ .

[Recall:  $o(\Delta t)$  denotes a quantity which becomes negligible when compared with  $\Delta t$  as  $\Delta t \rightarrow 0$ .]

To solve for  $P_i(t)$  we write a set of differential equations called the **Chapman Kolmogorov** equations.



# Chapman Kolmogorov equations

For  $i \geq 1$ :

$$\begin{aligned}P_i(t + \Delta t) &= P_i(t)(1 - \lambda_i \Delta t)(1 - \mu_i \Delta t) \\ &\quad + P_{i+1}(t)(\mu_{i+1} \Delta t)(1 - \lambda_{i+1} \Delta t) \\ &\quad + P_{i-1}(t)(\lambda_{i-1} \Delta t)(1 - \mu_{i-1} \Delta t) \\ &\quad + o(\Delta t).\end{aligned}$$

For  $i = 0$ :

$$\begin{aligned}P_0(t + \Delta t) &= P_0(t)(1 - \lambda_0 \Delta t) \\ &\quad + P_1(t)(\mu_1 \Delta t)(1 - \lambda_1 \Delta t) \\ &\quad + o(\Delta t).\end{aligned}$$

## Chapman Kolmogorov (2)

We derive differential equations by dividing through by  $\Delta t$  and taking the limit as  $\Delta t \rightarrow 0$  to get for  $i \geq 1$

$$\begin{aligned}\frac{dP_i(t)}{dt} &= -(\lambda_i + \mu_i)P_i(t) \\ &\quad + \mu_{i+1}P_{i+1}(t) \\ &\quad + \lambda_{i-1}P_{i-1}(t)\end{aligned}$$

and for  $i = 0$

$$\frac{dP_0(t)}{dt} = -\lambda_0 P_0(t) + \mu_1 P_1(t).$$

The time dependent solution is **difficult** for many systems of interest, so we will study the stationary solution.

## Stationary solution

We are interested in the long term probabilities after the system has reached an **equilibrium**.

These probabilities are independent of the initial conditions.

System reaches equilibrium if, for all  $i$ ,

$$\lim_{t \rightarrow \infty} P_i(t) = p_i \quad \text{exists.}$$

## Stationary solution (2)

The quantities  $p_i$  solve the Chapman Kolmogorov equations with

$$\frac{dP_i(t)}{dt} = 0$$

so that

$$0 = -(\lambda_i + \mu_i)p_i + \mu_{i+1}p_{i+1} + \lambda_{i-1}p_{i-1}$$

$$0 = -\lambda_0 p_0 + \mu_1 p_1.$$

Rewriting gives

$$p_{i+1} = \frac{\lambda_i + \mu_i}{\mu_{i+1}} p_i - \frac{\lambda_{i-1}}{\mu_{i+1}} p_{i-1} \quad (i \geq 1)$$

$$p_1 = \frac{\lambda_0}{\mu_1} p_0.$$

## Stochastic balance

Under steady-state conditions we require total flow into a state to equal total flow out of a state.

The total flow is the product of the steady state probabilities and the flow rates.

We enter state  $i$  at rate  $p_{i-1}\lambda_{i-1} + p_{i+1}\mu_{i+1}$ .

We exit state  $i$  at rate  $p_i\lambda_i + p_i\mu_i$ .

The equation

$$p_{i-1}\lambda_{i-1} + p_{i+1}\mu_{i+1} = p_i\lambda_i + p_i\mu_i \quad (i \geq 1)$$

equates the flow into and out of state  $i$ .

This is called the **global balance** equation.

## Stochastic balance (2)

The equations

$$p_i \lambda_i = p_{i+1} \mu_{i+1} \quad (i \geq 0)$$

describing flow from state  $i$  to state  $i + 1$  are the **detailed balance** equations.

Rewriting gives

$$p_{i+1} = \frac{\lambda_i}{\mu_{i+1}} p_i$$

which gives us the product solution

$$p_k = p_0 \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}} \quad \text{for } k \geq 1$$

for  $p_k$  ( $k \geq 1$ ) in terms of  $p_0$ .

