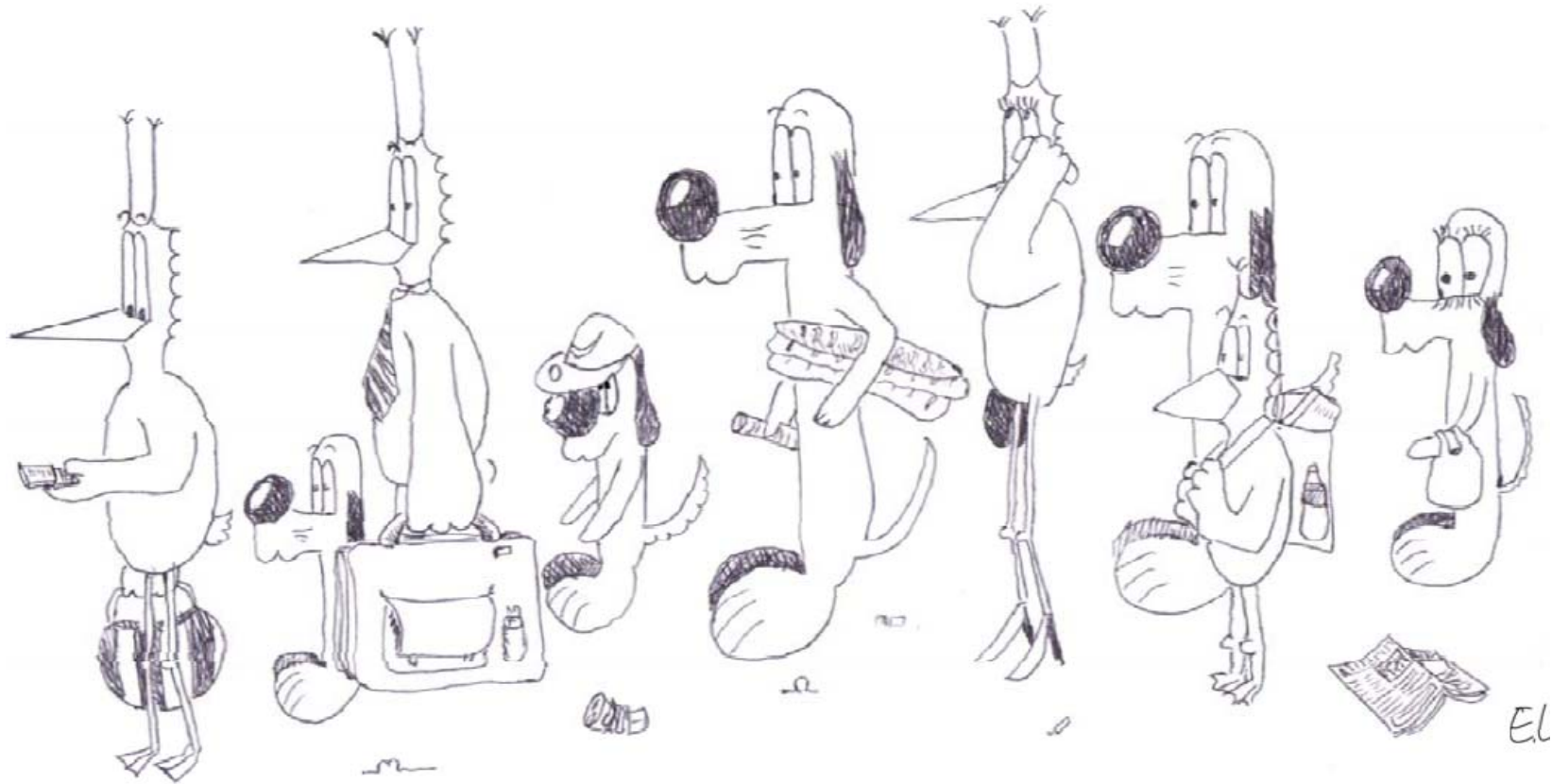


Queuing Theory

More Than a Primer



All You Need to Know About Queuing Theory

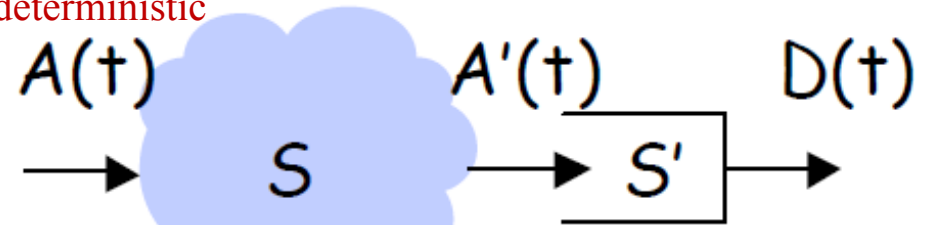
- Queuing is essential to understanding the behaviour of complex computer and communication systems
- In depth analysis of queuing systems is hard
- Fortunately, the most important results are easy

- We will study only simple concepts

1. Deterministic Queuing

- Easy but powerful
- Applies to deterministic and transient analysis
- Example: playback buffer sizing

It is quite plausibly
deterministic



Internet incurs
delay variation

EXAMPLE 8.1: **PLAYOUT BUFFER.** Consider a packet switched network that carries bits of information from a source with a constant bit rate r (Figure 8.2) as is the case for example, with circuit emulation. We have a first system S , the network, with input function $A(t) = rt$. The network imposes some variable delay, because of queuing points, therefore the output $A'()$ does not have a constant rate r . What can be done to re-create a constant bit stream? A standard mechanism is to smooth the delay variation in a playout buffer. It operates as follows. When the first bit of data arrives, at time $d(0)$, it is stored in the buffer until some initial delay has elapsed. Then the buffer is served at a constant rate r whenever it is not empty. This gives us a second system S' , with input $A'()$ and output $D()$. What initial delay should we take?

Use of Cumulative Functions

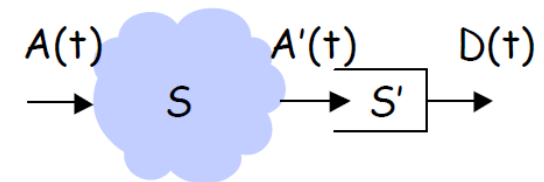
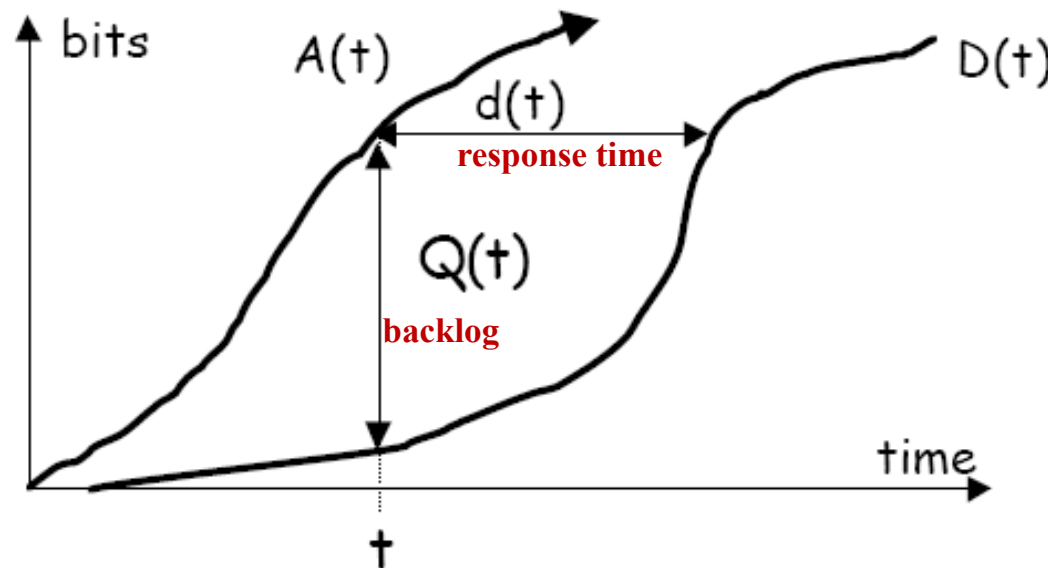
integral of the
arrival process

- $A(t)$ *input function* is the amount of work that arrives into the system in the time interval $[0, t]$
- $D(t)$ *output function* is the amount of work done in the time interval $[0, t]$
- $Q(t) := A(t) - D(t)$ is the backlog (unfinished work) at time t . **Undone work**
- Assume that there is some time $t_0 \leq 0$ at which $A(t_0) = D(t_0) = 0$. We interpret t_0 as an instant at which the system is empty.
- Let $Q(t) := A(t) - D(t)$; we interpret $Q(t)$ as the backlog (unfinished work) at time t .
- There is no loss of work.

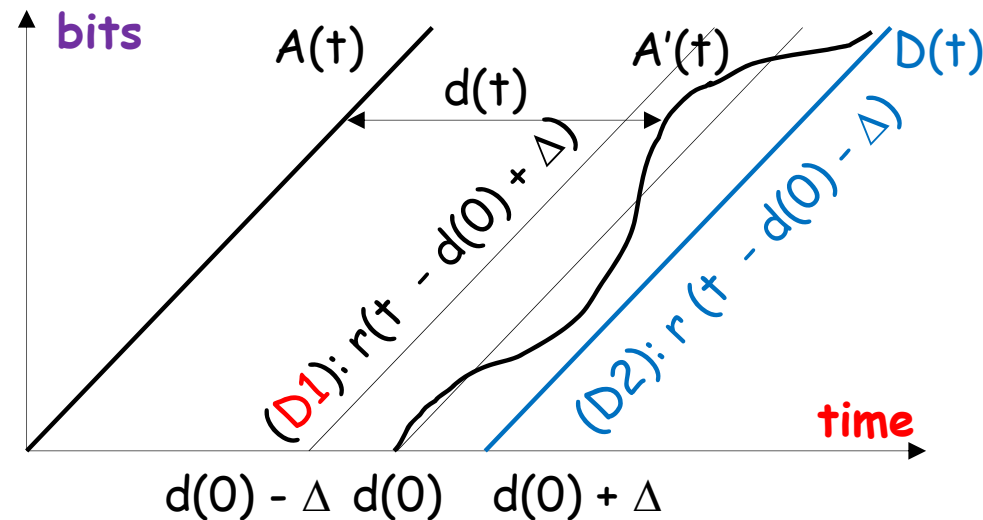
Also define

$$d(t) = \min \{u \geq 0 : A(t) \leq D(t + u)\}$$

The FIFO assumption means that $d(t)$ is the response time for a hypothetical atom of work that would arrive at time t .



Solution of Playback Delay Pb



around $d(0)$

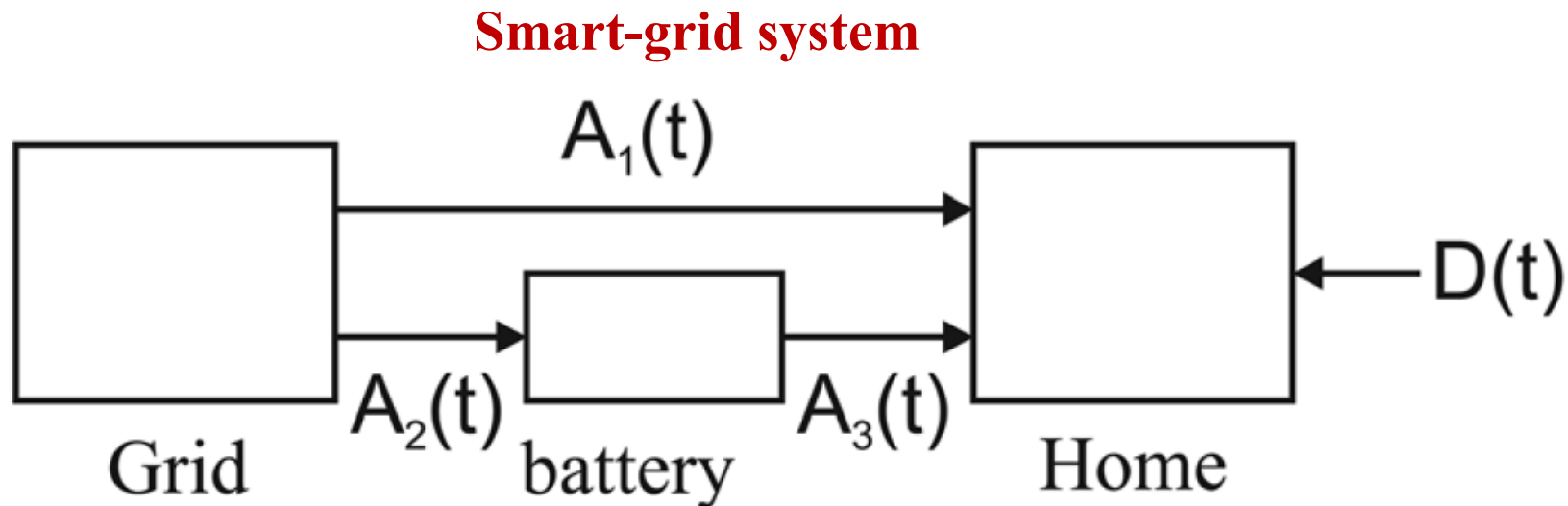
The second part of Figure 8.2 shows that if the variable part of the network delay (called *delay jitter*) is bounded by some number Δ , then the output $A'(t)$ is bounded by the two lines (D1) and (D2). Let the output $D(t)$ of the playout buffer be the function represented by (D2), namely $D(t) = r(t - d(0) - \Delta)$. This means that we read data from the playout buffer at a constant rate r , starting at time $d(0) + \Delta$. The fact that $A'(t)$ lies above (D2) means that there is never underflow. Thus the playout buffer should delay the first bit of data by an amount equal to Δ , a bound on delay jitter.

QUESTION 8.1.1. *What is the required playout buffer size ?*¹

A. $2r\Delta$ **worst case:** $A'(t)$ suddenly increases by $2r\Delta$.

Application of Deterministic Queuing

- Systems necessitating deterministic guarantees
- System dominated by deterministic elements with possible stochasticity



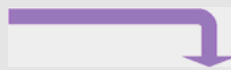
- Electricity outage is a **catastrophe!**
 - ▶ Must: to provide a (hard) **deterministic guarantee** for $A_1(t) + A_3(t) \geq D(t)$.
 - ▶ The best parts of **processes are deterministic** with sizable stochastic elements.

2. Operational Laws

THEOREM 8.2.2 (Operational Law). Consider a stationary system that is visited by a flow of customers (for a formal definition, see Theorem 7.3.4).

