Computer Systems Modelling

Timothy L Harris

tim.harris@cl.cam.ac.uk

Eight lectures covering:

- *Introduction to modelling*, what is it, when is it useful?
- *Simulation techniques*, random number generation, analysis of results from simulation or measurements
- *Operational analysis*, performance bounds, balanced systems
- *Queueing theory*, Markov chains, single/multiple servers, bounded queues, queuing networks

Recommended books

- Jain, *The Art of Computer Systems Performance analysis*, Wiley, 1991

  — Simulation, random number generation, operational analysis, some basic queuing theory, substantial sections on designing and analysing experiments

- Leung, *Quantitative Analysis of Computer Systems*, Wiley, 1988

  — Operational analysis, basic queueing theory, small section on simulation

- Kleinrock, *Queueing Systems – Volume 1: Theory*, Wiley, 1975

  — Queueing theory, much more emphasis on mathematical derivations

Why model?

1. A manufacturer may have a range of compatible systems with different characteristics – which configuration would be best for a particular application?

2. A system is performing poorly – what should be done to improve it? Which problems should be tackled first?

3. Fundamental design decisions may affect performance of a system. A model can be used as part of the design process to avoid bad decisions.

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 system 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 CPU service?
- What is the expected utilization of the resource (CPU, disk, . . .)?
- How many requests will typically be queued ?
- What is the expected queue length at time t ?

Techniques

More than one way to skin a cat . . .

- Measurement – if the system already exists then maybe it can be changed and the effects analysed
- Simulation – construct a computer program that emulates the system under study
- Operational analysis – analysis based on directly measured quantities and relationships between them: makes few assumptions about the system
- Queueing theory – stochastic processes, queueing networks
- Back of the envelope – efficient but gives limited insight

Techniques (2)

Choosing between these techniques depends on ...

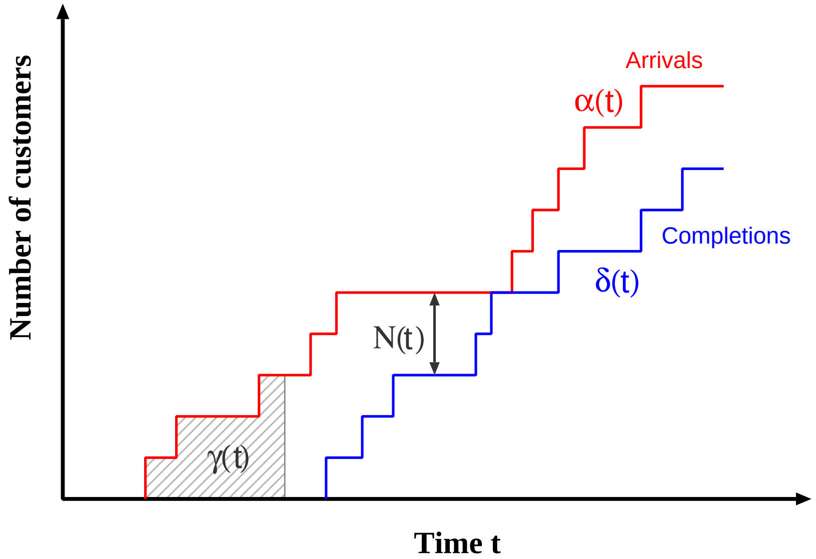
- Stage of development: can only measure an existing system
- Time available: measurements or simulations can take a long time to complete. How easily can different trade-offs be evaluated?
- Resources: systems with which to experiment, people with the relevant skills, cost
- Desired accuracy: do the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

Little's result

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

> *e.g.*: 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 $(0, t)$

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

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

Little's result (3)

Let

$$\lambda_t = \alpha(t)/t \quad (1)$$

$$T_t = \gamma(t)/\alpha(t) \quad (2)$$

$$\bar{N}_t = \gamma(t)/t \quad (3)$$

1. the average arrival rate during $(0, t)$
2. system time per customer averaged over all customers in $(0, t)$
3. average number of customers in system during $(0, t)$

Combining these:

$$\bar{N}_t = \lambda_t T_t$$

Little's result (4)

Assume the following limits exist as $t \to \infty$

$$\lambda = \lim_{t\to\infty} \lambda_t \quad (4)$$

$$T = \lim_{t\to\infty} T_t \quad (5)$$

If these exist then so does

$$\bar{N} = \lambda T \quad (6)$$

In English: the average number in system $=$ 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 Little in 1961.

Applications of Little's result

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

We can split T (average time in the system) into W (average time spent waiting) and $\bar{x}$ (average time spent being served).

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

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

$$\bar{N}_q = \lambda W$$

$$\bar{N}_s = \lambda \bar{x}$$

Probability theory refresher

We write the probability that an event will occur as $\mathbb{P}(\text{event})$

— e.g. $\mathbb{P}(X < 0.5)$ 'the random variable X is less than 0.5'

$\mathbb{P}(a, b)$ is the **joint** probability that *both* a and b will occur

— e.g. $\mathbb{P}(X < 0.5, Y < 0.5)$ 'both X and Y will be less than 0.5'

$\mathbb{P}(a|b)$ is the **conditional** probability that a will occur given that b has occured.

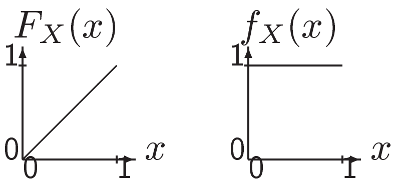
— e.g. $\mathbb{P}(X < 0.5|Y < 0.5)$ 'X will be less than 0.5 when we know that Y is less than 0.5'

$$\mathbb{P}(a|b) \triangleq \frac{\mathbb{P}(a, b)}{\mathbb{P}(b)}$$

Probability theory refresher (2)

We can describe the **distribution** from which a random variable is drawn using 2 functions: the *cumulative distribution function* (**CDF**) and the *probability density function* (**pdf**).

For example, for a random variable X uniformly distributed from 0 to 1, the CDF and pdf would be



$$F_X(x) \triangleq \mathbb{P}(X \leq x) \qquad f_X(x) \triangleq \frac{dF_X(x)}{dx}$$

Note that

$$F_X(x) = \int_{-\infty}^{x} f_X(y)dy\,, \quad \text{and} \quad \int_{-\infty}^{\infty} f_X(y)dy = 1.$$

Probability theory refresher (3)

The **expected value** of X, $\mathbb{E}(X)$ (sometimes called the average, or mean), is just

$$\mathbb{E}(X) \triangleq \overline{X} \triangleq \int_{-\infty}^{\infty} x f_X(x) dx$$

Also called the average, mean or **first moment** of the distribution

The n^{th} **moment** is defined as

$$\mathbb{E}(X^n) \triangleq \overline{X^n} \triangleq \int_{-\infty}^{\infty} x^n f_X(x) dx$$

When writing, be careful between $\overline{X^n}$ and $\overline{X}^n$

The n^{th} **central moment** is

$$\overline{(X - \overline{X})^n} = \int_{-\infty}^{\infty} (x - \overline{X})^n f_X(x) dx$$

Probability theory refresher (4)

The second central moment is known as the **variance**,

$$\sigma_X^2 \triangleq \overline{(X - \overline{X})^2} = \overline{X^2} - (\overline{X})^2$$

From this, we define:

$\sigma_X \triangleq \sqrt{\sigma_X^2}$ the **standard deviation**, the square root of the variance

$C_X \triangleq \dfrac{\sigma_X}{\overline{X}}$ the **coefficient of variation**, a dimensionless quantity

Numerically higher values signify 'more variable' data

Concretely, a coefficient of variation of 5 is considered large, and 0.2 is considered small

Probability theory refresher (5)

The previous examples have concerned **continuous** random variables whose distributions have been defined by their **CDF**

Similar definitions apply to the case of **discrete** random variables taking values x_i $(i \in I)$

$$\sum_{i \in I} P[X = x_i] = 1$$

The expected value of X is

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

Similarly, for the other moments, the integration becomes a summation over the set of possible values

Continuous distributions

The **uniform distribution** is used where a random variable takes values in some bounded range and there is no reason to favour one value over another

$$\begin{aligned}
a &= \text{lower limit} \\
b &= \text{upper limit}, b > a \\
f(x) &= \frac{1}{b-a} \\
F(x) &= \left| \begin{array}{ll} 0 & \text{if } x < a \\ \frac{x-a}{b-a} & \text{if } a \leq x < b \\ 1 & \text{if } b \leq x \end{array} \right.
\end{aligned}$$

The mean is $\frac{1}{2}(a+b)$, the variance is $\frac{1}{12}(b-a)^2$.