## Stochastic balance (3)

Since the sum of state probabilities is unity,

$$\begin{aligned} \rho_0 + \sum_{k=1}^{\infty} \rho_k &= 1 \\ \rho_0 + \sum_{k=1}^{\infty} \rho_0 \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}} &= 1 \\ \rho_0 \left[ 1 + \sum_{k=1}^{\infty} \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}} \right] &= 1 \end{aligned}$$

so that

$$\begin{aligned} \rho_0 &= \left[ 1 + \sum_{k=1}^{\infty} \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}} \right]^{-1} \\ \rho_k &= \rho_0 \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}}. \end{aligned}$$

## The $M/M/1$ queue

The birth death process maps well onto queueing systems. Births represent arrivals to queue, deaths represent departures as customers finish service.

The  $M/M/1$  queue is an infinite customer system, with infinite waiting room, and a state independent service rate.

This means that  $\lambda_i = \lambda$  and  $\mu_i = \mu$  for all  $i$  and we can solve the detailed balance equations as

$$\begin{aligned} p_k &= p_0 \prod_{i=0}^{k-1} \frac{\lambda}{\mu} \\ &= p_0 \left( \frac{\lambda}{\mu} \right)^k . \end{aligned}$$



## The $M/M/1$ queue (2)

Writing  $\rho = \frac{\lambda}{\mu}$  and for  $\rho < 1$  then

$$\begin{aligned} p_0 &= \frac{1}{1 + \sum_{k=1}^{\infty} \rho^k} \\ &= \frac{1}{1 + \rho \sum_{k=0}^{\infty} \rho^k} \\ &= \frac{1}{1 + \rho \left( \frac{1}{1-\rho} \right)} \\ &= 1 - \rho \end{aligned}$$

Consequently, the distribution of the number in the system,  $N$ , is

$$p_k = (1 - \rho)\rho^k, \quad k = 0, 1, 2, \dots$$

If  $\rho \geq 1$ , that is, if  $\lambda \geq \mu$  the system will not reach equilibrium.

## The $M/M/1$ queue (3)

What is the average number of customers,  $\mathbb{E}(N)$ , in the system?

$$\begin{aligned}\mathbb{E}(N) &= \sum_{k=0}^{\infty} k p_k \\ &= \sum_{k=0}^{\infty} k(1-\rho)\rho^k \\ &= (1-\rho)\rho \frac{\partial}{\partial \rho} \left( \sum_{k=0}^{\infty} \rho^k \right) \\ &= (1-\rho)\rho \frac{\partial}{\partial \rho} \left( \frac{1}{1-\rho} \right) \\ &= \frac{\rho}{1-\rho}.\end{aligned}$$

## The $M/M/1$ queue (4)

An arriving customer will find, on average  $\mathbb{E}(N)$  in the system, and will spend a time, say  $\mathbb{E}(T)$ , in the system. During  $\mathbb{E}(T)$  there will be, on average  $\lambda \mathbb{E}(T)$  arrivals, leaving  $\mathbb{E}(N)$  customers in the queue. Thus

$$\mathbb{E}(N) = \lambda \mathbb{E}(T)$$

which is Little's result restated. In our case

$$\begin{aligned}\mathbb{E}(T) &= \frac{\mathbb{E}(N)}{\lambda} = \frac{\rho}{\lambda(1-\rho)} \\ &= \frac{1}{\mu(1-\rho)} = \frac{1}{\mu - \lambda}\end{aligned}$$

which is the  $M/M/1$  average residence time.

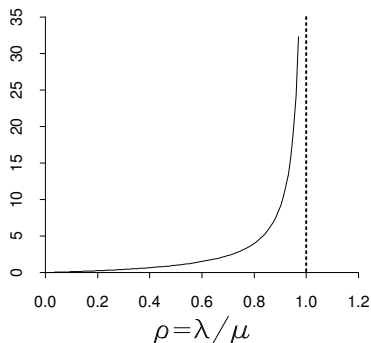
Note that

- ▶  $\frac{1}{\mu}$  is the average service time;
- ▶  $\rho$  is the utilization.

## Performance at high load

At high utilizations  $\rho$  approaches one and the residence time and queue lengths are unbounded.