- [Little]

event-based



$$\lambda \bar{R} = \bar{N}$$

time-based



where λ is the expected number of customers arriving per second, \bar{R} is the expected response time seen by an arbitrary customer and \bar{N} is the expected number of customers observed in the system ^{at} an arbitrary time

■ Intuition: **Likening the response time of each customer to the payment**

- ▶ Say every customer pays one SEK per minute present: *time=money*
- ▶ Payoff per customer = R (SEK)
- ▶ Rate at which we receive money = N (SEK/min)
- ▶ In average λ (/min) customers arrive per minute, $\therefore N = \lambda \times R$

Littleteness of Little in the vast sea of Palm

- Little's Law is a *tiny* subset of Palm Calculus

Published in 1983

Miyazawa's **Rate Conservation Law** (ext. for Càdlàg)

Palm Inversion Formula

Campbell's Formula
(lemma for **dispersed arrival load**)

Little's Law
(lemma for the case **load is no. of customers**)

- Which is why we first treat Little's Formula separately
- Grasping these theorems will be enormously beneficial for your research
 - ▶ Rather than too well-known boring formulae of M/G/1

Little Again:

Most Abstract Sketch of Proof of Little's Law

- Consider a simulation where you measure R and N . You use two counters `responseTimeCtr` and `queueLengthCtr`. At end of simulation, estimate

$$R = \text{responseTimeCtr} / \text{NbCust}$$

$$N = \text{queueLengthCtr} / T$$

where `NbCust` = number of customers served and `T`=simulation duration.

- Initially, both `responseTimeCtr`=0 and `queueLengthCtr`=0.
- Q: When an arrival or departure event occurs, how are both counters updated?

A: `queueLengthCtr` += $(t_{\text{new}} - t_{\text{old}}) \times q(t_{\text{old}})$ where $q(t_{\text{old}})$ is the number of customers in queue just before the event.

$$\text{responseTimeCtr} += (t_{\text{new}} - t_{\text{old}}) \times q(t_{\text{old}})$$

thus `responseTimeCtr` == `queueLengthCtr` and thus

$$N = R \times \text{NbCust} / T ;$$

- $\therefore \underline{\text{NbCust}/T}$ is our estimator of λ

Coming up with this expression is very roughly equivalent to proving the theorems : Campbell's Formula, Little's Law


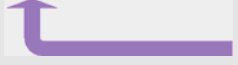
Other Operational Laws

λ : throughput as well as arrival rate

- **[Throughput]** The throughput, defined as the expected number of arrivals per second, is also equal to the inverse of the expected time between arrivals.
- **[Utilization Law]** If the system is a single server queue with arrival rate λ and expected service time \bar{S} :

If it is a s -server queue:

$$\mathbb{P}(\text{server busy}) = \rho := \lambda \bar{S}$$

time-based  event-based 

$$\mathbb{E}(\text{number of busy servers}) = s\rho$$

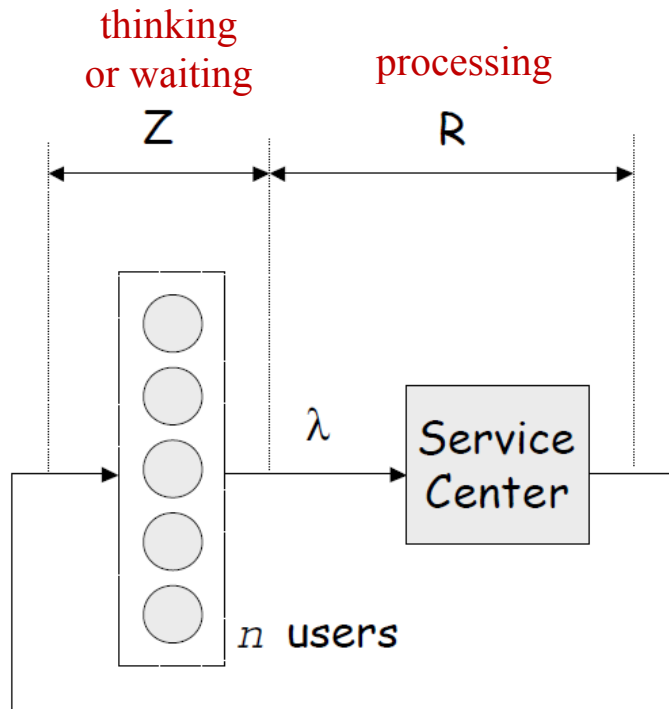
with $\rho := \frac{\lambda \bar{S}}{s}$.

Be astute: Utilization Law is nothing but an immediate consequence of Little's Law!

λ (arrival rate) \cdot \bar{S} (exp. response time during service) $:=$ ρ (exp. no. of customers in service)

Little's Law covers all stationary systems

:The Interactive User Model



THEOREM 8.2.3 (Interactive User).

$$\lambda(\bar{Z} + \bar{R}) = n$$

EXAMPLE 8.4: **SERVICE DESK**. A car rental company in a large airport has 10 service attendants. Every attendant prepares transactions on its PC and, once completed, send them to the database server. The software monitor finds the following averages: one transaction every 5 seconds, response time = 2 s.

What is the average think time?

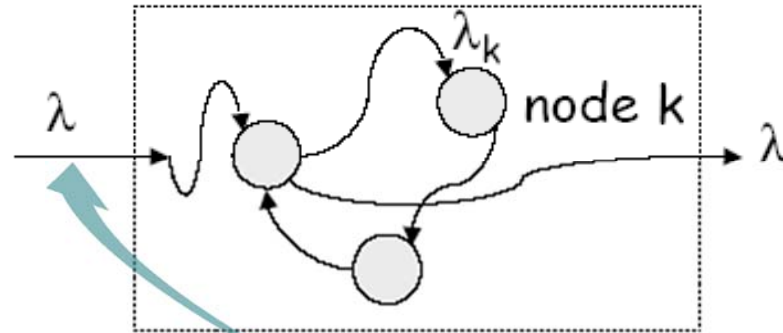
48 seconds

λ

Why?

$$\frac{1}{5 \text{ sec}} \times (\bar{Z} + 2 \text{ sec}) = 10$$

Network Laws



How customers are split?

- **[Forced Flows]** $\lambda_k = \lambda V_k$, where λ_k is the expected number of customers arriving per second at node k and V_k is the expected number of visits to node k by an arbitrary customer during its stay in the network. *per customer*
- **[Total Response Time]** Let \bar{R} [resp. \bar{R}_k] be the expected total response time \bar{R} seen by an arbitrary customer [resp. by an arbitrary visit to node k].

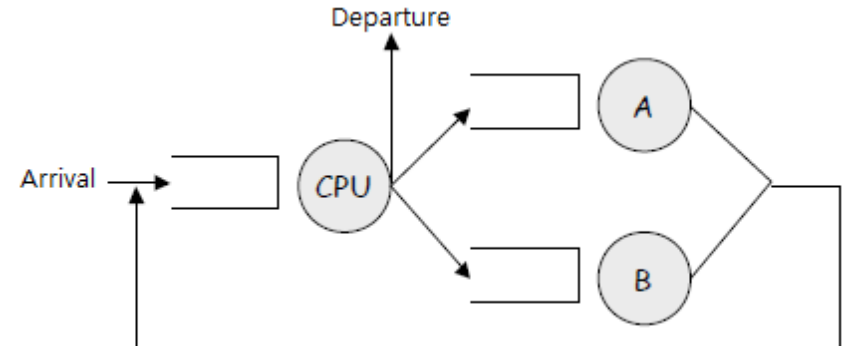
Total response time in terms of response times at nodes

$$\bar{R} = \sum_k \bar{R}_k V_k$$

EXAMPLE 8.5: Transactions on a database server access the CPU, disk A and disk B (Figure 8.5). The statistics are: $V_{\text{CPU}} = 102, V_A = 30, V_B = 68$ and $\bar{R}_{\text{CPU}} = 0.192 \text{ s}, \bar{R}_A = 0.101 \text{ s}, \bar{R}_B = 0.016 \text{ s}$

The average response time for a transaction is 23.7 s.

$$\therefore \bar{R} = 0.192 \times 102 + 0.101 \times 30 + 0.016 \times 68 = 23.702$$



Bottleneck Analysis

: Useful for Queueing Networks

- Apply the following two bounds
 1. waiting time is ≥ 0
 2. a server utilization is bounded by 1

□ Example

$$\lambda = \frac{n}{Z + \sum_k V_k \bar{R}_k}$$

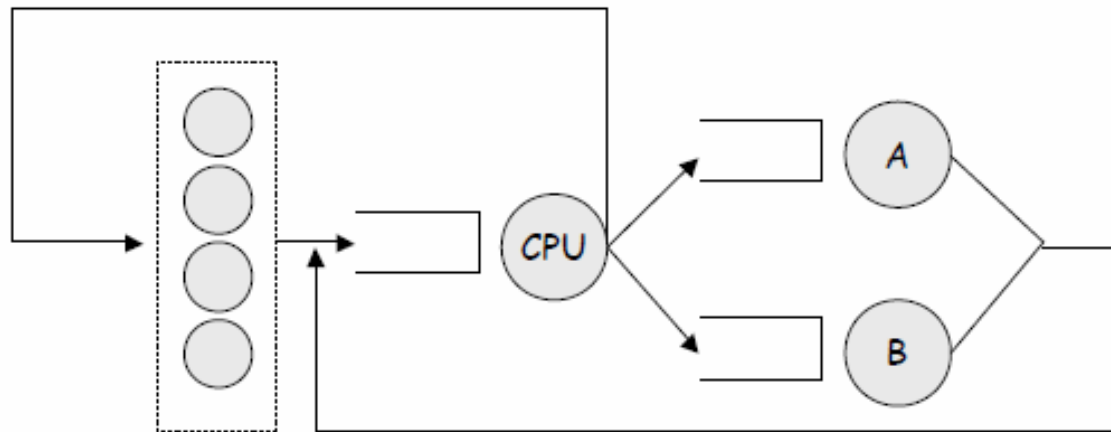
Little's Law to the entire network

$$(1) \quad \lambda \leq \frac{n}{Z + \sum_k V_k S_k}$$

$$\rho_k = \lambda V_k \bar{S}_k$$

Little's Law to each server

$$(2) \quad \lambda \leq \frac{1}{\max_k V_k \bar{S}_k}$$



n users
in think time

Figure 8.5: Network example used to illustrate bottleneck analysis. n attendants serve customers. Each transaction uses CPU, disk A or disk B. Av. numbers of visits per transaction: $V_{\text{CPU}} = 102$, $V_A = 30$, $V_B = 17$; av. service time per transaction: $\bar{S}_{\text{CPU}} = 0.004 s$, $\bar{S}_A = 0.011 s$, $\bar{S}_B = 0.013 s$; think time $Z = 1 s$.

Throughput Bounds

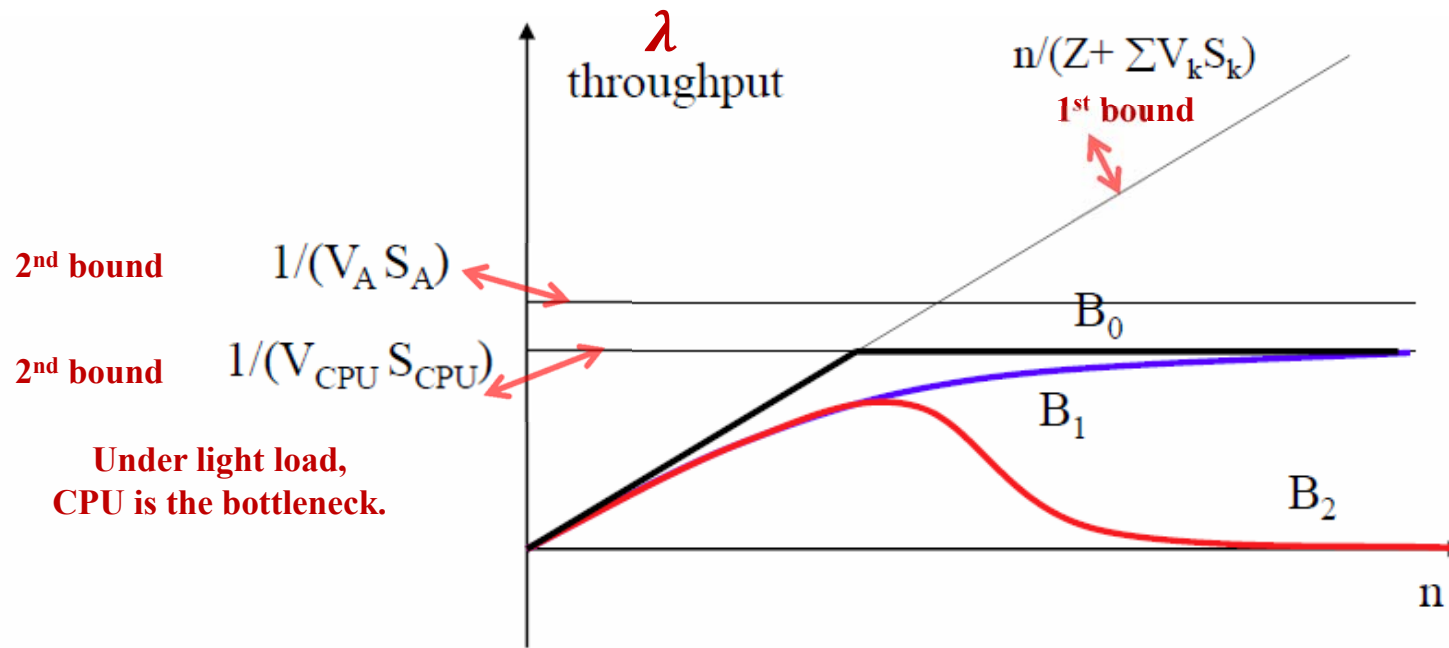


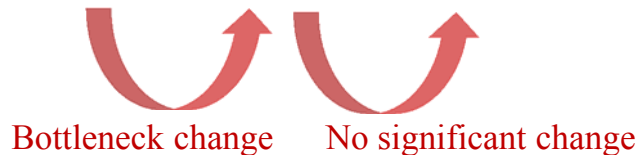
Figure 8.6: Throughput bound (B_0) obtained by bottleneck analysis for the system in Figure 8.5, as a function of the number of users n . B_1 , B_2 : typical throughput values for a system without [resp. with] congestion collapse.