Continuous distributions (2)

The **exponential distribution** is the only continuous distribution with the **memoryless property** that

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

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 already waited

$$\begin{aligned} 1/\lambda &= \text{scale parameter} \\ f(x) &= \lambda\, e^{-\lambda\, x} \\ F(x) &= 1 - e^{-\lambda\, x} \end{aligned}$$

The mean is $\frac{1}{\lambda}$, the variance is $\frac{1}{\lambda^2}$. Consequently λ may be viewed as the mean event rate.

Discrete distributions

The **Bernoulli distribution** is used where a random variable can take only two values, usually either 1 representing the *success* or 0 representing the *failure* of some operation

$$\begin{aligned} p &= \text{probability of success}, 0 \le p \le 1 \\ f(x) &= \begin{cases} 1-p & \text{if } x = 0 \\ p & \text{if } x = 1 \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

The mean is evidently p and the variance $p(1-p)$.

Discrete distributions (2)

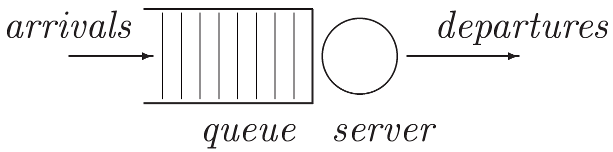
The number of successes in a series of Bernoulli trials has the **Binomial distribution**.

$$
\begin{aligned}
p &= \text{probability of success}, 0 \le p \le 1 \\
n &= \text{number of trials}, n \in \mathbb{N}^+ \\
f(x) &= \binom{n}{x} p^x (1-p)^{n-x}
\end{aligned}
$$

The mean is np and the variance is $np(1-p)$.

Note that $f(x)$ 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 will occur.

A simple queueing system



We characterise individual queueing systems by ...

- Arrival process $A(t) = \mathbb{P}(\text{ inter-arrival time} \le t)$
- Service process $B(x) = \mathbb{P}(\text{ service time} \le x)$
- Storage capacity available for waiting customers
- The number of servers/customers available
- The different kinds of arriving customers (big jobs, small jobs, ...)
- Queueing discipline used: FCFS, LCFS, SIRO, priority
- Defections, jockeying, 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 queue length
- l is the 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 systems notation (2)

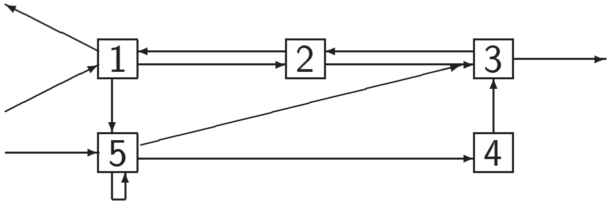
Typical values for A, B include:

- M – exponential distribution
- E_r – r stage Erlangian distribution
- H_r – r stage Hyperexponential 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

Queueing networks



More generally, we may wish to consider systems comprising multiple inter-connected service centres, forming a **queueing network**. We must specify:

- the properties of each node
  — e.g. using Kendall notation
- the way in which jobs move between the nodes
  — e.g. the links that exist between nodes and the way in which a job leaving one node selects between these links
- the workload being analyzed
  — e.g. that a disk-server workload comprises a $20:1$ mix of small and large requests

Queueing networks (2)

We can classify queueing networks as either

- **closed** systems in which a fixed set of jobs circulate between nodes, 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** systems in which jobs may enter and leave the system
  — e.g. the network on the previous slide: jobs may 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 node *at most* once

Simulation

Simulation allows arbitrarily complex systems to be evaluated

- Able to capture the dynamic behaviour of systems
- Captures the dynamics of complex systems by imitation
- Tracks the evolution of the system over time
- Examples include communication network design, road traffic modelling, studying chemical reactions, fluid flow, etc.

Simulation (2)

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

These closely follow the chronological order of events in the system being modelled

Consider simulation as a 'set of equations' describing evolution in time of system under study

The 'equations' are 'solved' by following their evolution in time

We **always** follow the **dynamic** evolution of the system, even if we only want a mean value – therefore, as well as techniques for implementing simulators, it is necessary to know how to analyse their results

Simulation is of particular use when studing systems which are not in a steady state condition

Types of simulation

Simulation studies may be classified as

- **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 in a vat

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

— e.g. a simulation of the number of students attending these lectures would be discrete time

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: the programmer may add arbitrary details.

There are four main problems with simulation:

- The design, coding and debugging of a simulation program is often time consuming and difficult to understand – it may 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 a poor estimate
- Execution can be computationally expensive
- Statistical analysis of the output is difficult – e.g. how long should the simulation run before taking an average of the results?

Events

Each event contains a time stamp identifying 'when it occurs'

Denotes some change in the state of the system to be simulated e.g. 'packet arrived'

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

- Initialize the clock to 0
- Initialise the event list
- WHILE termination criterion is not met
    - remove a smallest 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 **least** time stamp

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

Physical systems

Concerned with the problem of representing so-called 'physical systems'

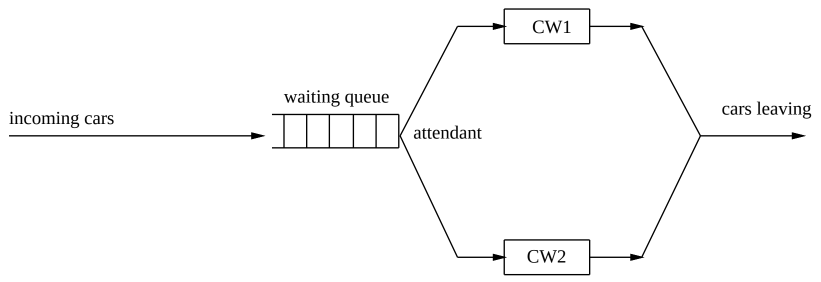
A physical system consists of one or more physical processes

Each physical process operates independently, aside from interaction via messages (events)

All state changes are summarized in the event exchanges

— note the similarity with object-oriented styles of programming in which objects communicate solely by method invocations. Older object-oriented languages tended to refer to these as message-send operations.

An example



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

A model of a car wash system consists of an attendant (**att**), two car washes, (**cw1**, **cw2**), the entrance to the system (**source**) and the 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 non-busy then the car at the head of the queue is sent to it

An 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

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

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
15	11	21	att	cw2	car 4
16	12	22	cw1	sink	car 3
17	-	22	cw1	att	idle
18	13	22	att	cw1	car 5
19	14	25	source	att	car 6
20	15	30	cw1	sink	car 5
21	-	30	cw1	att	idle
22	16	30	att	cw1	car 6

Modelling stochastic systems

To represent the stochastic nature of the systems being modelled, simulation requires the use of random variables

An input to the simulation may have a stochastic distribution e.g. arrivals at a queue

Simulation model needs to sample random variables from the given distribution to "re-create" the input process

Random variates can be generated for a wide range of theoretical and empirical distributions

This process requires only a sequence of independent random variates with a uniform distribution!

Random number generation

Many random number generators in use today are not particularly good

It is important **not** to generate random numbers with a method chosen at random – 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 – these mechanisms are not usually suited to generating large volumes of random data

How can we generate a large **random** sequence in a **deterministic manner**?

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

> — a deterministic algorithm running in bounded space *must* repeat the sequence after some time

Random number generation (2)

Important requirements include

- The algorithm should be fast
- The storage requirements should be low
- The random sequence should only repeat with a very long cycle
- 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

Linear congruential method

Many successful random number generators are special cases of the following scheme, introduced by Lehmer in 1951

It is based on 4 magic numbers:

The starting value X_0	$X_0 \geq 0$
The multiplier a	$a \geq 0$
The increment c	$c \geq 0$
The modulus m	$m \geq X_0, m > a, m > c$

The desired sequence of random numbers X_n is then generated by setting

$$X_{n+1} = (aX_n + c) \text{ modulo } m$$

Multiplicative congruential method

The special case where $c = 0$ yields this method

When $c = 0$ the period of the sequence is reduced

But, to our advantage, generation is a little faster

How do we choose the values?

Well, $a = 1$ can be rejected immediately

This would mean that $X_n = (X_0 + nc) \text{ modulo } m$

The case $a = 0$ is even worse.

So $a > 1$ is necessary.

Choice of values

Since the period can never be greater than m this should be chosen to be as large as possible

The following values were chosen for efficient computation of random numbers on the IBM/370

$$\begin{aligned} a &= 16807 \\ m &= 2147483647 \\ c &= 0 \\ X_0 &= 314159 \end{aligned}$$

Knuth has observed that the value a should end in the digits $x21$ where x is even e.g. 31415821

Implementation problem: what about overflow?