Expected number in the system



**Exercise:** what happens to the variance as  $\rho \rightarrow 1$ ?

## Residence time

Consider the effect on residence time by increasing the utilization by a constant load of 0.1.

utilization ( $\rho$ )			residence time ( $\mathbb{E}(T)$ )		
old	new	% $\uparrow$	old	new	% $\uparrow$
0.1	0.2	100.0	$1.11 \frac{1}{\mu}$	$1.25 \frac{1}{\mu}$	13
0.5	0.6	20.0	$2 \frac{1}{\mu}$	$2.5 \frac{1}{\mu}$	25
0.8	0.9	12.5	$5 \frac{1}{\mu}$	$10 \frac{1}{\mu}$	100

Predicting residence times is very difficult at high loads. Running systems at maximum utilization may please the **providers**, but it doesn't please the **users**.

## $M/M/1$ — an example

H.R. Cutt barber shop, no appointment needed!

Customers served FCFS.

On Saturday mornings he is very busy and is considering hiring a helper.

Measures the (Poisson) arrival rate of customers to be 5 per hour.

Customers are prepared to wait, and he spends on average 10 min per cut.

What are the average number of customers in the shop, the average number of customers waiting? What percentage of time can a customer receive a cut without waiting?

He has 4 seats in his waiting room. What is the probability that an arriving customer will find no seat and have to stand?

## M/M/1 example solution

The average number of customers in the shop:  $\lambda = 5$  per hour,  $\mu = 6$  per hour So

$$\rho = 5/6, \quad \text{and hence } \mathbb{E}(N) = 5.$$

Since the average number of customers in service is  $\rho$ , the utilization, the average number of customers waiting is

$$\mathbb{E}(N_q) = \mathbb{E}(N) - \rho = 4\frac{1}{6}.$$

How likely is the barber to be idle?

$$\rho_0 = 1 - \rho = \frac{1}{6}.$$

How often is no seat free?

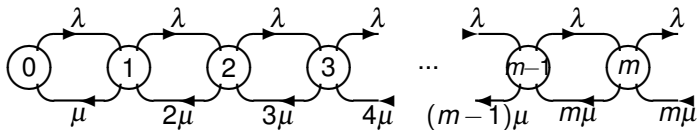
$$\begin{aligned} \mathbb{P}(\text{no seat}) &= \mathbb{P}(N \geq 5) \\ &= \rho^5 \\ &\approx 0.402 \end{aligned}$$

## $M/M/m$ — $m$ servers

This is just like the  $M/M/1$  system, except that there are  $m$  servers. For all  $k$ ,  $\lambda_k = \lambda$ , but now the service rate is a function of the number of customers in the system

$$\mu_k = \begin{cases} k\mu & \text{if } 0 < k \leq m \\ m\mu & \text{if } k \geq m \end{cases}$$

For an equilibrium distribution to exist we require that  $\frac{\lambda}{m\mu} < 1$ .





## $M/M/1/K$ — finite capacity

The system can hold up to  $K$  customers.

Now for  $k \geq K$ ,  $\lambda_k = 0$  and for  $k > K$ ,  $\rho_k = 0$ .

Using the equations from the  $M/M/1$  queueing system, but limiting the summation, and again writing  $\rho = \frac{\lambda}{\mu}$ ,

$$\rho_k = \rho_0 \rho^k \quad \text{for } k \leq K$$

$$\begin{aligned} \rho_0 &= \frac{1}{1 + \sum_{k=1}^K \rho^k} \\ &= \begin{cases} \frac{1}{1 + \frac{\rho - \rho^{K+1}}{1 - \rho}} & (\rho \neq 1) \\ \frac{1}{1+K} & (\rho = 1) \end{cases} \\ &= \begin{cases} \frac{1 - \rho}{1 - \rho^{K+1}} & (\rho \neq 1) \\ \frac{1}{1+K} & (\rho = 1) \end{cases} \end{aligned}$$

For this system with a finite state space an equilibrium distribution always exists whatever the arrival and departure rates.

Note that  $\rho_0$  is greater than  $1 - \rho$  which is the equilibrium probability that the system is empty in the  $M/M/1$  case.

## $M/M/1//N$ — finite population

Single server, unbounded queue capacity and a population of  $N$  customers. We solve this system by modifying the  $\lambda_k$  to model the arrival rate. Instead of having an arrival rate for the population as a whole, we assign an arrival rate to each customer, say  $\lambda$ .