Bottlenecks

or minimizes $1/V_k\bar{S}_k$

A node k that maximizes $V_k\bar{S}_k$ is called, in this model, a *bottleneck*. To see why a bottleneck determines the performance, consider improving the system by decreasing the value of $V_k\bar{S}_k$ (by reducing the number of times the resource is used, or by replacing the resource by a faster one). If k is not a bottleneck, this does not affect asymptote on Figure 8.6, and only marginally increases the slope of the bound at the origin, unlike if k is a bottleneck. On Figure 8.6, we see that the bottleneck is the CPU.

QUESTION 8.2.2. *What happens to the example of Figure 8.5 if the CPU processing time is reduced from 0.004 to 0.003 ? to 0.002 ?*⁴



⁴The disk A becomes the bottleneck. Decreasing the CPU processing time to 0.002 does not improve the bound significantly.

DASSA

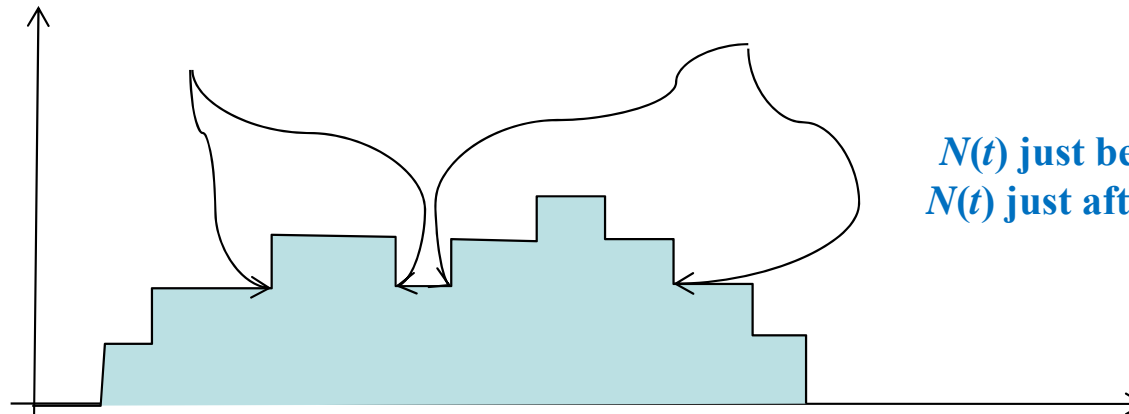
8.2.1 DEPARTURES AND ARRIVALS SEE SAME AVERAGES (DASSA)

n-th inter-arrival time

n-th inter-departure time

THEOREM 8.2.1. (DASSA) Consider a system where individual customers come in and out. Assume that the arrival process A_n and the departure process D_n are stationary point processes, and that they have no point in common (thus there are no simultaneous arrivals or departures). Let $N(t) \in \mathbb{N}$ be the number of customers present in the system at time t . Assume that $N(t)$, A_n and D_n are jointly stationary (see Section 7.2). Then the probability distribution of $N(t)$ sampled just before an arrival is equal to the probability distribution of $N(t)$ sampled just after a departure.

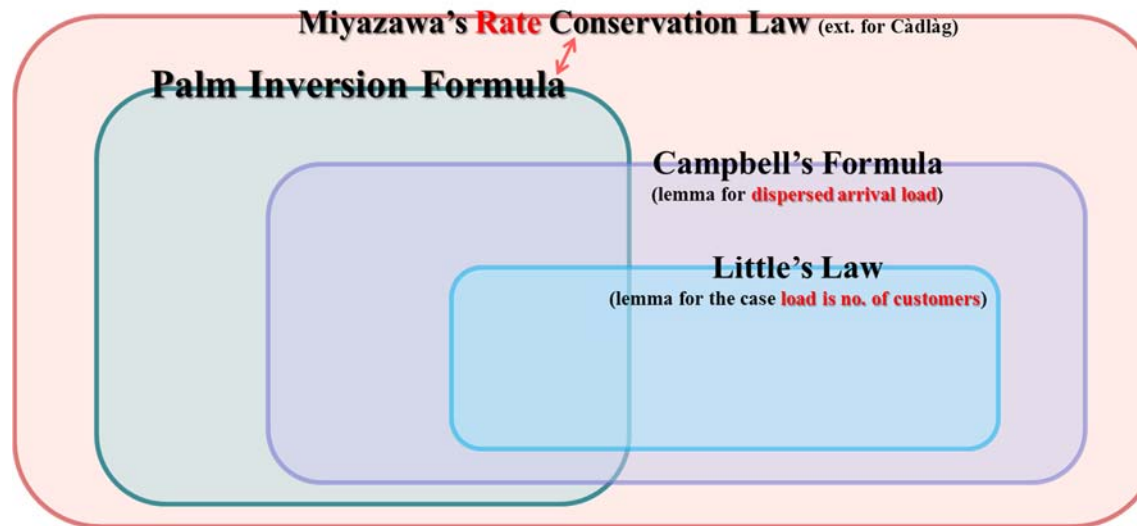
- **Intuition:** within one busy period, to every departure, we can associate one arrival with same number of customers left behind.



Example

$N(t)$ just before arrivals: 0,1,2,2,3
 $N(t)$ just after departures: 2,3,2,1,0

Recapitulation



- Little's Law **permeates through** several key techniques
 - ▶ Applicable to **all** imaginable **stationary** systems
 - ▶ Utilization Law
 - ▶ Bottleneck Analysis
 - ▶ See Theorem 7.3.4 to appreciate its generality
- Combination of DASSA and PASTA (to be discussed later)
 - ▶ Oftentimes provides best part of necessary **insights for analysis**
 - ▶ See Example 8.3

3. Single Server Queue

Kendall's Notation

The classical notation for a queue, in its simplest form, is of the type $A/S/B/K$ where:

- A (character string) describes the type of arrival process: G stands for the most general arrival process, $A = GI$ means that the arrival process is a point process with iid interarrival times, M is for a Poisson arrival process.
- S (character string) describes the type of service process: G for the most general service process, $S = GI$ means that the service times are iid and independent of the arrival process, $S = M$ is the special case of GI with exponential service times, $S = D$ with constant service times.
- B and K are integers representing the number of servers and the capacity (maximum number of customers allowed in the system, queued + in service). When $K = \infty$, it may be omitted.
- Let A_n be the arrival time and S_n the service time of the n th customer, labeled in order of arrival. We assume that the sequence (A_n, S_n) is stationary with respect to the index n and that it can be interpreted as a stationary marked point process (i.e. the expectation of $A_{n+1} - A_n$ is finite, see Theorem 7.4.1).
- The service discipline is by default FIFO, otherwise it is mentioned explicitly.

Stationarity of Single Server Queue

THEOREM 8.3.1. (Loynes [3, Thm 2.1.1])

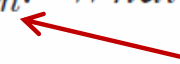
If $\rho < 1$ the backlog process has a unique stationary regime. In the stationary regime, the queue empties infinitely often.

Furthermore, for any initial condition, the waiting time of the n th customer converges in distribution as $n \rightarrow \infty$ to the waiting time for an arbitrary customer computed in the stationary regime.

If $\rho > 1$ the backlog process has no stationary regime.

Well, single server queue is expectably stationary for $\rho < 1$.

QUESTION 8.3.1. Consider a queuing system of the form $G/G/1$ where the service time S_n of customer n is equal to the inter-arrival time $A_{n+1} - A_n$. What are the values of ρ and of the expected number of customers \bar{N} ? ⁵



arrival time of n -th customer

⁵ $\lambda = \frac{1}{S}$ thus $\rho = 1$. There is always exactly one customer in the queue. Thus $\bar{N} = 1$.

M/GI/1 QUEUE Stability is for $\rho < 1$

Avg. no. of customers in system

Avg. no. of customers in waiting room

Avg. response time

Avg. waiting time

$$\left\{ \begin{array}{l} \bar{N} = \frac{\rho^2 \kappa}{1-\rho} + \rho \text{ with } \kappa = \frac{1}{2} \left(1 + \frac{\sigma_S^2}{S^2} \right) \\ \bar{N}_w = \frac{\rho^2 \kappa}{1-\rho} \\ \bar{R} = \frac{\bar{S}(1-\rho(1-\kappa))}{1-\rho} \\ \bar{W} = \frac{\rho \bar{S} \kappa}{1-\rho} \end{array} \right. \quad \text{CoV}^2$$

time-based

event-based

Stability is for $\rho < 1$ for all the examples below.

$$\bar{R} = \frac{\bar{S}(1 - \rho(1 - \kappa))}{1 - \rho} = \frac{\bar{S}(1 - \rho + \rho\kappa)}{1 - \rho} = \frac{\rho \bar{S} \kappa}{1 - \rho} + \bar{S} = \bar{W} + \bar{S}$$

QUESTION 8.3.4. Which of the quantities \bar{N} , \bar{N}_w , \bar{R} , \bar{W} are Palm expectations? ⁹

⁹ R, W

mnemonic:

customer \rightarrow time-based

time \rightarrow event-based (**Palm**)

M/M/1 QUEUE Stability is for $\rho < 1$.

$$\left\{ \begin{array}{l} \bar{N} = \frac{\rho}{1-\rho} \\ \bar{N}_w = \frac{\rho^2}{1-\rho} \\ \bar{R} = \frac{\bar{S}}{1-\rho} \\ \bar{W} = \frac{\rho \bar{S}}{1-\rho} \\ \sigma_N = \frac{\sqrt{\rho}}{1-\rho} \\ \sigma_R = \frac{\bar{S}}{1-\rho} \\ \mathbb{P}(N = k) = (1-\rho)\rho^k \\ \mathbb{P}^0(R \leq x) = 1 - e^{-(1-\rho)\frac{x}{\bar{S}}} \end{array} \right.$$

M/M/1/K QUEUE Stability is for any ρ .

$$\left\{ \begin{array}{l} \mathbb{P}(N = k) = \eta(1-\rho)\rho^k 1_{\{0 \leq k \leq K\}} \\ \eta = \frac{1}{1-\rho^{K+1}} \\ \mathbb{P}^0(\text{arriving customer is discarded}) = \mathbb{P}(N = K) \end{array} \right.$$

max. queue size is K

M/D/1 QUEUE Stability is for $\rho < 1$.

$$\left\{ \begin{array}{l} \bar{N} = \frac{\rho^2}{2(1-\rho)} + \rho \\ \bar{N}_w = \frac{\rho^2}{2(1-\rho)} \\ \bar{R} = \frac{\bar{S}(2-\rho)}{2(1-\rho)} \\ \bar{W} = \frac{\rho \bar{S}}{2(1-\rho)} \\ \sigma_N = \frac{1}{1-\rho} \sqrt{\rho - 1.5\rho^2 + \frac{5}{6}\rho^3 - \frac{1}{12}\rho^4} \\ \sigma_R = \frac{\bar{S}}{1-\rho} \sqrt{\frac{1}{3}\rho - \frac{1}{12}\rho^2} \end{array} \right.$$

Multiple Server Queue

no. of servers is s

M/M/s QUEUE Stability is for $\rho < 1$. Let

$$u = \frac{\sum_{i=0}^{s-1} \frac{(s\rho)^i}{i!}}{\sum_{i=0}^s \frac{(s\rho)^i}{i!}} \text{ and } p = \frac{1-u}{1-\rho u}$$

$$\left\{ \begin{array}{l} \bar{N} = \frac{p\rho}{1-\rho} + s\rho \\ \bar{N}_w = \frac{p\rho}{1-\rho} \\ \bar{R} = \frac{p\bar{S}}{s(1-\rho)} + \bar{S} \\ \bar{W} = \frac{p\bar{S}}{s(1-\rho)} \\ \sigma_R = \frac{\bar{S}}{s(1-\rho)} \sqrt{p(2-p) + s^2(1-\rho)^2} \\ \sigma_W = \frac{1}{1-\rho} \sqrt{p\rho(1+\rho-p\rho)} \\ \mathbb{P}(N = k) = \begin{cases} \eta \frac{(s\rho)^k}{k!} & \text{if } 0 \leq k \leq s \\ \eta \frac{s^s \rho^k}{s!} & \text{if } k > s \end{cases} \\ \eta^{-1} = \sum_{i=0}^{s-1} \frac{(s\rho)^i}{i!} + \frac{(s\rho)^s}{s!(1-\rho)} \\ \mathbb{P}^0(W \leq x) = 1 - p e^{-s(1-\rho)\frac{x}{\bar{S}}} \\ \mathbb{P}(\text{all servers busy}) = \mathbb{P}(N \geq s) = p \text{ (Erlang-C formula)} \end{array} \right.$$

M/M/s/s QUEUE (Erlang Loss Formula) Stability is for any ρ .

$$\left\{ \begin{array}{l} \mathbb{P}(N = k) = \eta 1_{\{0 \leq k \leq s\}} \frac{(s\rho)^k}{k!} \\ \mathbb{P}^0(\text{arriving customer is discarded}) = \mathbb{P}(N = s) \text{ Erlang-B formula} \\ \eta^{-1} \text{ such that } \sum_{i=0}^s \mathbb{P}(N = i) = 1 \end{array} \right.$$

**Finite buffer
implies stability.**

Insight 1/2: Non Linearity of Response Time

In fact, response time $\propto \frac{1}{1-\rho}$

Mean Response Time in seconds

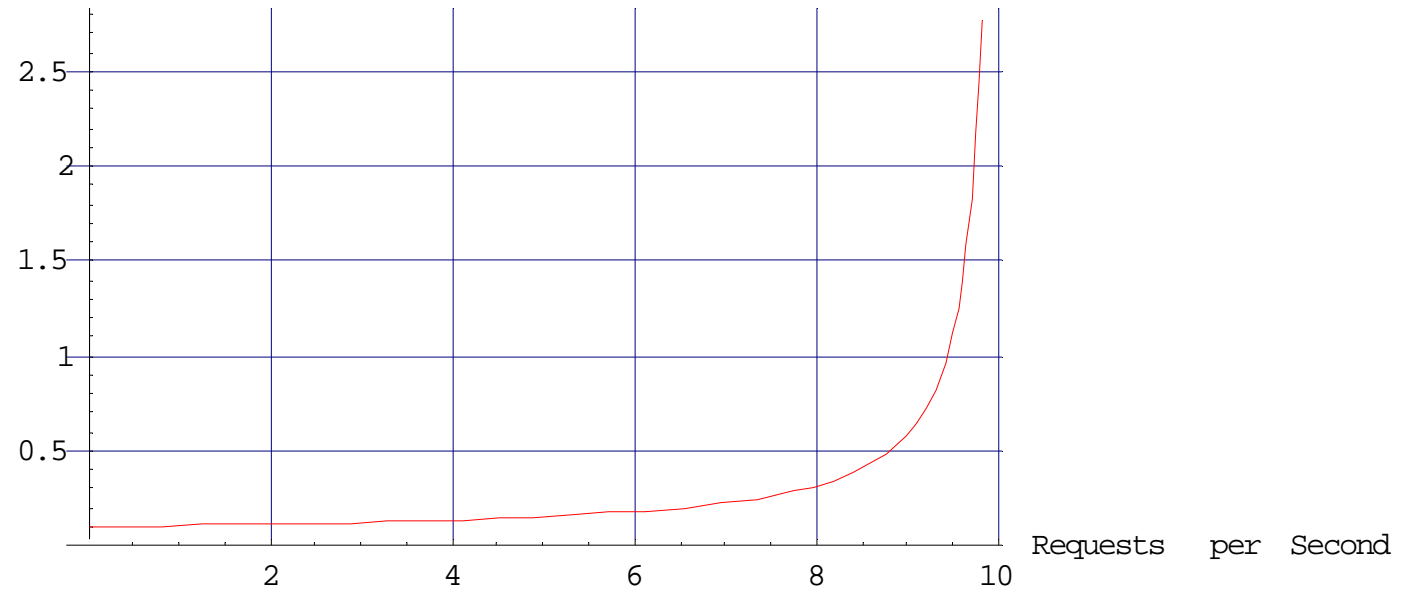


Figure 8.7: Average response time versus requests per second for a database server modeled as M/GI/1 queue. The time needed to process a request is 0.1 second and its standard deviation is estimated to 0.03. The maximum load that can be served if an average response time of 0.5 second is considered acceptable is 8.8 requests per second. If the traffic volume increases by 10%, the response time becomes 1.75, thus is multiplied by a factor of 3.5.