Tests for randomness

Various statistical tests exist

These include tests for uniformity (the Kolmogorov Smirnov test and Chi-Square test)

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 checks the correlation between every m numbers starting with the i^{th} number.

Random variate generation

We have a sequence of pseudo-random uniform variates. How do we generate variates from different distributions?

Random behavior can be programmed so that the random variables appear to have been drawn from a particular probability distribution

If $f(x)$ is the desired pdf, then consider the CDF

$$F_x(x) = \int_{-\infty}^{x} f(x)dx$$

This is non-decreasing and lies between 0 and 1

Random variate generation (2)

Given a sequence of random numbers r_i distributed over the same range $(0, 1)$

Let each value of r_i be a value of the function $F_x(x)$

Then the corresponding value x_i is uniquely determined

The sequence x_i is randomly distributed and has the probability density function $f(x)$

Uniform distribution

Consider the **Uniform** random variable on the interval (a, b) where

- a, b are arbitrary real numbers
- $a < b$

The distribution function is

$$F_x(x_i) = (x_i - a)/(b - a)$$

for x_i in the interval (a, b)

Given r_i on interval $(0, 1)$ the problem is that of obtaining x_i

Starting with $r_i = (x_i - a)/(b - a)$ it is obvious that $x_i = (b - a)r_i + a$.

Exponential distribution

For the exponential distribution

$$F_x(x_i) = 1 - \mathrm{e}^{-\lambda x_i}$$

for positive x_i

Thus

$$\begin{aligned} r_i &= 1 - \mathrm{e}^{-\lambda x_i} \\ 1 - r_i &= \mathrm{e}^{-\lambda x_i} \\ \ln(1 - r_i) &= -\lambda x_i \\ x_i &= -\frac{1}{\lambda}\ln(1 - r_i) \end{aligned}$$

Note that r_i has the same distribution as $1 - r_i$ so we would in reality use

$$x_i = -\frac{1}{\lambda}\ln(r_i)$$

Other random variates can be derived in a similar fashion.

Simulation performance measures

As the simulation itself is stochastic, so too will be the output of the observations of a stochastic process

It is critical to realize that a simulation can only yield **estimates** for performance measures

There will **always** be some error in the estimates

The error can be made small by

- running the simulation for a very long time until sufficient samples have been taken
- running the same simulation with a different random number generator, and collecting the results from multiple runs

Utilization

The utilization is the fraction 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

In the case of a k-server, the busy times can be estimated together and divided by kT

There are 2 obvious ways to aggregate the busy times:

- Sample the system and observe whether servers are busy or idle
- When the server becomes busy the time is saved, and when it becomes idle again the difference in the two times is the total busy time.

Throughput, queue length

A simple count is kept of the number of customers receiving service. The **throughput** is this value divided by the total time.

There are at least two ways to estimate **queue length**:

- Estimate the queue length distribution, then use to obtain the mean
- View the queue length as a function of time: the mean queue length is $1/T$ of integral of this function.

Queue length

- Each time the queue length changes the time t_{i+1} is stored
- subtract previous recorded time t_i from current time
- multiply by previous queue length n_i

Sum these areas to give mean queue length (for M observations) by

$$\bar{N} = \frac{1}{T} \sum_{i=1}^{M} n_i (t_{i+1} - t_i) \tag{7}$$

Queueing time

Two obvious ways to obtain **queueing time**

- observe the queueing times and take their average
- simply use Little's law

Statistical analysis of results

Each independent replication of a simulation experiment will yield a different outcome

To make a statement the about accuracy we have to estimate the distribution of the estimator

Need to determine that the distribution becomes asymptotically centered around the true value

Cannot be established with certainty in the case of a finite simulation

The usual method used to estimate variability is to produce "confidence interval" estimates

Confidence intervals

Given some point estimate p a we produce a confidence interval $(p - \delta, p + \delta)$

The "true" value is estimated to be contained within the interval with some chosen probability, e.g. 0.9

The value δ depends on the confidence level – the greater the confidence required the larger the value of δ

Let $x_1, x_2, \ldots, x_n$ be the values of a random sample from a population determined by the random variable X

Let the mean of X be $\mu = \mathbb{E}(X)$ and variance σ^2

Assume: either X is normally distributed or n is large

Then: by the law of large numbers, $\bar{X} \approx$ normally distributed

Confidence intervals (2)

Then, given σ the $100(1-\alpha)\%$ confidence interval is given by

$$\bar{x} \pm \delta$$

where

$$\delta = \frac{z_{\alpha/2}\sigma}{\sqrt{n}} \qquad (8)$$

z_α is defined to be the largest value of z such that $\mathbb{P}(Z > z) = \alpha$ and Z is the standard normal random variable

Can be taken from tables of the normal distribution

For example, for a 95% confidence interval $\alpha = 0.05$ and $z_{\alpha/2} = z_{0.025} = 1.96$

Using Student's T

When we know neither μ nor σ we can use the observed sample mean $\bar{x}$ and sample standard deviation s

If n is large then we simply use s for σ in Equation (8).

If n is small and X is normally distributed then we may use

$$\delta = \frac{t_{\alpha/2}s}{\sqrt{n}} \qquad (9)$$

$t_{\alpha/2}$ is defined by $\mathbb{P}(T > t_{\alpha/2}) = \alpha/2$

T has a Student-t distribution with $n-1$ degrees of freedom

This is the more frequently used formula in simulation models

Stopping rules

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

We also need to be aware of transient conditions at the start of the simulation

We can also repeat the simulation several times to obtain many samples for the ensemble average

These multiple runs are called replications

Given a performance measure L we can approach the ensemble process by

$$L = \lim_{n \to \infty} \sum_{i=1}^{n} \frac{L_i}{n}$$

Stopping rules (2)

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

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

Can we avoid this?

If we use only a single, long run, and break up our samples into n blocks, each of these can form a sample L_i

Stopping rules (3)

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, or once the confidence interval around L becomes sufficiently narrow

Operational Analysis

Based on the idea of **observation** rather than a probabilistic description of system behaviour

It is also concerned with quantities 'directly related' to these observed quantities

Operational analysis makes very weak assumptions about the system being modelled

— unlike simulation which requires detailed system knowledge, or the techniques from queuing theory (in the next section) which depend extensively on the probability distributions involved

We begin by examining some **fundamental quantities** and **operational laws**

Operational analysis (2)

We examine a system for some time recording the customer arrivals and departures, and define the quantities of interest:

- T, the length of time we observe the system
- A, the number of arrivals observed
- C, the number of departures (or completions of service) observed
- W, the job-time product: the sum of the durations of all customers over the observation period

If the system is a single resource, then we can also measure:

- B, the time for which the resource was busy.

Operational analysis (3)

From these we can define the following quantities:

- Arrival rate, $\lambda \triangleq \frac{A}{T}$

  — the mean number of arrivals per unit time

- Throughput, $X \triangleq \frac{C}{T}$.

  — the mean number of departures per unit time

- Mean number of customers, $N \triangleq \frac{W}{T}$

  — expressing the job-time product in terms of N and T

- Mean residence time, $R \triangleq \frac{W}{C}$

  — expressing the job-time product in terms of R and C

Operational analysis (4)

For a single resource, we can also define:

- Utilization, $U \triangleq \frac{B}{T}$

  — the proportion of the time that the resource is busy

- Average service requirement, $S \triangleq \frac{B}{C}$

  — the mean time that the resource spends for each departure

The utilization law

Our first "law" is just an algebraic identity

$$U \triangleq \frac{B}{T} = \frac{C}{T}\frac{B}{C} = XS$$

This is termed the **utilization law**.

For example, if the throughput (X) is 5 departures/sec and the service demand (S) is 0.1 sec/departure then the utilization (U) is 50%.

Little's law

Similarly, we can derive the by-now-familiar **Little's Law**

$$N \triangleq \frac{W}{T} = \frac{C}{T}\frac{W}{C} = XR$$

For example, if the throughput (X) is 5 customers/sec and the mean residence time is 1 sec then the average number in the system is 5.

- Very weak assumptions about the system
- Applicable to a wide range of systems
- Can be applied recursively to subsystems and to individual resources

— but take care that mutually-consistent values are used for X and R, in particular whether they apply to the queue, the server or the entire system

An example

Observe system for $T = 10$ sec

4 customers spend 10 s in the system

One customer who spends 5 s

Then we have $W = 4 \times 10 + 1 \times 5 = 45$ s.