If there are no customers in the system, then all of them are eligible to be born, so that

$$\lambda_0 = N\lambda.$$

As we have more customers in the system, we have less eligible to be born. So that,

$$\lambda_k = (N - k)\lambda, \quad \text{for } 0 \leq k \leq N.$$

With a single server the service rate is constant

$$\mu_k = \mu, \quad \text{for } k \geq 1.$$

## $M/M/m/m$ — $m$ server loss system

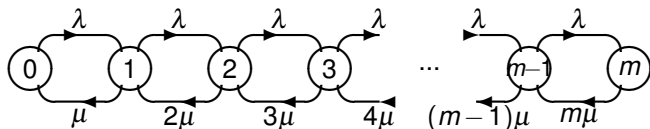
An application of this system is to model a link in a telephone network. Such a link contains  $m$  circuits each of which carries a single call. Suppose that calls arrive at the link according to a Poisson process of rate  $\lambda$ .

If there is a free circuit the call is connected and holds the circuit for an exponentially distributed length of time, with mean  $\frac{1}{\mu}$ .

At the end of this holding period the call terminates and the circuit is freed.

If there are no free circuits then the call is lost.

## M/M/m/m loss system



$$\lambda_k = \begin{cases} \lambda & k < m \\ 0 & k \geq m \end{cases}$$

$$\mu_k = k\mu \quad \text{for } 1 \leq k \leq m$$

The flow balance equations give for  $k \leq m$

$$\begin{aligned} p_k &= p_0 \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}} = p_0 \prod_{i=0}^{k-1} \frac{\lambda}{(i+1)\mu} \\ &= p_0 \left( \frac{\lambda}{\mu} \right)^k \frac{1}{k!} \end{aligned}$$

## M/M/m/m loss system (2)

Solving for  $p_0$  gives

$$\sum_{k=0}^m p_k = p_0 \sum_{k=0}^m \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!} = 1$$
$$\Rightarrow p_0 = \left[ \sum_{k=0}^m \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!} \right]^{-1}$$

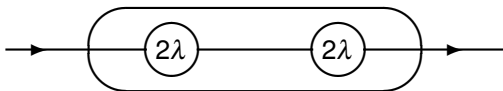
The probability that an arriving call finds all circuits occupied,  $p_m$ , is called the **loss probability** for the telephone link. Thus,

$$p_m = p_0 \left(\frac{\lambda}{\mu}\right)^m \frac{1}{m!}$$
$$= \left(\frac{\lambda}{\mu}\right)^m \frac{1}{m!} \left[ \sum_{k=0}^m \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!} \right]^{-1}$$

This expression for the loss probability is known as **Erlang's formula**.

## Extensions

First we relax our constraints on the arrival process distribution. We want to model systems in which the coefficient of variation of the interarrival time is less than one. Consider a system in which a birth occurs in two stages.



Each stage has an exponentially distributed residence time.

## Extensions (2)

If the desired birth rate is  $\lambda$  then let each stage have a rate  $2\lambda$ .  
The average time,  $\tau$ , to get through the combined birth process will be

$$\mathbb{E}(\tau) = \frac{1}{2\lambda} + \frac{1}{2\lambda} = \frac{1}{\lambda}.$$

Since each stage has exponentially distributed residence times, the variance of each stage is

$$\sigma_{\text{single}}^2 = \frac{1}{(2\lambda)^2}.$$

The two stages are independent, so the variance of  $\tau$ , the time to get through both stages is

$$\sigma_{\tau}^2 = \frac{1}{(2\lambda)^2} + \frac{1}{(2\lambda)^2} = \frac{1}{2\lambda^2}$$

## Extensions (3)

So the coefficient of variation is

$$C_{\tau} = \frac{\sqrt{\frac{1}{2\lambda^2}}}{\frac{1}{\lambda}} = \frac{1}{\sqrt{2}}.$$

In general, if we use  $r$  stages each with rate  $r\lambda$  we get an average time through all stages of  $\frac{1}{\lambda}$  and a coefficient of variation of  $\frac{1}{\sqrt{r}}$ .

The distribution that describes this  $r$ -stage process is the **Erlang distribution**, denoted  $E_r$ .

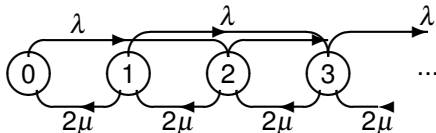


## Example, $M/E_2/1$

Allow the state of the process to represent the number of stages remaining to be served.