Insight 2/2: Impact of Variability

Mean Response Time

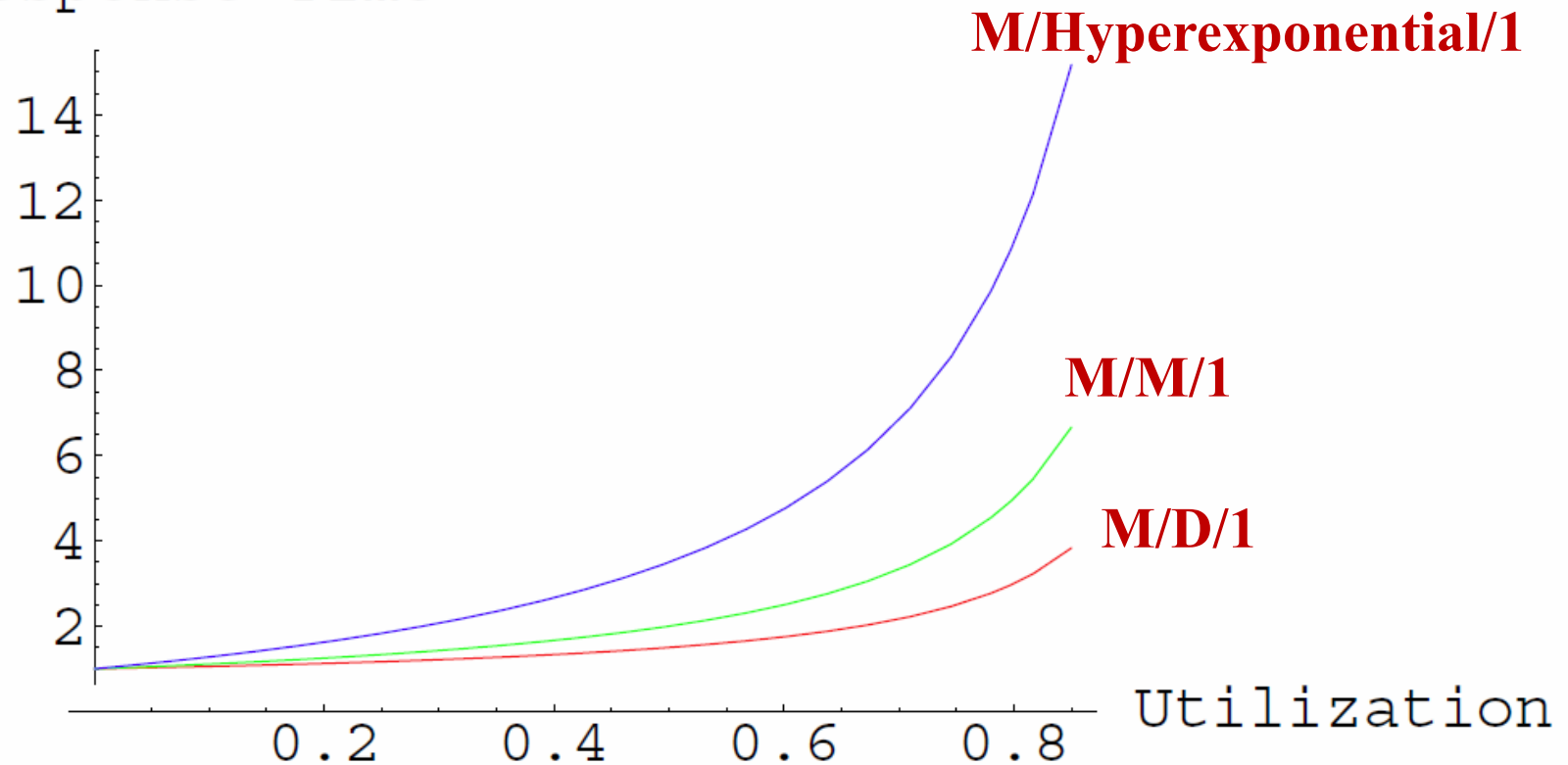


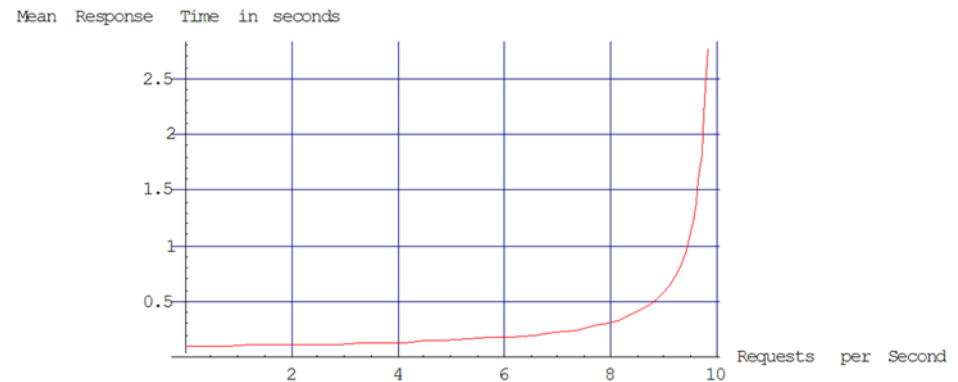
Figure 8.8: Mean response time for M/GI/1 queue, relative to service time, for different values of coefficient of variation $CoV_S = \frac{\sigma_S}{S}$: from top to bottom: $CoV_S = 1.4$, $CoV_S = 1$ (M/M/1 queue) and $CoV_S = 0$ (M/D/1 queue).

Variability of service time has an adverse effect on response time.

Confidence Interval of Delay Statistics

- **Super-linearity** is the **quintessential** property of delay performance
 - ▶ in almost all queueing networks (except closed ones)
- Delay averages in **ALL** server queues with infinite buffer so far take:

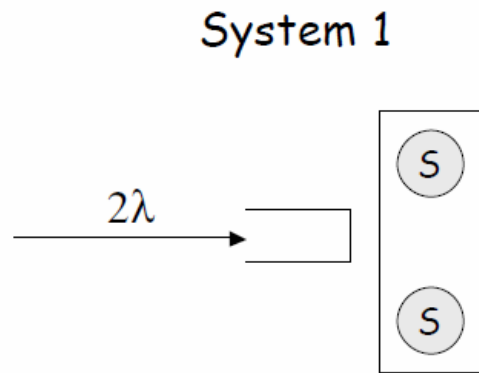
$$\bar{W} \text{ or } \bar{R} = K_1 + \frac{K_2}{1-\rho}$$



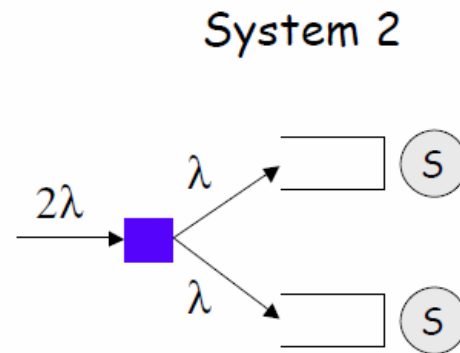
- Yet another alternative to compute confidence interval: **Take logarithm!**
 - ▶ Note: The arrival rate should be carefully chosen for stability.
- How to justify the logarithmic transformation?
 - ▶ Claim 1: By appealing to queueing theory, since **delay performance is inherently super-linear (steeper than exponential)**, the transformation is necessary to process them.
 - ▶ Claim 2: **Perceived satisfaction** level of human being with respect to delay performance is logarithmic, rightfully on which scale, you should compute confidence intervals.

Optimal Sharing

- Compare the two in terms of
 - ▶ Response time
 - ▶ Capacity

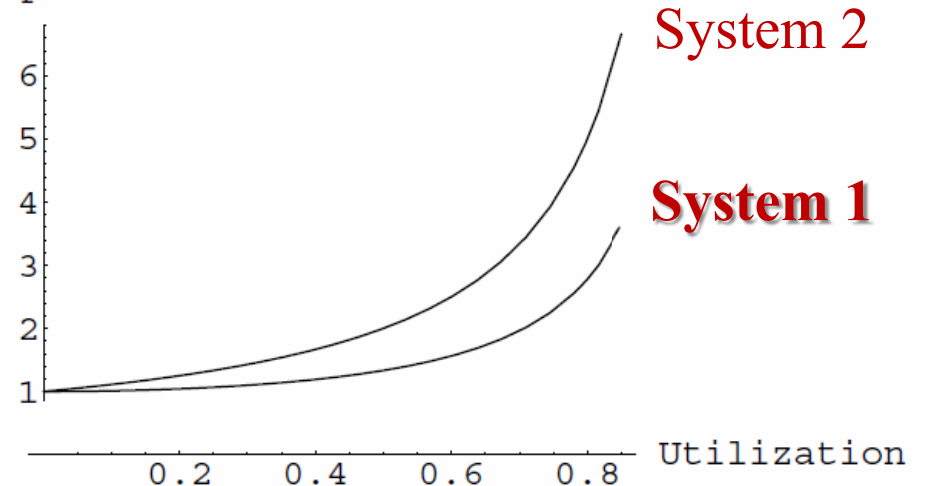


e.g., M/M/2



e.g., two parallel M/M/1
(inexpressible by Kendall's notation)

Mean Response Time



We see that for very small loads, the systems are similar, as expected. In contrast, for large loads, the response time for the first system is much better, with a ratio equal to $1 + \rho$. For example, for $\rho = 0.5$, the second system has a response time 1.5 times larger. However, the capacity is the same for both systems.

***Load sharing* leads to the decrease of response time, but the capacity remains unchanged.**

Why?: Throughput Operational Law

The Processor Sharing Queue (M/GI/1/PS)

N.B.

Optimal Sharing: *Customers* are shared by multiple servers.
Processor Sharing Queue: The *server* is shared by *all* customers.

■ Models: processors, network links

This is a special case of the single server queue, with the *Processor Sharing (PS)* service discipline instead of FIFO. Here we assume that the server divides itself equally into all present customers; this is an idealization when $\delta \rightarrow 0$ of the round robin service discipline, where the server allocates times slices of duration δ in turn to each present customer. If there are N customers in the queue, the residual service time for each of them decreases at a rate $1/N$. This is also called *egalitarian processor sharing*. Loynes's theorem applies and the system is stable when $\rho < 1$.

■ **Insensitivity**: whatever the service requirements:

$$\mathbb{P}(N(t) = k) = (1 - \rho)\rho^k \quad \text{Depends only on the mean}$$

In general M/GI/1 queues, the above expression depend not only on the means but also on the variances (i.e., CoV).

■ **Egalitarianism** (PS implies customers with large service times wait longer)

$$\mathbb{E}^0(R_0 | S_0 = x) = \frac{x}{1 - \rho} \quad \text{Large packets}$$

the average response time R_0 of an arbitrary customer, conditional to its service time S_0