If also $A = C = 5$ then

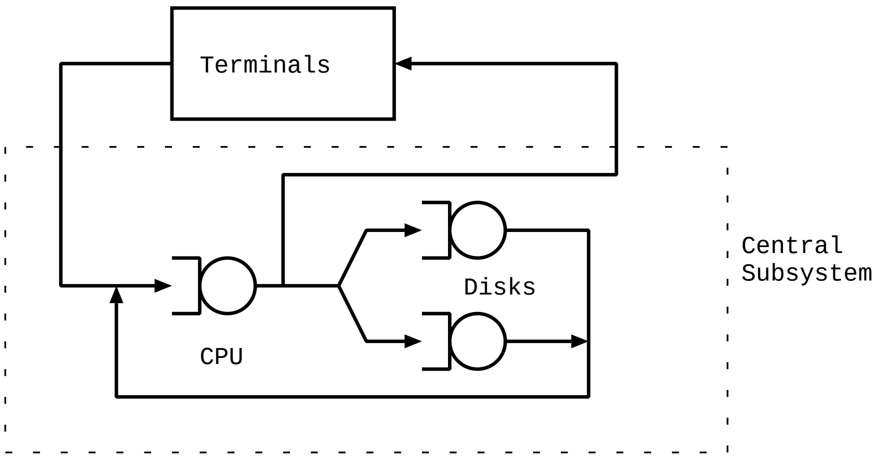
$X = C/T = 5/10 = 0.5$ customers per second

$\lambda = 5/10 = 0.5$ customers per second

$N = 45/10 = 4.5$ customers

$R = 45/5 = 9.0$ s per customer

A simple interactive system



Fixed number, M, of users logged on

Customer is at the terminal whilst thinking

The think time, Z is the average time a user spends between receiving a prompt and responding

A customer not thinking is somewhere inside the central subsystem

A simple interactive system (2)

We can use Little's Law to relate some observable quantities in the central subsystem:

- N is the number of customers in the central subsystem $0 \leq N \leq M$
- X is the rate at which customers complete in the central subsystem
- R is the average time a customer spends in the central subsystem (intuitively equivalent to "response time")

If we observe that system throughput is 0.5 interactions per second and we find on average 7.5 users in the subsystem then

$$R = \frac{N}{X} = \frac{7.5}{0.5} = 15 \text{ s}$$

from Little's Law

A simple interactive system (3)

We can also apply Little's law to the entire system.

This is a closed system so the number of customers is fixed as M.

We can split the time spent during an interaction into the response time (R) and the *think time* (Z). The *residence time* is $R + Z$.

We consequently derive the **interactive system** version of Little's Law:

$$M = X(R + Z)$$

A simple interactive system (4)

With 10 users logged on, 5 s average think time and an average response time of 15 s,

$$X = \frac{M}{R + Z} = \frac{10}{15 + 5} = 0.5 \quad \text{interactions/sec}$$

Under heavy load (M large or Z small)

$$U \approx 1$$

Using the Utilization Law the throughput $X \approx \frac{1}{S}$ and hence

$$R = \frac{M}{X} - Z \approx \frac{1}{S}M - Z$$

$\Rightarrow$ The response time grows approximately linearly with the number of users M.

Visit counts and forced flow

Extend our notation to allow the modelling of multiple devices

We will use subscripts $i = 1, 2, \ldots, K$ to identify each device, e.g. X_i is the throughput at device i

Assume that the service required by a customer is an inherent property of the **customer** not of the state of the system

A **visit count** for a device is the number of completions at that device for every completion from the system

$$V_i \triangleq \frac{C_i}{C}$$

Where C_i is the number of completions at device i

Recall that in a **feed-forward** queueing network, $0 \leq V_i \leq 1$ because each customer visits a given device at most once

The forced flow law

Since $X = \frac{C}{T}$, we have

$$X_i = \frac{C_i}{T} = \frac{C_i}{C}\frac{C}{T} = V_i X$$

the **Forced Flow Law**

For example, if the throughput from the entire system is 20 customers per second and each customer visits a given device 3 times then the throughput of that device must be 60 completions per second.

If devices are load independent, then define the *service demand* a customer makes on a device i,

$$D_i \triangleq V_i S_i$$

— be careful to distinguish the *service requirement* (S_i) from the *service demand* (D_i)

Queue lengths at a server

Applying the utilization law at each device:

$$U_i = X_i S_i = (XV_i)S_i = X(V_i S_i) = XD_i$$

Similarly, applying Little's law at each device:

$$N_i = X_i R_i$$

R_i is the residence time at device i and can be decomposed into the time spent queuing and the time spent in service, approximated by R_i^*:

$$\begin{aligned} R_i^* &= N_i S_i + S_i \\ &= R_i^* X_i S_i + S_i \\ &= R_i^* U_i + S_i \end{aligned}$$

Queue lengths at a server (2)

Hence:

$$R_i^* = \frac{S_i}{1 - U_i}$$

substituting:

$$\begin{aligned} N_i &= X_i R_i^* \\ &= \frac{X_i S_i}{1 - U_i} \\ &= \frac{U_i}{1 - U_i} \end{aligned}$$

Observe that:

- N_i is zero when U_i is zero
- N_i grows rapidly without bound as U_i approaches one

Bottleneck analysis

A bottleneck in a system is a hindrance to movement or progress

Given the forced flow assumption, at high loads system performance is determined by the device with the highest utilization: **the bottleneck**

The ratio of the completion rates of any two devices is

$$\frac{X_i}{X_j} = \frac{V_i X}{V_j X} = \frac{V_i}{V_j}$$

Since $U_i = X_i S_i$, we have a similar property for utilizations

$$\frac{U_i}{U_j} = \frac{X_i S_i}{X_j S_j} = \frac{V_i S_i}{V_j S_j}$$

Bottleneck analysis (2)

A system is **load independent** if:

- V_i are intrinsic properties of customers,
- S_i are independent of the queue length at i

In such cases, the throughput and utilization ratios are the same for all loads

This can be used to determine asymptotes for X and R:

In general, $U_i \leq 1$ and $X_i \leq \frac{1}{S_i}$

A device i becomes **saturated** as $U_i \to 1$

Thus, as $U_i \to 1$, we see $X_i \to \frac{1}{S_i}$: device i is working as fast as it can and consequently serves one customer every S_i.

Bottleneck analysis (3)

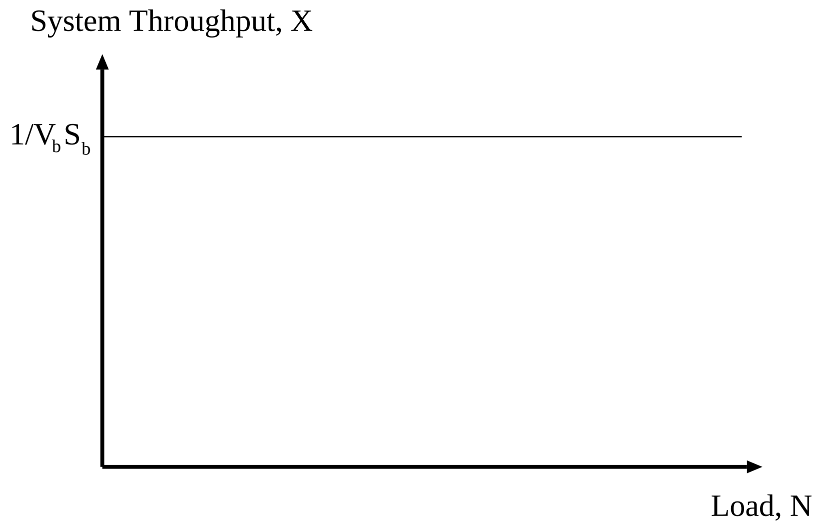
Use the subscript b to denote a device capable of saturating

Since the utilization ratios are fixed, the device i with the largest $V_i S_i$ will be the first to achieve 100% utilization as N increases:

$$V_b S_b = \max\{V_1 S_1, ..., V_K S_K\}$$

so the bottleneck is determined by device and workload (the V_i and S_i) properties

Maximum throughput



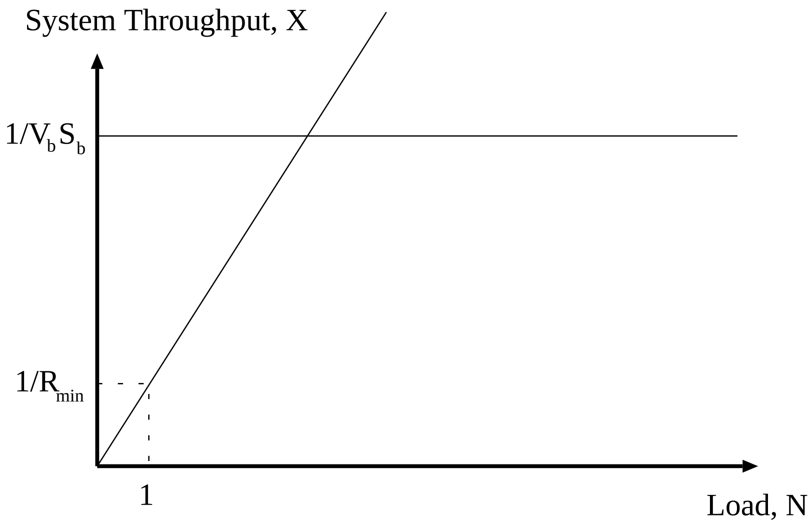
By the forced flow law:

$$X = \frac{X_b}{V_b}$$

So, as $U_b \to 1$ and $X_b \to 1/S_b$:

$$X_{\max} = \frac{X_b}{V_b} \to \frac{1}{V_b S_b}$$

Maximum throughput (2)