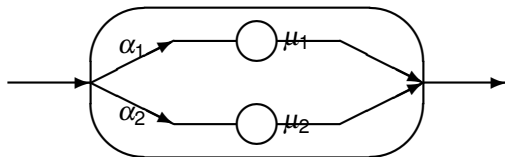
An arrival increases the number of stages remaining to be served by 2 and occurs at rate  $\lambda$ .

A departure from a stage reduces the number of stages to be served by 1 and occurs at rate  $2\mu$ .



## Parallel Servers

Combining stages in series reduces the coefficient of variation. If, instead, we combine them in parallel with a probability  $\alpha_i$  of choosing the  $i^{\text{th}}$  parallel stage we get a service distribution with coefficient of variation larger than 1.



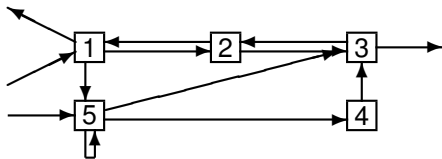
The coefficient of variation is given (see Kleinrock) by

$$C_\tau = \frac{2 \sum_{i=1}^r \frac{\alpha_i}{\mu_i^2}}{\left( \sum_{i=1}^r \frac{\alpha_i}{\mu_i} \right)^2} - 1 \geq 1.$$

# Queueing Networks

We have seen the solution of several queueing systems of interest. In general we want to solve a system of such queues representing a real world performance problem e.g. a distributed computing system. We represent the system under study as a network of connected queues.

Customers move (instantaneously) between service centres where they receive service.



## Model definition

- ▶ **customers**: typically these represent programs or data packets etc
- ▶ **service centres**: the resources in the system e.g. disks, CPU, transmission links
- ▶ **service time distributions**: may vary according to customer type and visit
- ▶ **load dependence**: multi-processor systems have load dependent service rates
- ▶ **waiting lines and scheduling**: may have limited capacity and various scheduling algorithms
- ▶ **customer types**: multiple customer classes may exist

# Open Queueing Networks

Customers arrive as a Poisson stream at node  $i$  at rate  $\gamma_i$ .

Customers may leave the network on completion of service.

Assume we have  $N$  nodes, each a single server queue with infinite waiting room.

Each server  $i$  has exponential service time with mean  $1/\mu_i$ .

A customer completing at node  $i$  moves to node  $j$  with probability  $q_{ij}$  for  $(i, j = 1, 2, \dots, N)$ .

Note that

$$\sum_{j=1}^N q_{ij} \leq 1.$$

## Open Queueing Networks (2)

A job leaves the network from node  $i$  with probability

$$q_{i0} = 1 - \sum_{j=1}^N q_{ij}.$$

The probabilities  $q_{ij}$  are called the **routing probabilities**.

Written as an  $N \times N$  matrix this is called the routing matrix  $Q = (q_{ij})$ .

An open network with parameters  $\gamma_i$ ,  $\mu_i$  and  $Q$  is called a **Jackson network**.

The system state is  $(k_1, k_2, \dots, k_N)$ , where  $k_i$  is the number of jobs present at node  $i$ .

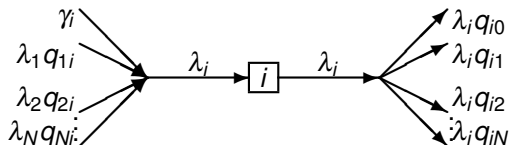
## Steady state solution

Let  $\lambda_j$  be the average number of arrivals to node  $j$  per unit time.

On average the departure count per unit time is therefore  $\lambda_j$ .

A fraction  $q_{ji}$  go to node  $i$ .

The rate of traffic from node  $j$  to node  $i$  is thus  $\lambda_j q_{ji}$ .



## Traffic equations

Adding together all contributions,

$$\lambda_i = \gamma_i + \sum_{j=1}^N \lambda_j q_{ji} \quad i = 1, 2, \dots, N.$$

These are known as the **traffic equations**.

A necessary and sufficient condition for the existence of an equilibrium distribution is that

$$\rho_i := \frac{\lambda_i}{\mu_i} < 1$$

where  $\lambda_i$  is the solution of the traffic equations.