packet size

PS versus FIFO

■ PS $\mathbb{E}^0(R_0 | S_0 = x) = \frac{x}{1 - \rho}$

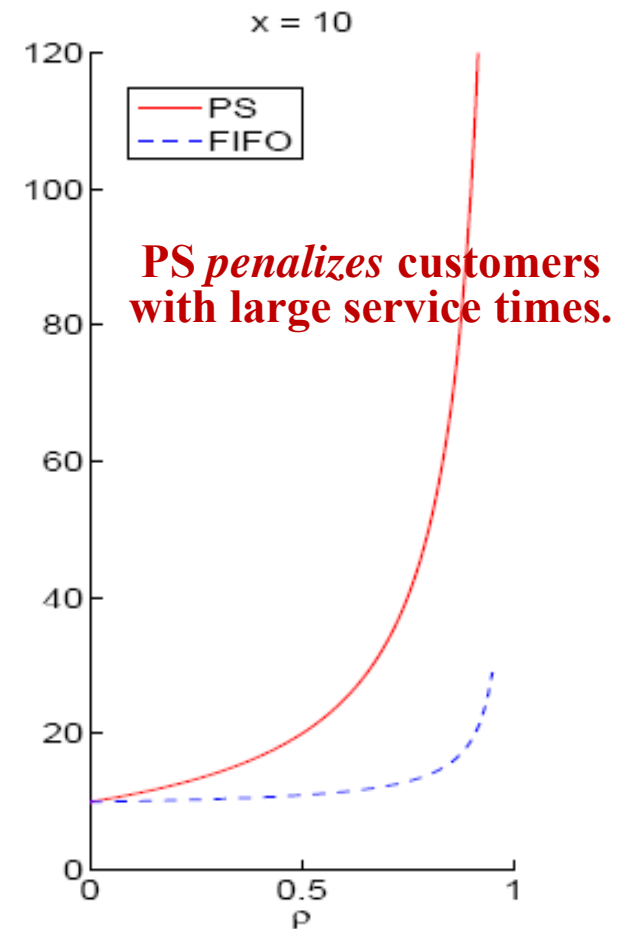
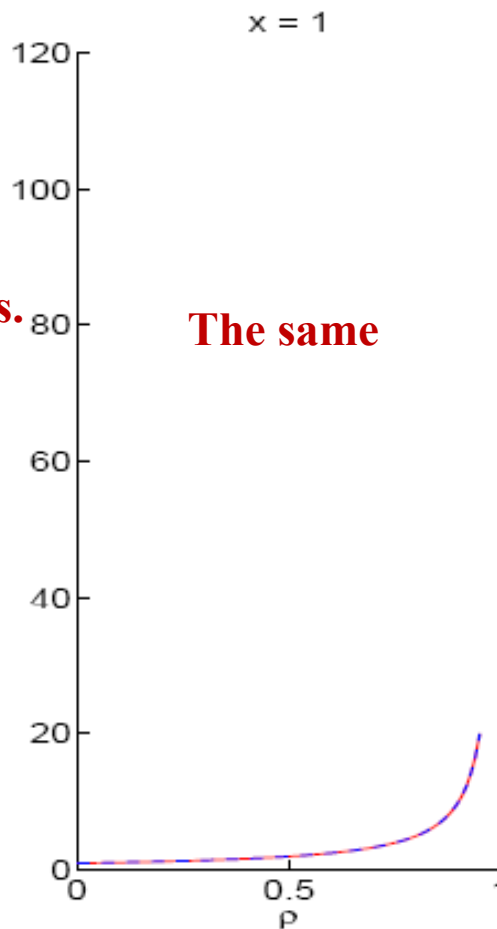
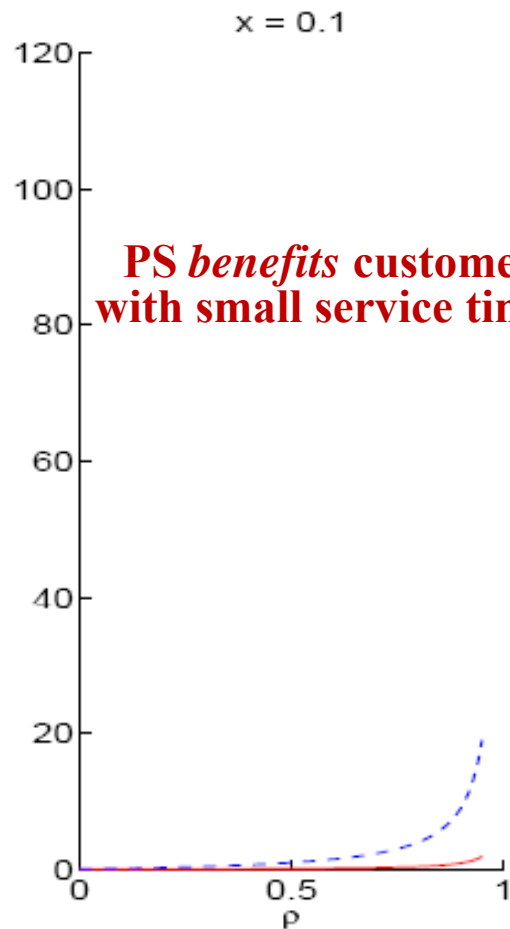
■ FIFO $\mathbb{E}^0(R_0 | S_0 = x) = x + \frac{\rho \bar{S}}{1 - \rho}$

Negligible for $\rho \approx 1$

$\rho \approx 1$



Mean Response Time

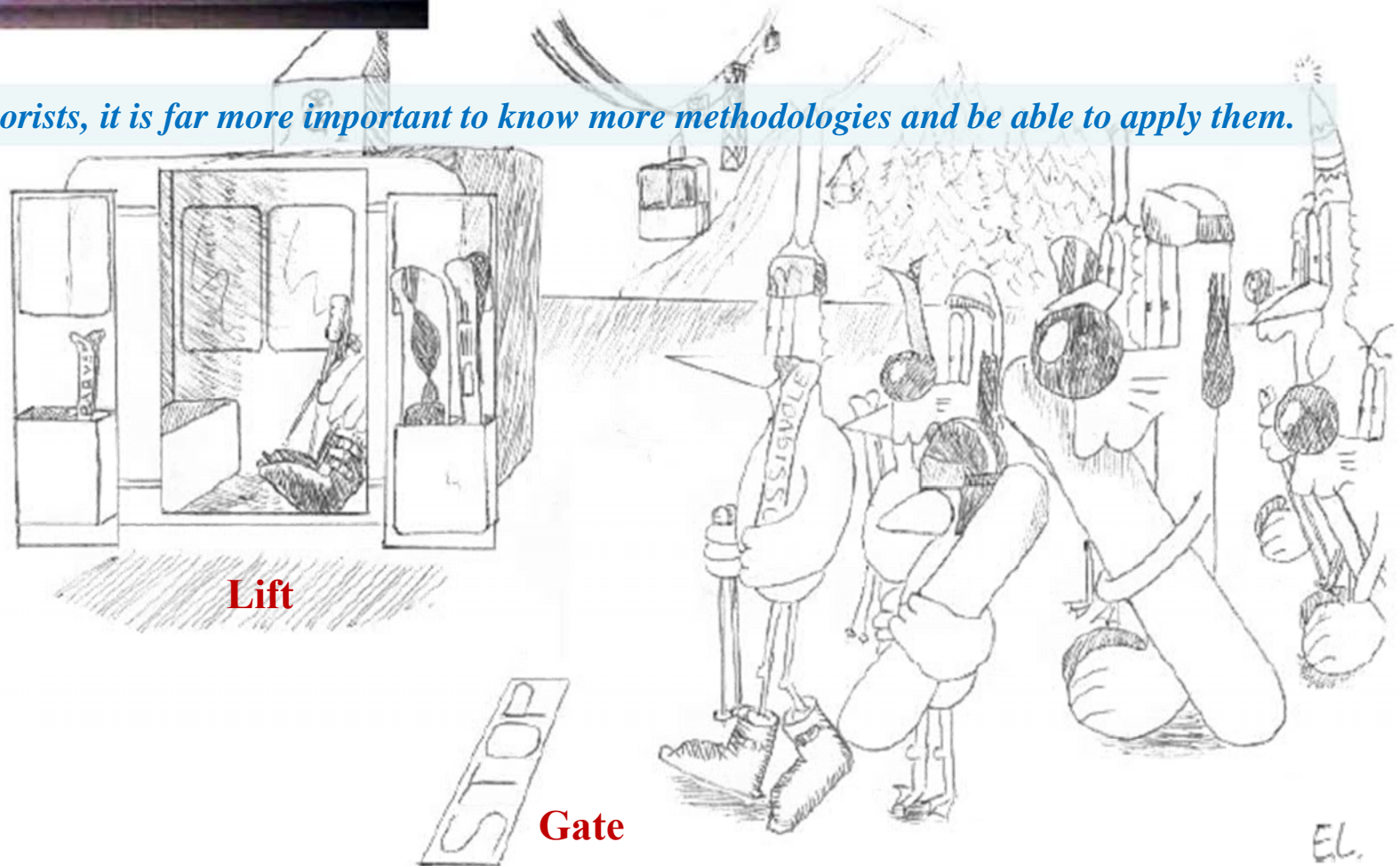




4. A Case Study

- Impact of capacity increase ?
- Optimal Capacity ?

For non queuing theorists, it is far more important to know more methodologies and be able to apply them.



EL.

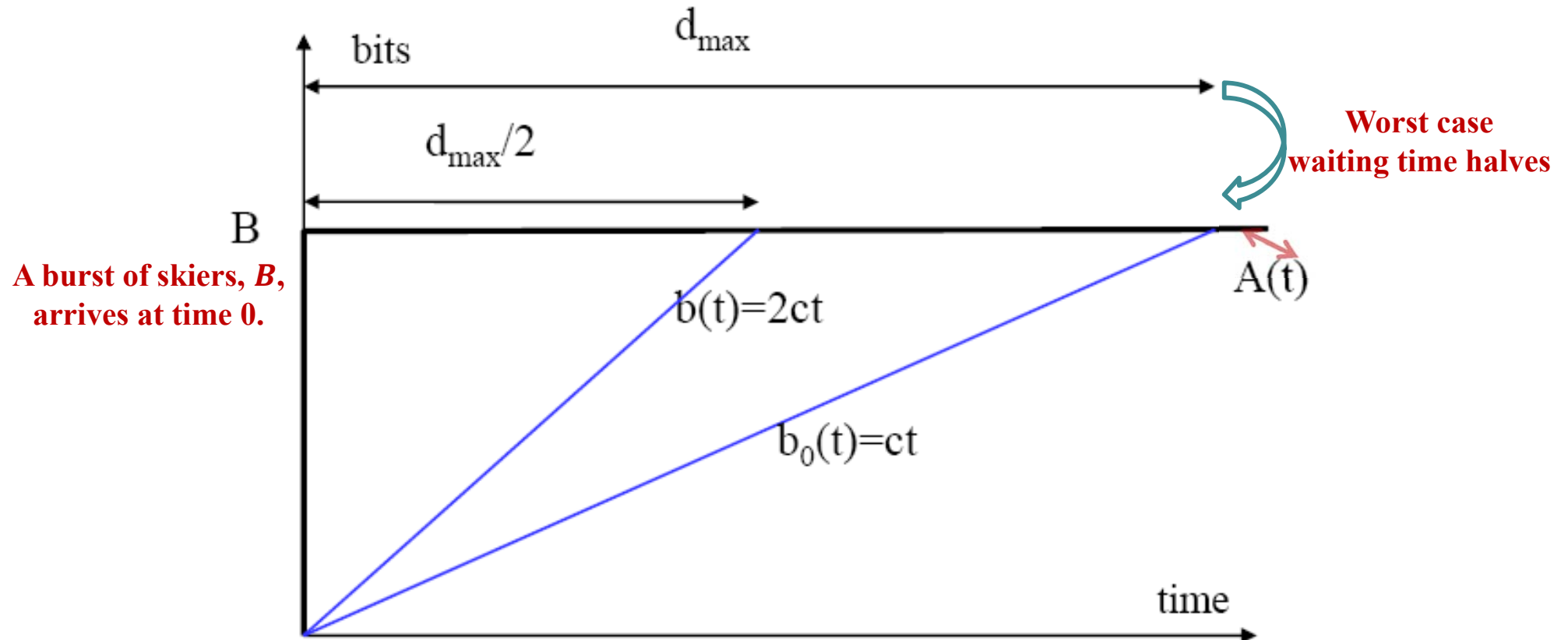
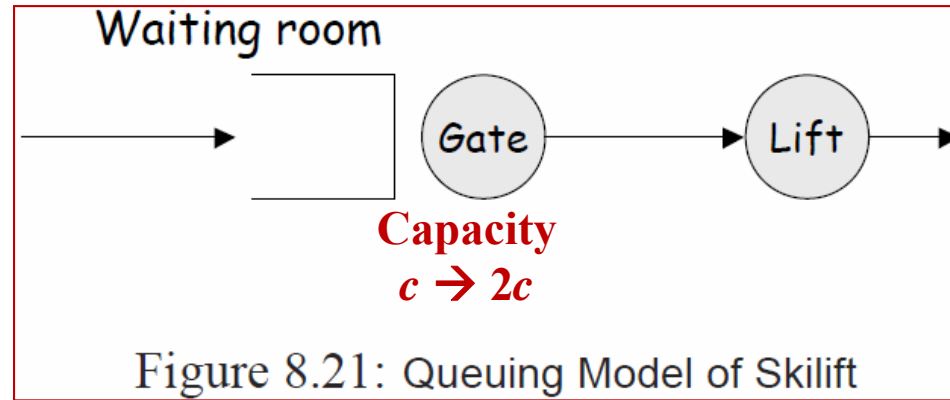
Methodology

We want to verify the advertisement of a ski resort:

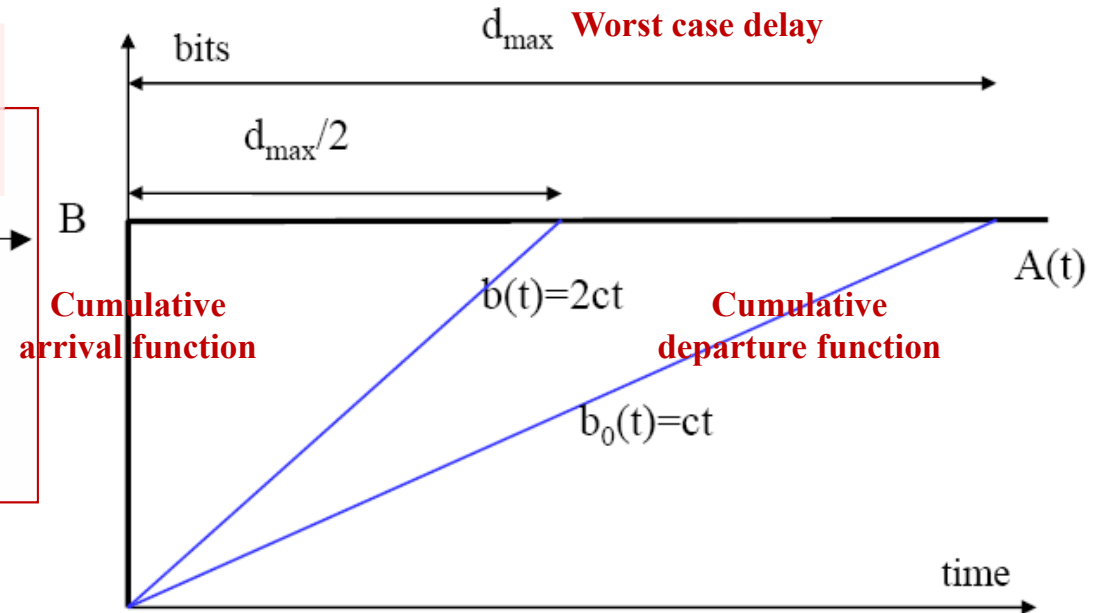
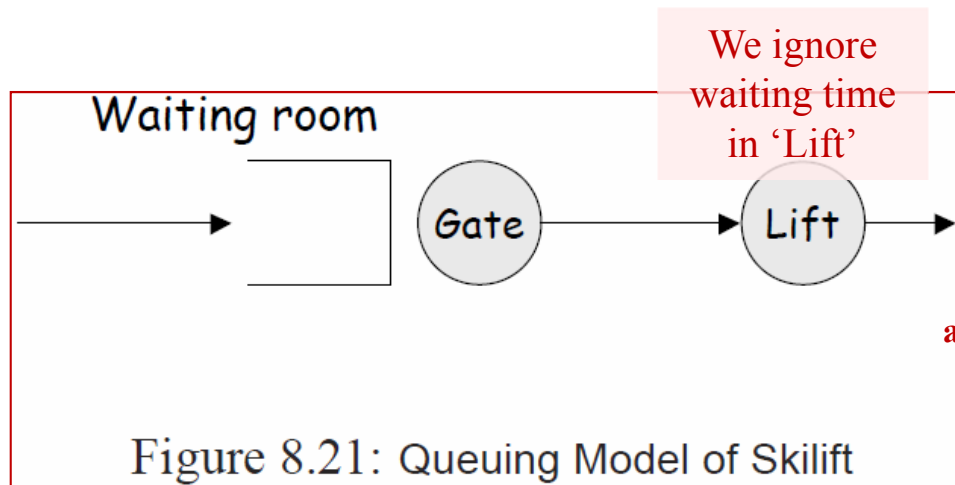
“Capacity doubled, waiting time halved!”