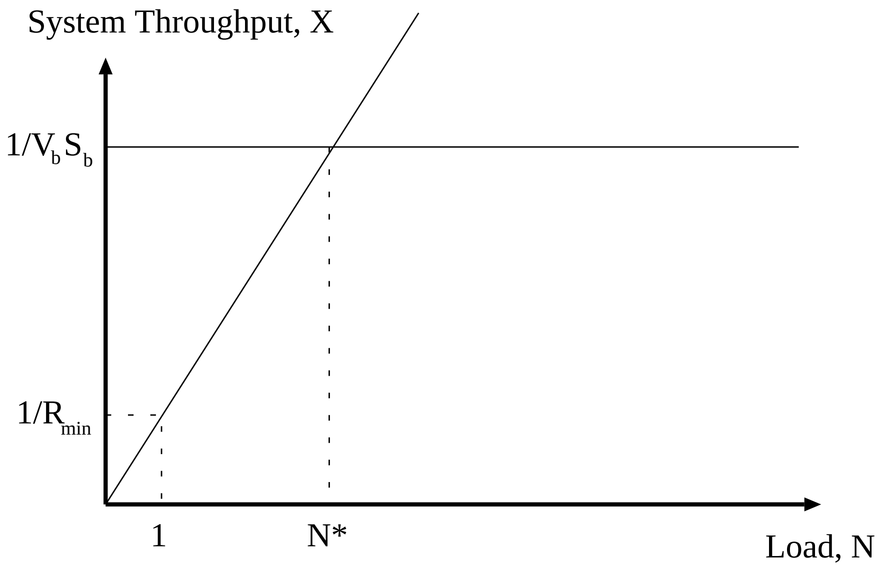
The total per-customer service required:

$$R_{\min} = \sum_{i=1}^{K} V_i S_i$$

$$\Rightarrow \quad X \leq \frac{N}{R_{\min}}$$

$R_{\min}$ denotes the smallest possible value of mean response time, occurring when $N = 1$

Maximum throughput (3)



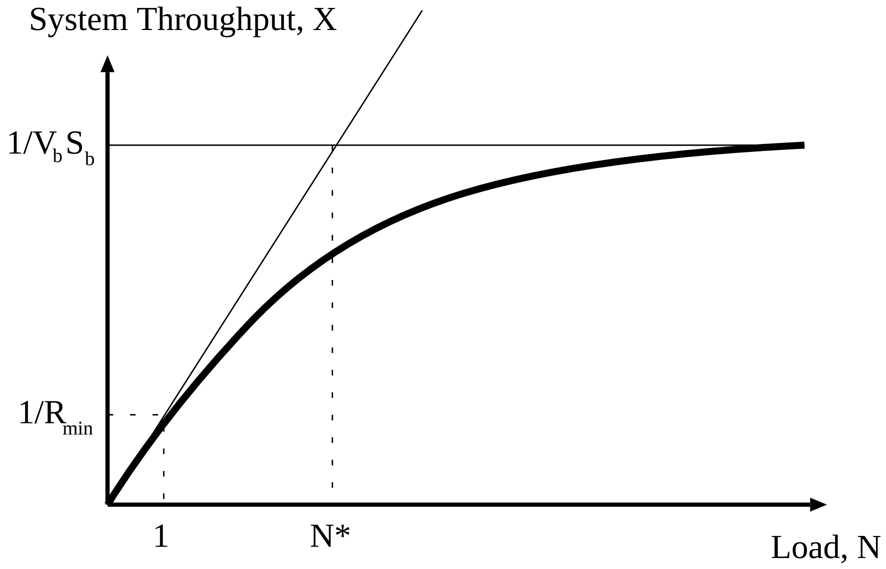
If $k \leq K$ jobs always avoid each other then:

$$X = \frac{k}{R_{\min}} \leq \frac{1}{V_b S_b}$$

$$k \leq \frac{R_{\min}}{V_b S_b} = \frac{\sum_{i=1}^{K} V_i S_i}{V_b S_b} = N^*$$

Beyond N^* queueing is certain

Maximum throughput (4)



- It stays below $1/(V_b S_b)$ because, at that point, a bottleneck will be operating at maximum utilization
- It stays below the straight line $X = N/R_{min}$ because the throughput is limited by the number of customers in service

Interactive response time

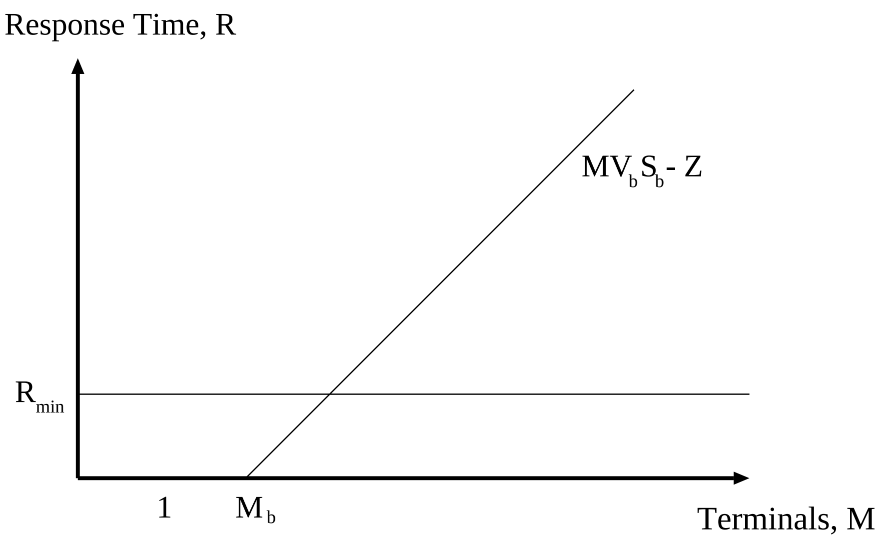
- X throughput
- M terminals
- Average think time Z
- Recall the interactive system version of Litle's law:

$$R = \frac{M}{X} - Z$$

By considering a bottleneck device b:

$$\begin{array}{rrcll} & X & \leq & \dfrac{1}{V_b S_b} & \\ \Rightarrow & R & \geq & M V_b S_b - Z & \\ \Rightarrow & R & \geq & M V_i S_i - Z & \forall i \in \{1..K\} \end{array}$$

Interactive response time (2)



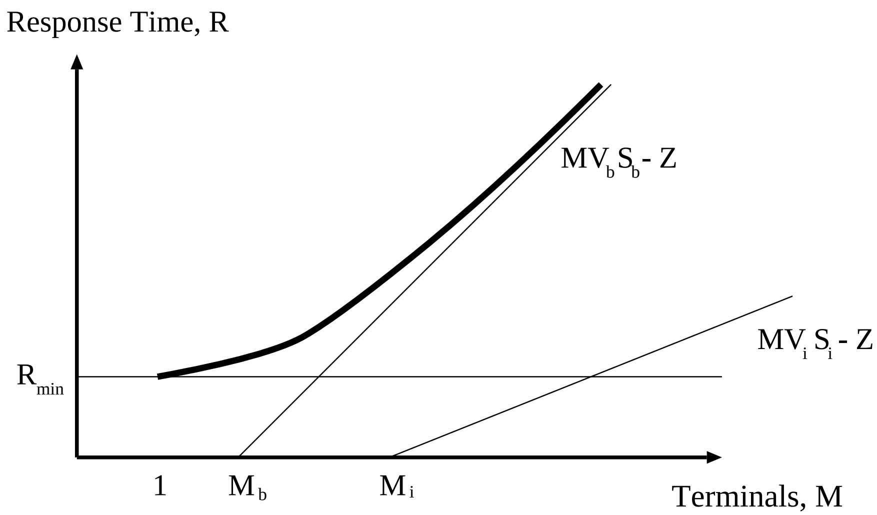
The response time asymptote meets the x axis at

$$M_b = \frac{Z}{V_b S_b}$$

It intersects the minimum response time $R_{\min}$ at M_b^* where

$$M_b^* V_b S_b - Z = R_{\min}$$

Interactive response time (3)