# Jackson's Theorem

Let  $p(k_1, k_2, \dots, k_N)$  be the equilibrium distribution. **Jackson's Theorem** states that

$$p(k_1, k_2, \dots, k_N) = p_1(k_1)p_2(k_2) \cdots p_N(k_N)$$

where  $p_i(k_i)$  is the equilibrium distribution that there are  $k_i$  jobs in an  $M/M/1$  queue with traffic intensity  $\rho_i$ .

Jackson's theorem has some important implications.

- ▶ The numbers of jobs at the various nodes are independent.
- ▶ Node  $i$  behaves as if subjected to a Poisson stream with rate  $\lambda_i$ .

Jackson's theorem may be generalized to the case where node  $i$  has  $n_i$  servers and so the nodes behave as independent  $M/M/n_i$  queues.

## Closed Queueing Networks

Frequently used to model systems at high load or where a limited, constant number of jobs is admitted to service.

No external arrivals or departures.

Now the routing probabilities satisfy

$$\sum_{j=1}^N q_{ij} = 1, \quad i = 1, 2, \dots, N$$

The number of jobs in the system is always a constant, denoted by  $K$ . The states of the system, described by the vector  $(k_1, k_2, \dots, k_N)$ , thus satisfy the constraint,

$$\sum_{i=1}^N k_i = K.$$

## Closed Queueing Networks (2)

The state space,  $S$ , is then finite. The number of states is

$$\binom{K+N-1}{N-1}$$

The traffic equations become

$$\lambda_i = \sum_{j=1}^N \lambda_j q_{ji}, \quad i = 1, 2, \dots, N.$$

With a finite state space there always exists an equilibrium distribution.

Analogous to Jackson's theorem for the open network case it may be shown that

$$p(k_1, k_2, \dots, k_N) = \frac{1}{G} r_1(k_1) r_2(k_2) \cdots r_N(k_N)$$

where  $r_i(k_i)$  is the probability that there are  $k_i$  jobs in an  $M/M/1$  queue with traffic intensity given by a solution to the traffic equations.

# Open vs closed networks

The normalization constant  $G$  has to be determined by

$$G = \sum_{s \in S} r_1(k_1)r_2(k_2)\cdots r_N(k_N)$$

obtained by summing over all states  $s = (k_1, k_2, \dots, k_n)$  in the state space  $S$ .

With closed networks need to compute the normalization constant  $G$  — a computationally hard problem.

The constraint  $\sum_{i=1}^N k_i = K$  means that the numbers of jobs in the various queues are no longer independent. For instance, consider the extreme case where all  $K$  jobs are at one node.

## Size of state space, $S$

We require to show that

$$p(K, N) := \binom{K + N - 1}{N - 1}$$

is the number of (ordered) partitions of a positive integer  $K$  into  $N$  integer summands

$$K = \sum_{i=1}^N k_i.$$

**Proof:** consider  $K + N - 1$  boxes aligned in a row and select  $N - 1$  of these boxes (without replacement) which can be done in  $p(K, N)$  ways. Place a “/” symbol in each of the boxes and a “1” in each of the other boxes. The boxes now represent an (ordered) partition of  $K$  into  $N$  groups of “1” which when added together give the  $k_i$  summands.

## The $M/G/1$ queue

It is usually easier to justify the memoryless property for arrivals than for service times.

For arrivals, we can appeal to asymptotic results on the merging of large numbers of independent streams to help justify the memoryless property for arrivals.

For service times, it is easy to think of examples where the service times have a quite different distribution to the exponential. For example, the service times might be constant corresponding to certain packet lengths in a communication network.

This leads to an interest in the  $M/G/1$  queue with general service times given by a distribution function  $B(x) = \mathbb{P}(\text{service time} \leq x)$ , say.

## (Lack of) Markov property

With general service times we no longer find that  $X(t)$ , the number of customers in the system at time  $t$ , has the Markov property.

This follows since the future evolution of  $X(t)$  now depends not just on the number present but on the remaining service time of the customer (if any) currently in service.

Recall, that in the  $M/M/1$  case the remaining service time always has the same memoryless (that is, exponential) distribution whenever we observe the queue.

## Performance measures

The determination of a full description of the  $M/G/1$  model is possible but difficult. Instead, we shall look at some steady state performance measures.