- Goal: evaluate impact of doubling the capacity of a skilift on the response time.
- Factors: $c = \text{capacity of skilift in people per second}$.
- Metrics: response time. A more detailed reflection leads to considering the waiting time, as this is the one that affects customer's perception.
- Load: we consider two load models : (1) heavy burst of arrival (after a train or a bus arrives at the skilift) (2) peak hour stationary regime

4.1. Deterministic Analysis



Deterministic Analysis



Burst arrivals of B at $t=0$

number of skiers that entered the skilift in $[0, t]$. Thus the delay $d(t)$ is the waiting time, excluding the time spent on the skilift. We also have $\beta(t) = ct$, with c = the capacity of the skilift, in skiers per second. We have $A(t) = B$ for $t \geq 0$. **Figure 8.22 shows that doubling the capacity does divide the worst case waiting time by two.**

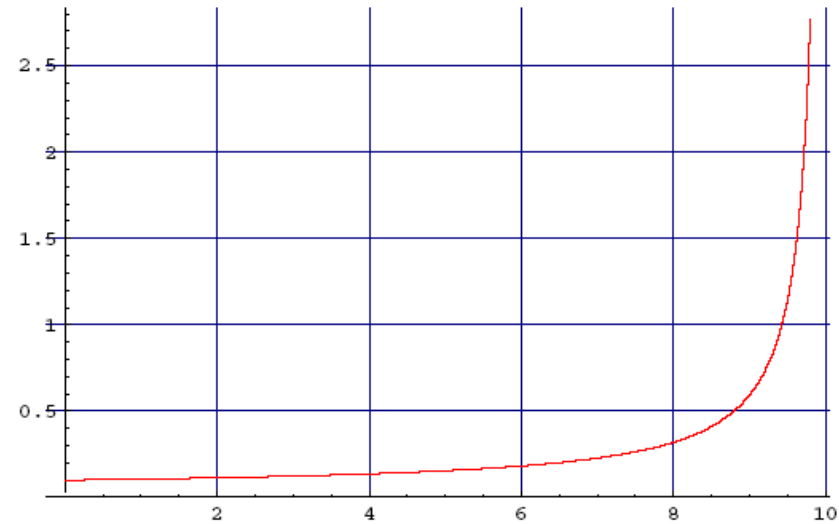
Is the average waiting time also divided by 2? To answer this question we take the viewpoint of an arbitrary customer. We see that the waiting time seen by a customer arriving as number y ($0 \leq y \leq B$) is linear in y , thus the average waiting time is equal to the worst case response time divided by a 2. Here too, **doubling the capacity divides the average waiting time by 2.**

QUESTION 8.9.1. *In reality, even if the arrival of skiers is bursty, it may not be as simultaneous as we just described. We can account for this by taking $A(t) = kct$ for $0 \leq t \leq t_0$ and $A(t) = A(t_0)$ for $t \geq t_0$, with $k \geq 1$. What is now the conclusion?* ¹⁹

Draw the function $A(t)=kct$ on the graph!

4.2 Single Queue Analysis

- Assume **no feedback** loop:



Assume now we are observing the system in the middle of the peak hour. We can model the gate as a single queue, with one or perhaps several servers. It is difficult to give a more accurate statement about the arrival process without performing actual measurements. Whatever the details, doubling the capacity halves the utilization factor ρ . A major pattern of single queue systems is the non linearity of response time, as in Figure 8.7.

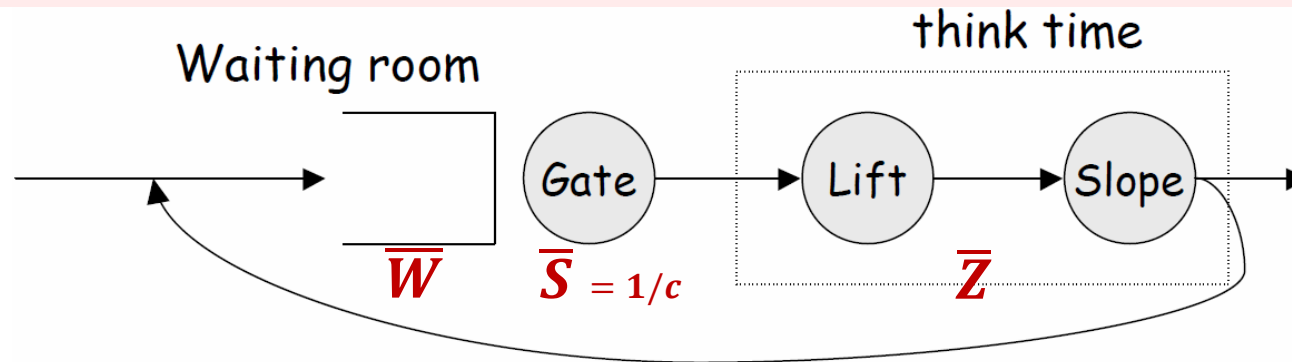
The effect on response time depends on where we stood on the curve. If the system was close to saturation, as was probably the case, the effect is a large reduction of the average waiting time, probably much larger than 2. With this model, **doubling the capacity decreases the waiting time by more than two.**

This is because of the *super-linearity* of the response time w.r.t. to the utilization factor in M/GI/1.

4.3 Operational Analysis

- A refined model, with circulating users, **with feedback**

A variant of the 'Interactive User Model'. A closed queuing network with mean number of customers \bar{N} .

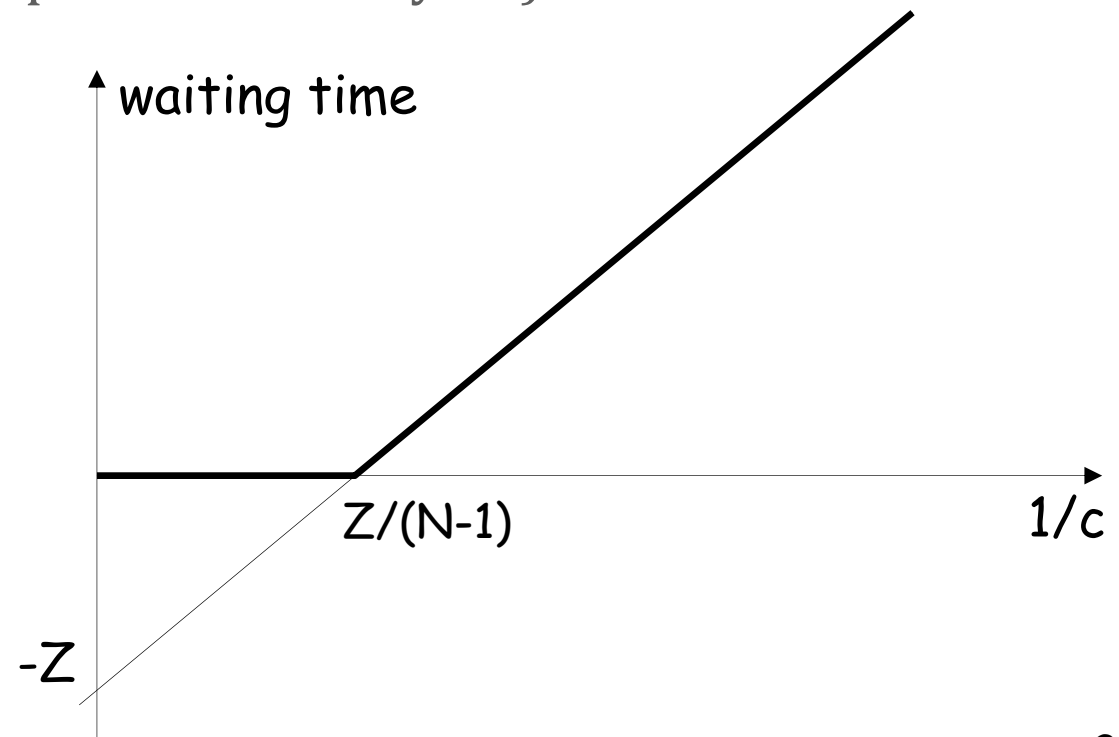


- Apply Bottleneck Analysis (= Operational Analysis)

$$\begin{cases} \lambda(\bar{W} + \bar{S} + \bar{Z}) = \bar{N} \\ \lambda \leq c \end{cases}$$

$$\bar{W} \approx \max\left(\frac{\bar{N}}{c} - \bar{Z}, 0\right)$$

$$\begin{aligned} \bar{W} + \bar{Z} &= \frac{\bar{N}}{\lambda} - \bar{S} \\ \therefore \bar{W} + \bar{Z} &\geq \frac{\bar{N} - 1}{c} \\ \therefore \bar{W} &\geq \max\left(\frac{\bar{N} - 1}{c} - \bar{Z}, 0\right) \end{aligned}$$



Cross (+) : Analytical Solution (exact solution from the queuing network model, so-called MVA)

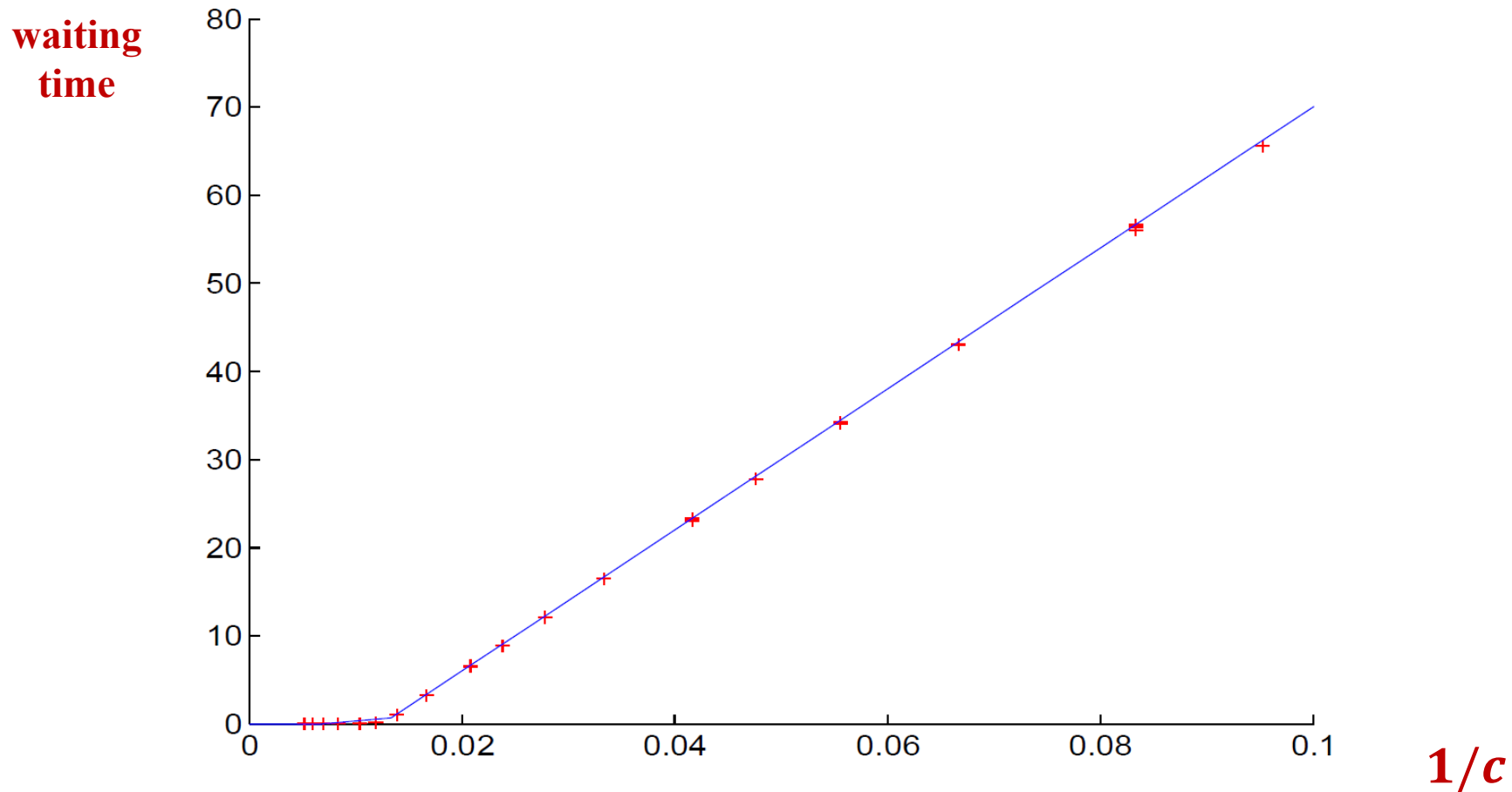


Figure 8.23: Waiting time in minutes for this model versus $\frac{1}{c}$, where c is skilift capacity (in people per minute). The solid line is the approximation by bottleneck analysis. The crosses are obtained by analytical solution of the queuing network model in Figure 8.24, with the following parameters: population size $K = 800$ skiers; number of servers at gate $B \in \{1, 2, \dots, 7, 8\}$; service time at gate $\bar{S} \in \{2.5, 5, 10, 20\}$ seconds; time between visits to the gate $\bar{Z} = 10$ minutes.

This strongly suggests that the function f that maps $\frac{1}{c}$ to the average response time is convex; the graph of a convex function is below its chords, thus

$$f\left(\frac{1}{2c}\right) < \frac{1}{2}f\left(\frac{1}{c}\right)$$

$$f\left(\frac{1}{2c}\right) < \frac{f(0) + f\left(\frac{1}{c}\right)}{2}$$

\therefore convexity of $f(\cdot)$

and doubling the capacity **does reduce the waiting time by more than 2.**

We also see that a key value is $c^* = \frac{\bar{N}}{\bar{Z}}$. Note that $\frac{1}{Z}$ is the rate at which one customer would arrive at the gate if there would be no queuing, thus c^* is the rate of customers if the gate would not delay them. If c is much larger than c^* , the waiting time is small, so doubling the capacity has little effect anyhow. For c much smaller than c^* , the waiting time increases at an almost constant rate. Thus we should target c of the order of c^* , in other words, we should match the capacity of the gate to the “natural” rate c^* .

$$c^* = \frac{\bar{N} - 1}{\bar{Z}}$$

w.r.t. $1/c$

QUESTION 8.9.2. *Assume the system is highly congested before doubling the capacity. What is the reduction in waiting time after doubling capacity?* ²⁰

²⁰For a highly congested system ($2c$ much smaller than c^*) the offset at 0 becomes negligible and the response time is almost linear in $1/c$. Thus doubling the capacity does reduce the waiting time by 2, roughly speaking – but the system is still congested after doubling the capacity.