Thus

$$M_b^* = \frac{R_{min} + Z}{V_b S_b} = N^* + M_b\,, \quad \text{say}$$

When there are more than M_b^* terminals, queueing is certain to exist

Summary

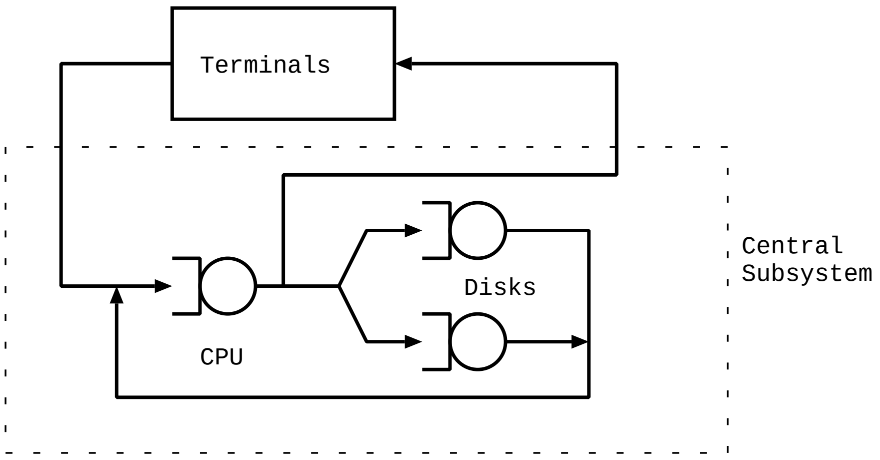
- The largest of the $V_i S_i$ determines the bottleneck b.
- The sum of these products determines the smallest possible response time $R_{\min}$.
- Queueing cannot be avoided when N exceeds

$$N^* = \frac{R_{\min}}{V_b S_b}$$

- Queueing cannot be avoided in an interactive system when the number of logged-on terminals exceeds

$$M_b^* = N^* + \frac{Z}{V_b S_b}$$

Example: interactive system

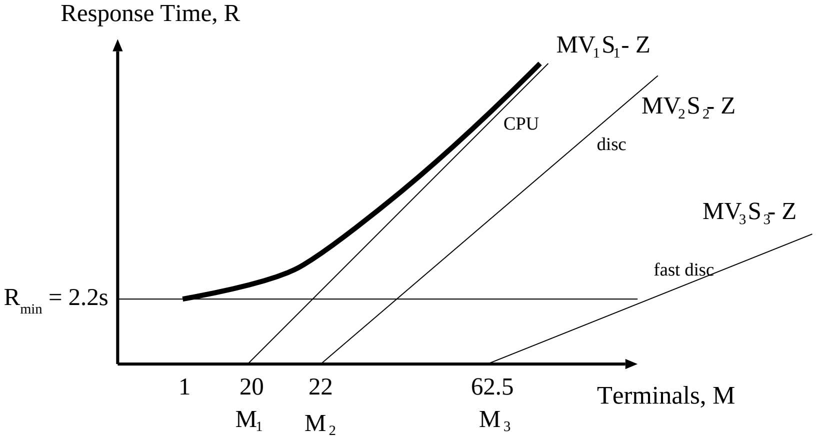


Suppose that $Z = 20$ s.

No.	device	S_i (s)	V_i	$D_i = V_i S_i$
1	CPU	0.05	20	1.00
2	disk	0.08	11	0.88
3	fast disk	0.04	8	0.32
			$R_{\min}$	2.20

Question: Is an 8 second response time feasible with 30 users logged on? If not, what changes are required?

Example: interactive system (2)



$$\begin{aligned} V_1 S_1 &= 1\text{s} \quad \textit{(bottleneck)} \\ V_2 S_2 &= 0.88\text{s} \\ V_3 S_3 &= 0.32\text{s} \\ \Rightarrow R_{\min} &= \sum_{i=1}^{3} V_i S_i = 2.2\text{s} \end{aligned}$$

For $M = 30$, the response time asymptote requires:

$$R \geq 30 \times 1 - 20 = 10 \quad \text{s}$$

So the answer is **no**

Example: interactive system (3)

To make an 8 second response time feasible, we need to speed up the CPU, so that the new service time obeys

$$M V_1 S_1' - Z \leq 8$$

or

$$S_1' \leq \frac{20 + 8}{30 \times 20} = .047 \quad \text{s}$$

which is a 7% speed up in the CPU.

Then $V_1 S_1' = 0.93$ is still the largest product so the CPU is still the bottleneck.

Example: interactive system(4)

Question: Is 10s response time feasible when 50 users are logged on? If not, how much CPU speedup is required?

If $S_1 \to 0$, the disk will become bottleneck:

$$R \geq MV_2S_2 - Z$$

For $M = 50$, this is

$$R \geq 50 \times 0.88 - 20 = 24 \quad \mathrm{s}$$

so a 10s response time is not feasible with $M = 50$ and no amount of CPU speedup is capable of achieving it.

Balanced system bounds

Balanced system bounds provide **tighter** bounds at mid-range loads than bottleneck analysis

A system is **balanced** if for any load the utilizations of all devices are equal

Balanced systems exhibit the following important property that

$$U_i(N) = \frac{N}{N+K-1}$$

So the system throughput is given by

$$X(N) = \frac{U_i}{D_i} = \frac{N}{N+K-1} \times \frac{1}{D_i}$$

Balanced system bounds (2)

For example,

$$\begin{array}{lll} N = 1 & K = 2 & U_1 = U_2 = \frac{1}{2} \\ N = 1 & K = 100 & U_1 = \ldots = U_{100} = \frac{1}{100} \\ N = 100 & K = 2 & U_1 = U_2 = \frac{100}{101} \\ N = 2 & K = 2 & U_1 = U_2 = \frac{2}{3} \end{array}$$

Look at the system and find

- $D_{\max}$ – maximum demand at any device
- $D_{\min}$ – minimum device demand
- D_{av} – average demand at each device
- $D = D_{\mathrm{tot}}$ – total demand across all devices

So, we have $D_{\mathrm{av}} = D/K$

Balanced system bounds (3)

We can construct imaginary balanced systems related to our system

pess1: balanced system with K devices each with a demand of $D_{\max}$

opt1: balanced system with K devices each with a demand of $D_{\min}$

The throughput of system **pess1** is

$$\frac{N}{N+K-1} \times \frac{1}{D_{\max}}$$

The throughput of system **opt1** is

$$\frac{N}{N+K-1} \times \frac{1}{D_{\min}}$$

Balanced system bounds (4)

So for the system being modelled we have

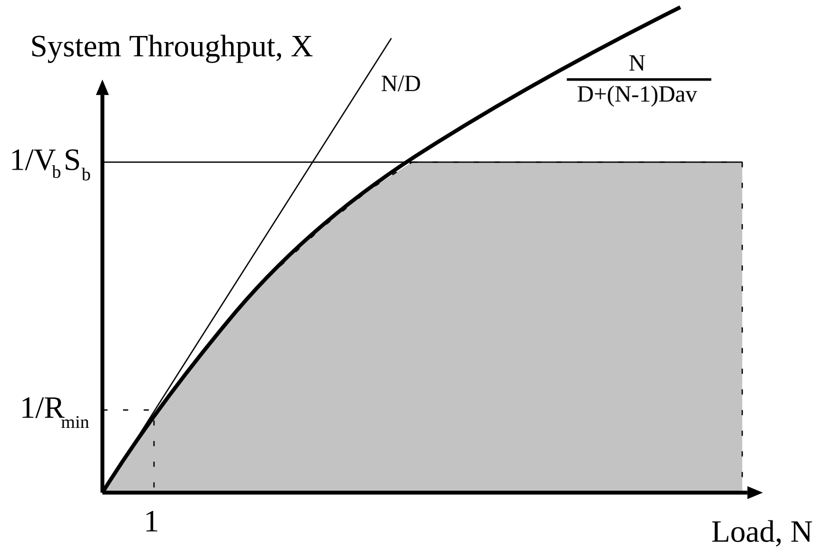
$$\frac{N}{N+K-1} \times \frac{1}{D_{\max}} \leq X(N) \leq \frac{N}{N+K-1} \times \frac{1}{D_{\min}}$$

And since $N = XR$,

$$(N+K-1)D_{\max} \geq R(N) \geq (N+K-1)D_{\min}$$

This will give tighter bounds on the mid-range performance than the bottleneck bounds

Balanced system bounds (5)