Let  $1/\mu$  be the mean service time of a customer in the  $M/G/1$  queue, obtained from the distribution function of the service time distribution  $B(\cdot)$ , say. Then the mean queueing time,  $\mathbb{E}(T_q)$ , of a customer before it receives service is given by

$$\mathbb{E}(T_q) = \mathbb{E}(N_q) \frac{1}{\mu} + \rho \mathbb{E}(R)$$

where  $\mathbb{E}(N_q)$  is the average number of customers waiting in the queue at the time of arrival,  $\mathbb{E}(R)$  is the average remaining service time of the customer, if any, in service and  $\rho = \lambda/\mu$ , the traffic intensity, gives the utilization of the server.



## Remaining service time, $R$

A result from **renewal theory** is that  $\mathbb{E}(R) = \mu\mathbb{E}(S^2)/2$ .

Notice that this involves the 2nd moment,  $\mathbb{E}(S^2)$  of the service time  $S$ . For the exponential case,  $\mathbb{E}(S^2) = 2/\mu^2$  so that  $\mathbb{E}(R) = 1/\mu$  as might be intuitively expected (recall the Memoryless property).

## Performance measures (2)

From Little's law,

$$\mathbb{E}(N_q) = \lambda \mathbb{E}(T_q)$$

and so

$$\begin{aligned}\mathbb{E}(T_q) &= \lambda \mathbb{E}(T_q) \frac{1}{\mu} + \rho \mathbb{E}(R) \\ &= \frac{\rho \mathbb{E}(R)}{(1 - \rho)} \\ &= \frac{\rho \mu \mathbb{E}(S^2)}{2(1 - \rho)} \\ &= \frac{\lambda \mathbb{E}(S^2)}{2(1 - \rho)}.\end{aligned}$$

## Performance measures (3)

Let  $C_S$  be the coefficient of variation of the service time distribution then

$$C_S^2 = \frac{\mathbb{E}(S^2)}{(\mathbb{E}(S))^2} - 1$$

where  $\mathbb{E}(S) = 1/\mu$  so

$$\mathbb{E}(S^2) = \frac{(1 + C_S^2)}{\mu^2}$$

Hence,

$$\begin{aligned}\mathbb{E}(T_q) &= \frac{\lambda(1 + C_S^2)}{\mu^2 2(1 - \rho)} \\ &= \frac{\rho(1 + C_S^2)}{2\mu(1 - \rho)}.\end{aligned}$$

## Pollaczek-Khintchine formula

Consider now the total time,  $\mathbb{E}(T)$ , for a customer to on average pass through the system given by their waiting time in the queue and their own service time.

Thus,

$$\mathbb{E}(T) = \mathbb{E}(T_q) + \frac{1}{\mu} = \frac{1}{\mu} \left( 1 + \frac{\rho(1 + C_S^2)}{2(1 - \rho)} \right).$$

Using Little's law for the entire system we can now find,  $\mathbb{E}(N)$ , the mean number of customers in an  $M/G/1$  queueing system by

$$\begin{aligned} \mathbb{E}(N) &= \lambda \mathbb{E}(T) \\ &= \rho + \frac{\rho^2(1 + C_S^2)}{2(1 - \rho)} \end{aligned}$$

This is known as the **Pollaczek-Khintchine** (P-K) formula.

## Pollaczek-Khintchine formula (2)

The Pollaczek-Khintchine formula tells us that the mean number of customers is determined not only by the mean interarrival and mean service times but also by the **coefficient of variation** of the service time distribution,  $C_S$ .

There are several cases.

- ▶  $C_S = 0$ : this is the case of constant service times. For example, in ATM networks where the cells (that is, the packets) are of fixed length (53 bytes).
- ▶  $C_S < 1$ : this is the case where the variability is less than in the case of exponential service times, thus the  $M/M/1$  model will be conservative in its performance estimates.

## Pollaczek-Khintchine formula (3)

- ▶  $C_S \approx 1$ : this is where the  $M/M/1$  model works best and many systems correspond to this model. For example, batch jobs on a mainframe.
- ▶  $C_S > 1$ : this is the case where the  $M/G/1$  model is required. An example, is the observed packet lengths in Internet traffic. The distribution of packet sizes (and hence service times) is often found to be bimodal with many small packets and many longer packets of length determined by the MTU.