5. Networks of Queues: **Stability**

- Queuing networks are frequently used models.
- The stability issue may, in general, be a hard one.
- **Necessary** condition for stability (**Natural Condition**)

server utilization < 1

at **every** queue

- **All closed queuing networks are stable unconditionally.**

Instability Examples

IIE Transactions (1997) 29, 213–219

Simulation studies of multiclass queueing networks

J. BANKS and J. G. DAI

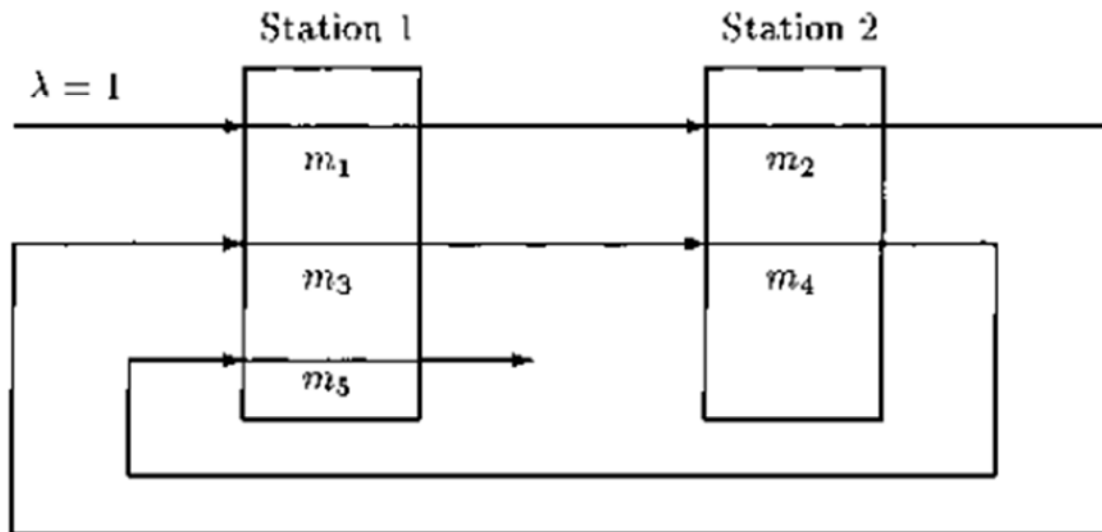


Fig. 1. An example of reentrant lines.

- Poisson arrivals ; jobs go through stations **1,2,1,2,1** then leave
- A job arrives as type 1, then becomes 2, then 3 etc.
- Exponential, independent service times with mean m_i
- **Priority scheduling**
 - ▶ Station 1 : $5 > 3 > 1$
 - ▶ Station 2: $2 > 4$
- Q: What is the *natural* stability condition ?
- A: $\lambda (m_1 + m_3 + m_5) < 1$
 $\lambda (m_2 + m_4) < 1$

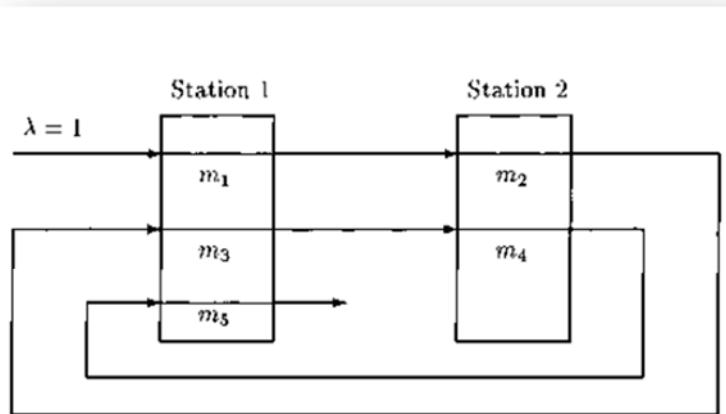


Fig. 1. An example of reentrant lines.

- $\lambda = 1$
 - $m_1 = m_3 = m_4 = 0.1$
 - $m_2 = m_5 = 0.6$
- Utilization factors
 - ▶ Station 1: 0.8
 - ▶ Station 2: 0.7
- Network is unstable !
- If $\lambda (m_1 + \dots + m_5) < 1$ network is stable; why?

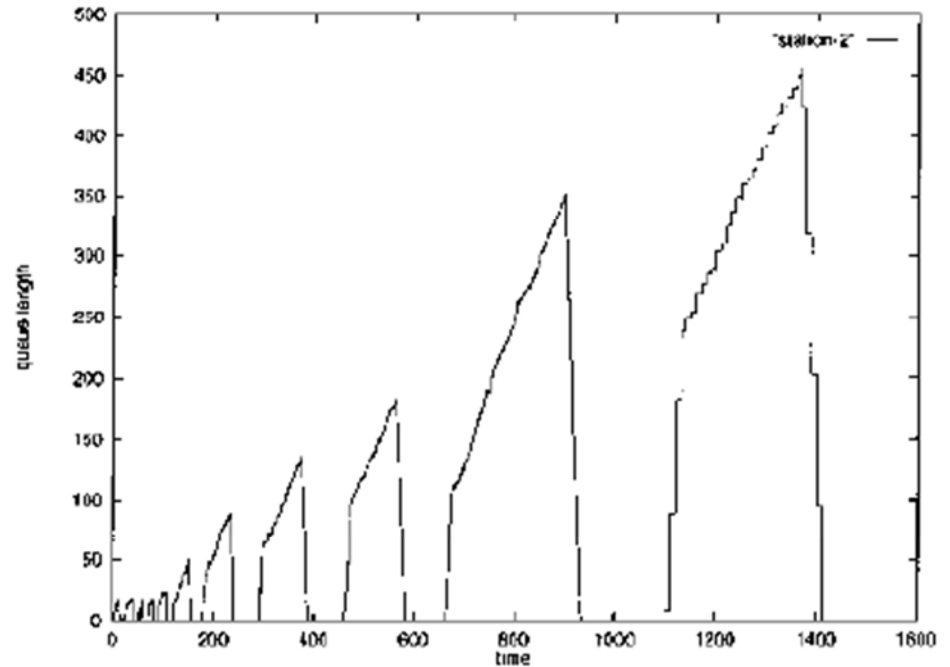
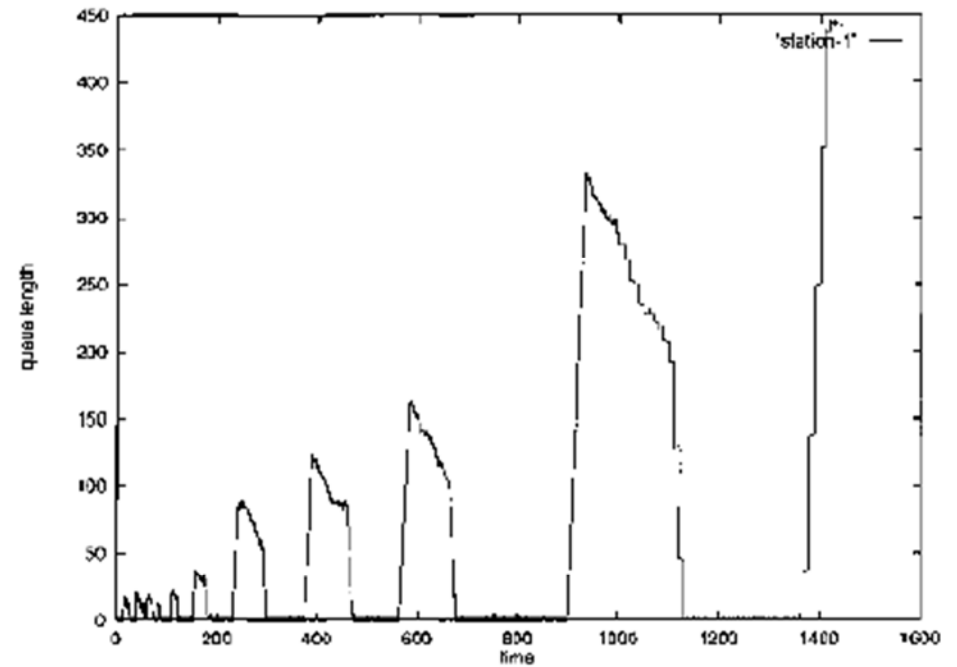


Fig. 2. Job size plots at stations 1 and 2.

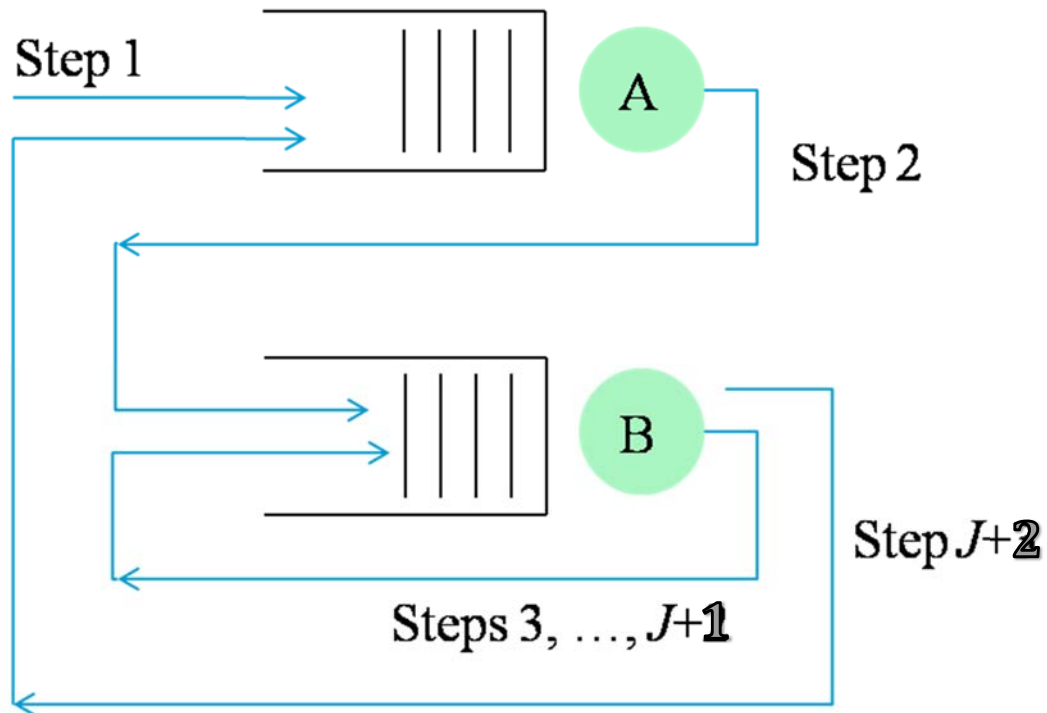
Bramson's Example 1: A Simple FIFO Network

The Annals of Applied Probability
1994, Vol. 4, No. 2, 414-431

INSTABILITY OF FIFO QUEUEING NETWORKS

BY MAURY BRAMSON¹

$S \quad L \quad S \quad S \quad L$
 A, B, \dots, B, A
 J times



- Poisson arrivals; jobs go through stations A, B, B, ..., B, A then leave
- Exponential, independent service times **but dependent on step**
 - ▶ Steps 2 and last: mean is L
 - ▶ Other steps: mean is S
- Q: What is the natural stability condition ?
- A: $\lambda (L + S) < 1$
 $\lambda ((J-1)S + L) < 1$
- Bramson showed: may be unstable whereas natural stability condition holds

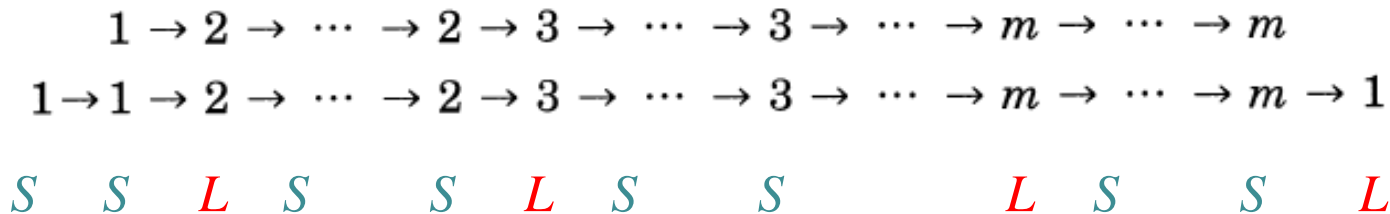
Bramson's Example 2

A FIFO Network with Arbitrarily Small Utilization Factor

The Annals of Applied Probability
1994, Vol. 4, No. 3, 693–718

INSTABILITY OF FIFO QUEUEING NETWORKS WITH QUICK SERVICE TIMES¹

BY MAURY BRAMSON



- m queues
- 2 types of customers
- $\lambda = 0.5$ each type
- routing as shown, ... = 7 visits
- FIFO
- Exponential service times, with mean as shown

- Utilization factor at every station $\leq 4 \lambda S$
- **Network is unstable for 'very small utilization'**
 $S \leq 0.01$
 $L \leq S^8$
 $m = \text{floor}(-2 (\log L)/L)$

Take Home Message

- The natural stability condition is necessary but may not be sufficient
 - ▶ It does **not** bring about **stability** in general queueing networks.
- There is a class of networks where this never happens.
 - ▶ **Product Form Queuing Networks**

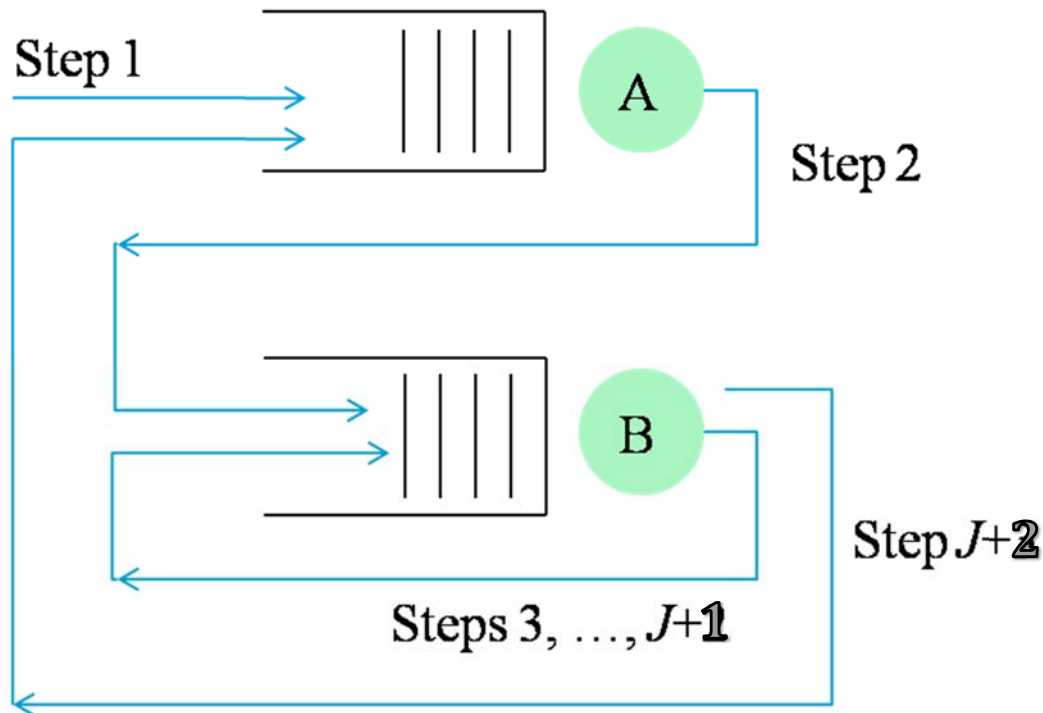
Product Form Networks

- Customers have a **class** attribute
- Customers visit **stations** according to *Markov Routing*

routing matrix $Q = \left(q_{c,c'}^{s,s'} \right)_{s,s',c,c'}$

- External arrivals, if any, are Poisson

A class- c customer leaving station s joins station s' as class c' with probability $q_{c,c'}^{s,s'}$



2 Stations
Class = step, J+2 classes

Can you reduce the number of classes ?