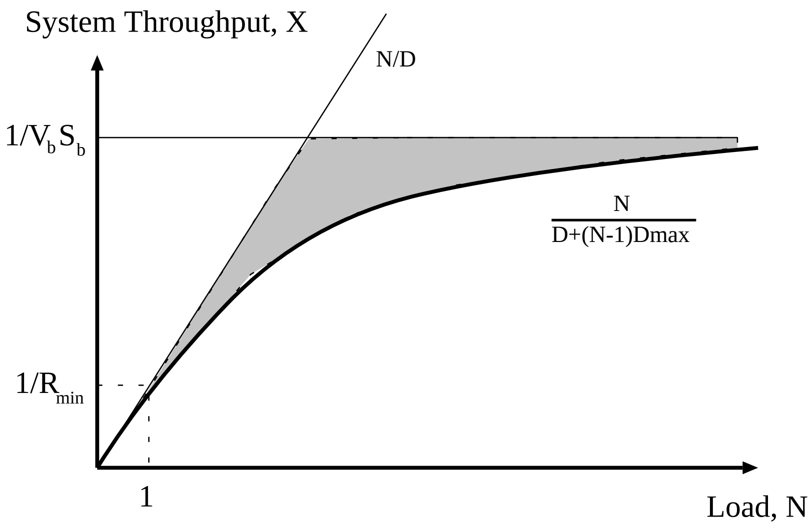


We can do even better by considering what the **best** performance is that the system can achieve : spread the load out evenly among all the devices.

opt2: D_{av} at each of the K devices

$$X(N) = \frac{N}{N+K-1} \times \frac{1}{D_{\text{av}}} = \frac{N}{D+(N-1)D_{\text{av}}}$$

Balanced system bounds (6)



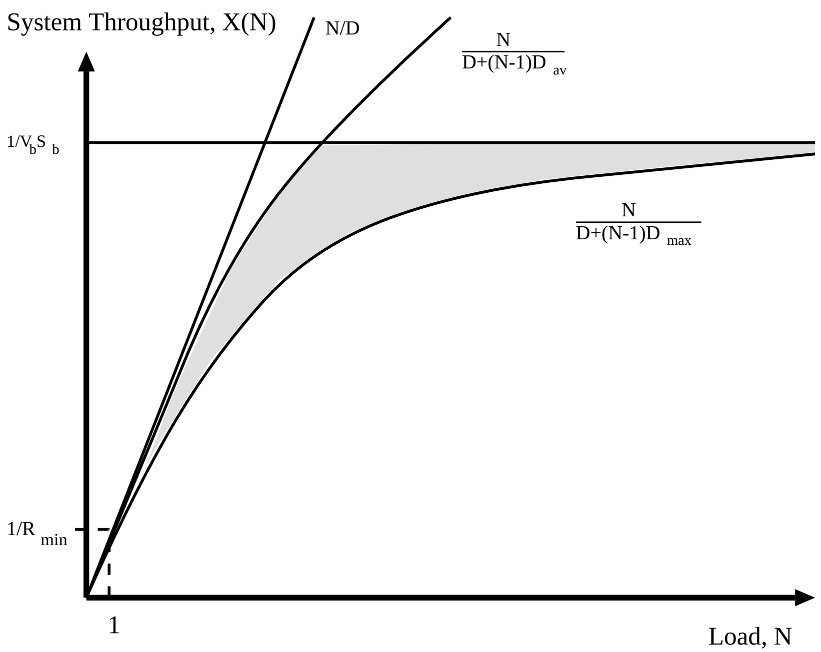
Now, what is the **worst** system subject to the constraints that D and $D_{\max}$ remain fixed?

Answer: place $D_{\max}$ at as many devices as possible and 0 at the rest

pess2: $D_{\max}$ at each of the $\frac{D}{D_{\max}}$ devices

$$X(N) = \frac{N}{N+K-1} \times \frac{1}{D_{\max}} = \frac{N}{D+(N-1)D_{\max}}$$

Balanced system bounds (7)



$$\frac{N}{D+(N-1)D_{\max}} \leq X(N) \leq \frac{N}{D+(N-1)D_{\mathrm{av}}}$$

$$D+(N-1)D_{\max} \geq R(N) \geq D+(N-1)D_{\mathrm{av}}$$

Balanced system bounds (8)

As $N \to \infty$,

$$\frac{N}{D + (N-1)D_{\max}} \to \frac{1}{D_{\max}}$$

Note that at high loads the bottleneck bounds are the limiting high bound

The asymptotic bottleneck bound $\frac{1}{D_{\max}}$ and the optimistic balanced bound intersect at $N^\dagger$ where

$$\frac{1}{D_{\max}} = \frac{N^\dagger}{D + (N-1)D_{\mathrm{av}}}$$

So that

$$N^\dagger = \frac{D - D_{\mathrm{av}}}{D_{\max} - D_{\mathrm{av}}}$$

Markov processes

A **Markov process** is a family of random variables $X(t)$ indexed by a time parameter t in which

- $X(t)$ denotes the **state** at time t
- The next state depends only on the current state

We are concerned with the particular case of Markov processes in which the state space is discrete, termed **Markov chains**

If we denote successive states as X_1, X_2, $\ldots$

$$\begin{aligned} \mathbb{P}\left(x_{n+1} = X_{n+1} | x_1 = X_1, \ldots, x_n = X_n\right) &= \\ \mathbb{P}\left(x_{n+1} = X_{n+1} | x_n = X_n\right) & \qquad (10) \end{aligned}$$

We **cannot know** how long process has been in a given state

Other types of stochastic process

A Markov process is a particular kind of **stochastic process**

- Birth death process: a Markov process in which transitions occur only between neighboring states
- Semi-Markov Process: relax memoryless condition for state residence time but Markov property holds at transition instants
- Random Walks: refer to Probability 1a notes
- Renewal Process: $X(t)$ counts state transitions in $(0, t)$

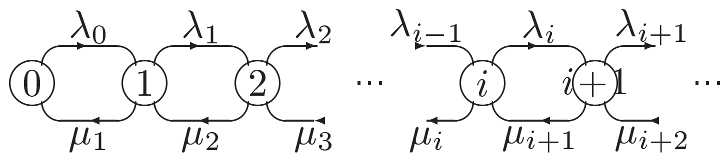
Birth death processes

A **birth death process** is a special case of a continuous-time Markov chain in which we allow transitions only between neighboring states

The state space is the set of non-negative integers and we allow only a single birth or death per transition. Thus given $X_n = i$

$$X_{n+1} = \begin{cases} i+1 & \text{birth (or arrival)} \\ i-1 & \text{death (or departure)} \end{cases}$$

We use λ_i to represent the birth rate in state i, and μ_i to represent the death rate in state n



Time dependent solution of BDP

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

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

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

Probability of > 1 event in Δt is $o(\Delta t)$

$o(\Delta t)$ denotes a quantity which becomes negligible when compared with Δt as $\Delta t \to 0$

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

Chapman Kolmogorov equations

For $i \geq 1$

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

For $i = 0$

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

Chapman Kolmogorov equations

We derive differential-difference equations from these by dividing through by Δt and taking the limit as $\Delta t \to 0$:

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

for $i \geq 1$ and

$$\begin{aligned} \frac{dP_0(t)}{dt} &= -\lambda_0 P_0(t) \\ &+ \mu_1 P_1(t) \end{aligned} \tag{11}$$

The time dependent solution is **extremely difficult** for 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,

$$\lim_{t \to \infty} P_i(t) = p_i \qquad \text{exists}\,.$$

Now $dP_i(t)/dt$ is 0 and Equations (11) become

$$\begin{aligned} 0 &= -(\lambda_i + \mu_i)p_i + \mu_{i+1}p_{i+1} + \lambda_{i-1}p_{i-1} \\ 0 &= -\lambda_0 p_0 + \mu_1 p_1 \end{aligned} \tag{12}$$

Rewriting gives

$$\begin{aligned} p_{i+1} &= \frac{\lambda_i + \mu_i}{\mu_{i+1}} p_i - \frac{\lambda_{i-1}}{\mu_{i+1}} p_{i-1} \quad (i \geq 1) \\ p_1 &= \frac{\lambda_0}{\mu_1} p_0 \end{aligned} \tag{13}$$

Stochastic balance

Provides short cut method to obtaining Equation (13)

Require total flow into a state to equal total flow out of a state under steady-state conditions

Intuitively the total flow is the product of the steady state probabilities and the flow rates (λ, μ)

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

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

describes flow into state i.

This is called the **global balance** equation

Stochastic balance (2)

The two equations:

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

describing flow from state i to state $i+1$, and

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

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 rise to the product solution

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

Stochastic balance (3)

Since the sum of state probabilities must be unity,

$$\begin{aligned} p_0 + \sum_{k=1}^{\infty} p_k &= 1 \\ p_0 + \sum_{k=1}^{\infty} p_0 \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}} &= 1 \\ p_0 \left[1 + \sum_{k=1}^{\infty} \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}} \right] &= 1 \end{aligned}$$

so that

$$\begin{aligned} p_0 &= \left[1 + \sum_{k=1}^{\infty} \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}} \right]^{-1} \\ p_k &= p_0 \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}} \end{aligned}$$

which are known as the **general flow balance equations**

The M/M/1 queue

The BDP maps well onto our domain of study – 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 balance equations:

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

The M/M/1 queue (2)

Writing $\rho = \dfrac{\lambda}{\mu}$

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

Consequently, the number in the system is geometrically distributed

$$p_k = (1-\rho)\rho^k, \qquad k = 0, 1, 2, \ldots$$

If $\rho > 1$, i.e. if $\lambda > \mu$ the system will not reach equilibrium

The M/M/1 queue (3)

What is the average number of customers, $\overline{N}$ in the system?

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

The M/M/1 queue (4)

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

$$\overline{N} = \lambda\overline{T}$$

which is Little's result restated. In our case

$$\begin{aligned}\overline{T} &= \frac{\overline{N}}{\lambda} \\ &= \frac{\frac{1}{\mu}}{1-\rho}\end{aligned}$$

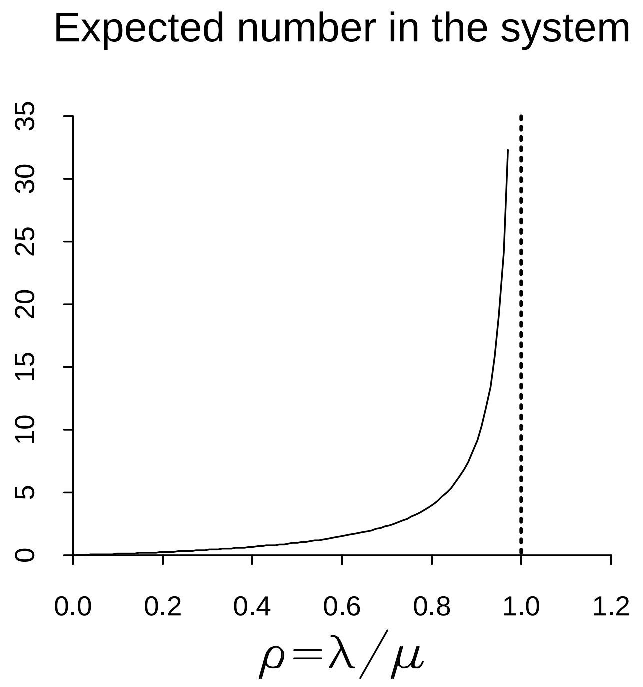
which is the M/M/1 average response time.

Note that

- $\frac{1}{\mu}$ is the average service time
- ρ is the utilization

Performance at high load

At high utilizations ρ approaches one and the response time and queue lengths are unbounded.



Response time

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

utilization (ρ)			response time ($\overline{T}$)		
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 response 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

He 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?

Example Solution

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

$$\rho = 5/6, \text{ and hence } \bar{N} = 5$$

Average number of customers waiting:

$$\bar{N}_q = 4\frac{1}{6}$$

How often is the barber idle?

$$p_0 = 1 - \rho = \frac{1}{6}$$

How often is no seat free?

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

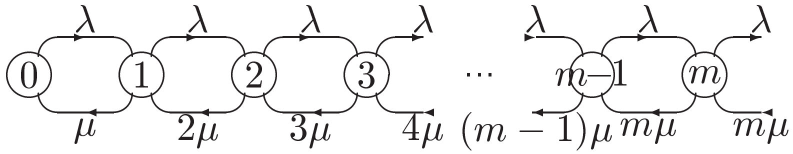
M/M/m — m servers

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 $k > K$, $p_k = 0$.

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

$$
\begin{aligned}
p_k &= p_0 \rho^k \quad \text{for } k \leq K \\
p_0 &= \frac{1}{1 + \sum_{k=1}^{K} \rho^k} \\
&= \frac{1}{1 + \frac{\rho - \rho^{K+1}}{1-\rho}} \\
&= \frac{1 - \rho}{1 - \rho^{K+1}}
\end{aligned}
$$

Note that p_0 is greater than in the M/M/1 case.

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

M/M/1//N – finite population

Single server, unbounded queue capacity and a population of N customers.

We solve this system by modifying the λ_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 λ.

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.

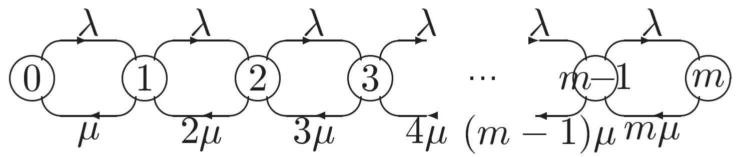
Suppose that calls arrive at the link according to a Poisson process of rate λ.

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



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

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

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

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

and $p_k = 0$ for $k > m$.

M/M/m/m (2)

Solving for p_0 gives

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

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

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

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

BDM 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.

BDM extensions (2)

If the desired birth rate is λ, then let each stage have a rate 2λ.

The average time to get through the combined birth process will be

$$\overline{\tau} = \frac{1}{2\lambda} + \frac{1}{2\lambda} = \frac{1}{\lambda}.$$

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

$$\sigma^2_{single} = \frac{1}{(2\lambda)^2}.$$

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

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

BDM 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 $\frac{1}{\sqrt{r}}$.

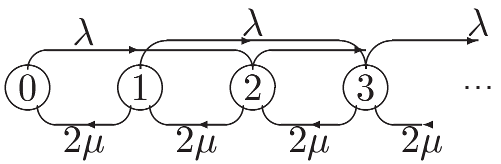
The distribution that describes this process is the Erlangian distribution denoted E_r.

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 λ

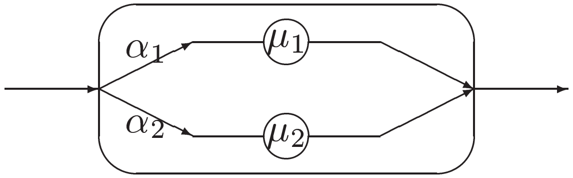
A departure from a stage reduces the number of stages to be served by 1 and occurs at rate 2μ.



Parallel Servers

Combining stages in series reduces the coefficient of variation.

If, instead, we combine them in parallel with a probability α_i of choosing the i^{th} parallel stage we get a service distribution with coefficient of variation larger than 1.



The coefficient of variation is given by

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

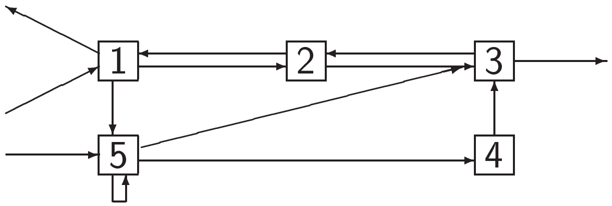
Queueing Networks

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

Allow customers to arrive from an external source, with rate γ

Customers may leave the network on completion

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

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

Customers arrive as a Poisson stream at node i at rate γ_i

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

Note

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

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 γ_i, μ_i and Q is called a **Jackson network**

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

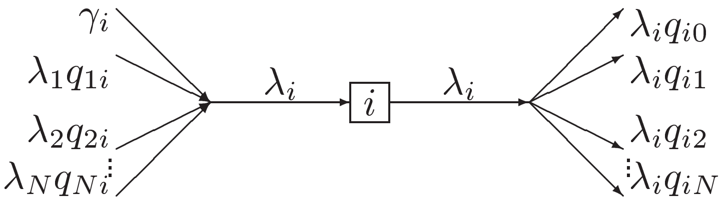
Steady state solution

Let λ_j be the average number of arrivals to node j per unit time

On average the departure count per unit time is therefore λ_j

A function q_{ji} go to node i

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



Jackson's theorem

Adding together all contributions,

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

These are known as the *traffic equations*

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

$$\rho_i \triangleq \frac{\lambda_i}{\mu_i} < 1$$

where λ_i is the solution of the traffic equations.

Jackson's theorem (2)

Let $p(k_1, k_2, \dots, k_N)$ be the steady state equilibrium distribution. Then *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 ρ_i.

Jackson's theorem has some important implications.

1. The numbers of jobs at the various nodes are independent.

2. Node i behaves as if subjected to a Poisson stream with rate λ_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

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

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

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

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 $s = (k_1, k_2, \ldots, k_n)$ 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