No. 'Markov Routing' does not remember the step state. 'Class' must capture all kinds of 'state'.

Chains

- Customers can switch *class*, but remain in the same *chain*

We say that two classes c, c' are *chain equivalent* if $c = c'$ or if it is possible for a class c -customer to eventually become a class c' customer, or vice-versa. This defines an equivalence relation between classes, the equivalence classes are called *chains*.

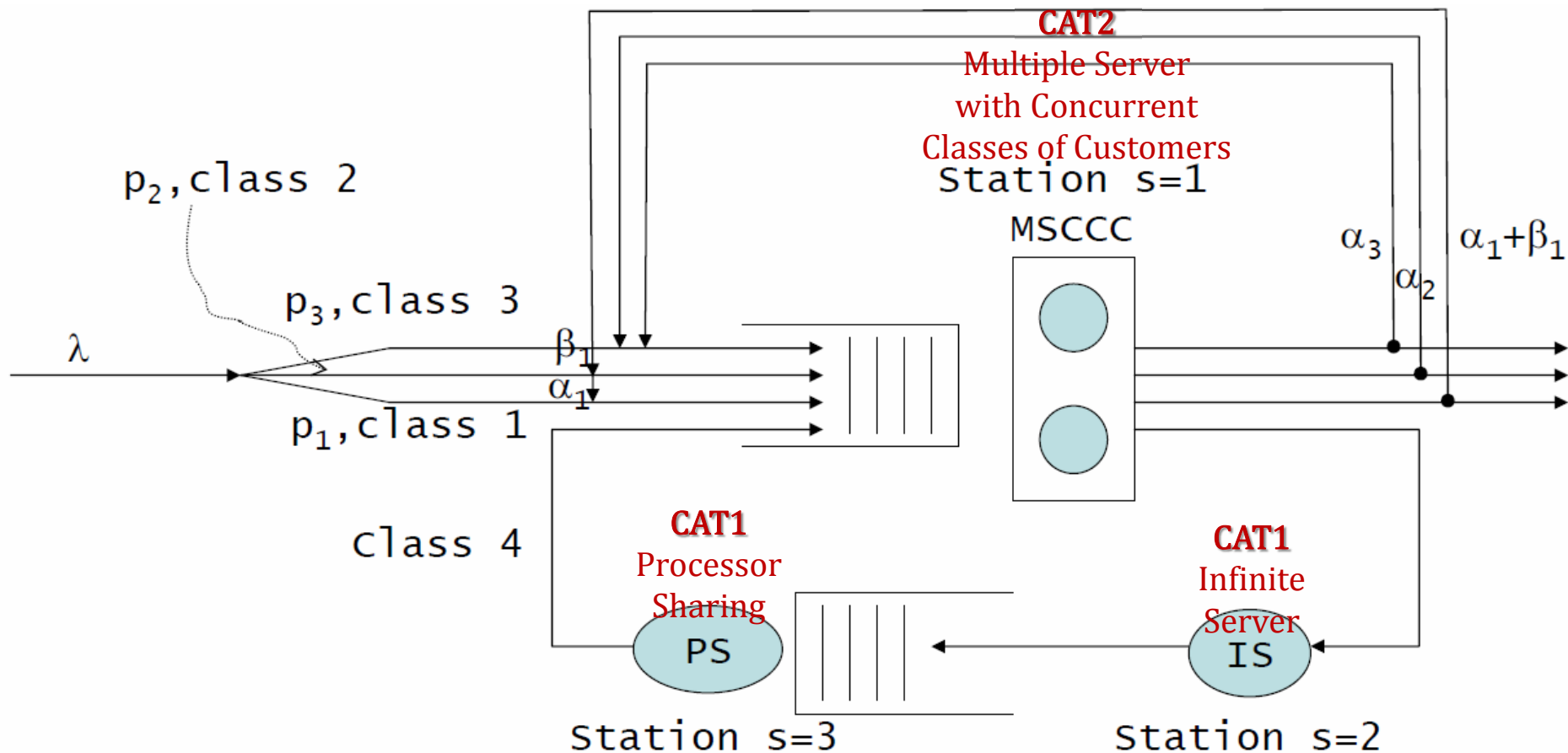


Figure 8.11: A Simple Product Form queuing network with 2 chains of customers, representing a machine with dual core processor. Chain 1 consists of classes 1, 2 and 3. Chain 2 consists of class 4.

Chains may be open or closed

- **Open chain**: with Poisson arrivals. Customers must **eventually** leave
- **Closed chain**: no arrival, no departure; number of **customers is constant**
- There is **no other type** of chain than these two.

- **Closed network has only closed chains.**
- **Open network has only open chains.**
- **Mixed network may have both.**

**Components of
Markov Routing Matrix:
 $\alpha_1, \alpha_2, \alpha_3, \beta_1$**

3 Stations
4 classes
1 open chain \rightarrow Chain 1
1 closed chain \rightarrow Chain 2

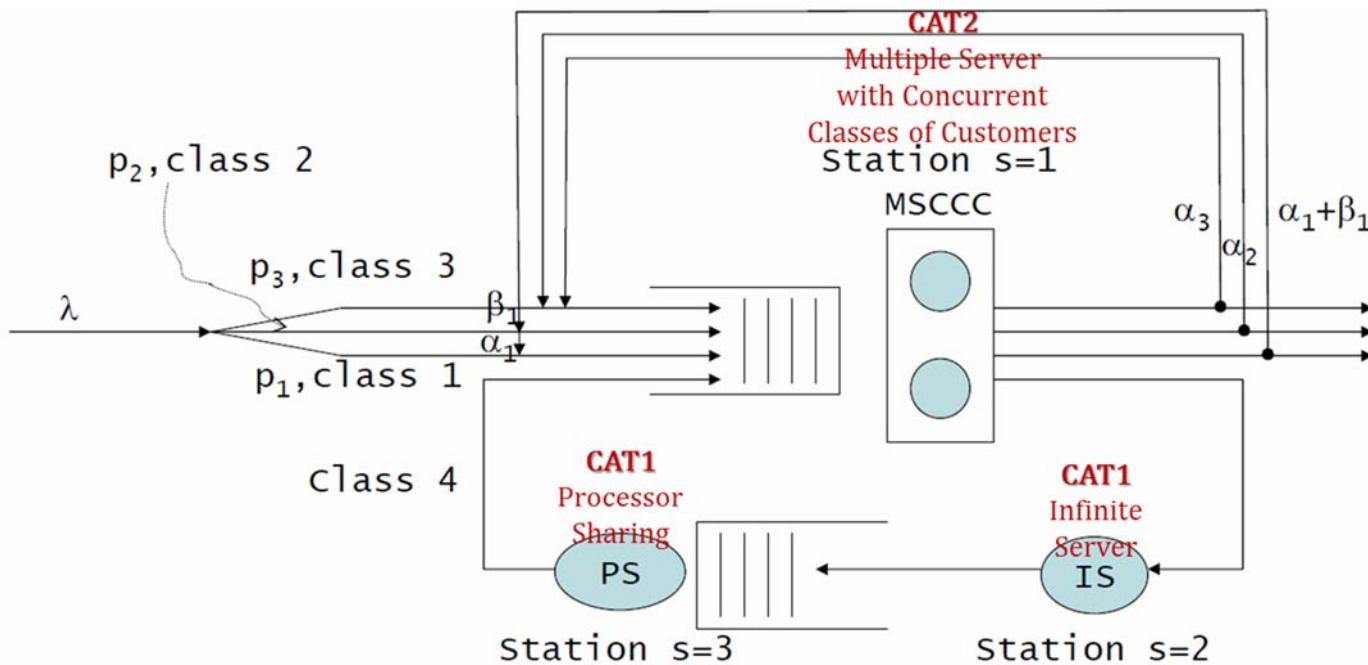


Figure 8.11: A Simple Product Form queuing network with 2 chains of customers, representing a machine with dual core processor. Chain 1 consists of classes 1, 2 and 3. Chain 2 consists of class 4.

Visit Rates

of class c at station s

We define the numbers θ_c^s (visit rates) as one solution to

the queuing network equivalent of
'stationary distribution'

$$\theta_c^s = \sum_{s',c'} \theta_{c'}^{s'} q_{c',c}^{s',s} + \nu_c^s \quad (8.24)$$

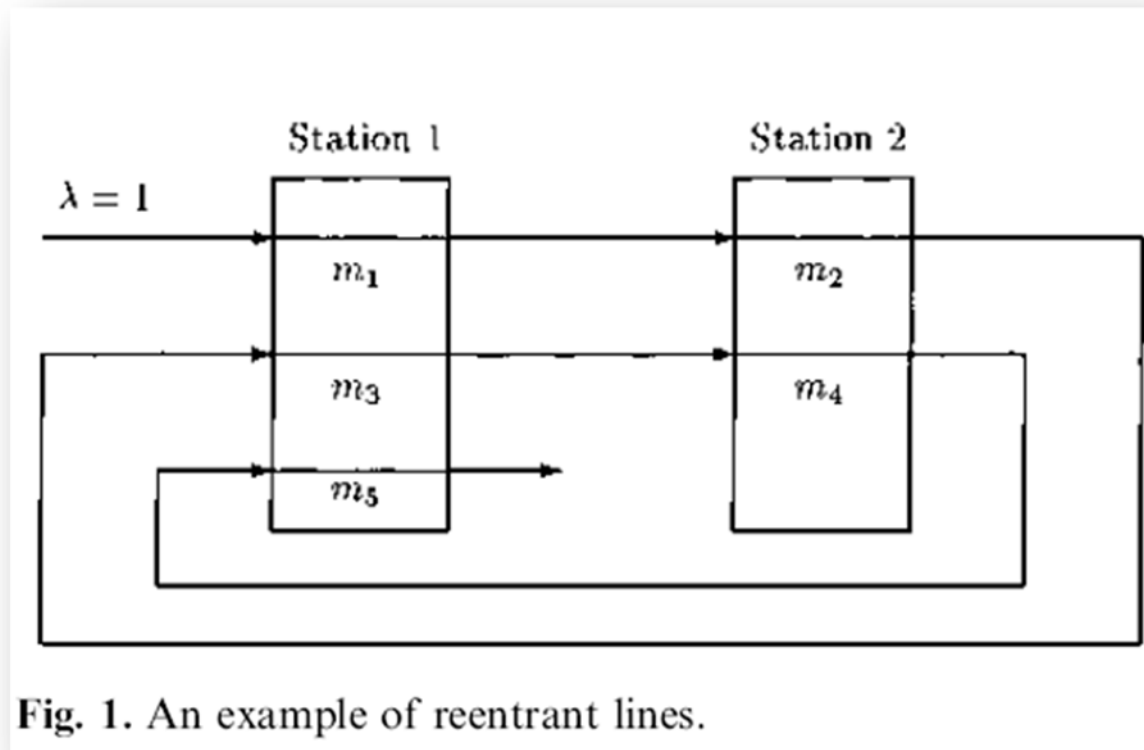
If the network is open, this solution is unique and θ_c^s can be interpreted¹² as the number of arrivals per time unit of class- c customers at station s . If c belongs to a closed chain, θ_c^s is determined only up to one multiplicative constant per chain. We assume that the array $(\theta_c^s)_{s,c}$ is one non identically zero, non negative solution of Eq.(8.24).

The constant depends on the total number
of customers in the closed network.

Distinction between routing matrix and visit rate vector:

For each customer, routing matrix decides the probability she transitions to a station whereas visit rate decides how many customers is present in a station.

¹²This interpretation is valid when the network satisfies the stability condition in Theorem 8.5.1



2 Stations
 5 classes
 1 chain
 Network is **open**

Visit rates
 $\theta^1_1 = \theta^1_3 = \theta^1_5 = \theta^2_2 = \theta^2_4 = \lambda$
 $\theta^s_c = 0$ otherwise

Plugging

$$\left\{ \begin{array}{l} q_{1,1}^{1,1} = \alpha_1; \quad q_{1,2}^{1,1} = \beta_1; \\ q_{2,3}^{1,1} = \alpha_2; \\ q_{3,3}^{1,1} = \alpha_3; \\ q_{4,4}^{1,2} = 1; \quad q_{4,4}^{2,3} = 1; \quad q_{4,4}^{3,1} = 1; \\ q_{c,c'}^{s,s'} = 0 \text{ otherwise} \end{array} \right.$$

yields:

Class 1:	$\theta_1^1 = \nu \frac{p_1}{1-\alpha_1};$	$\theta_1^2 = 0;$	$\theta_1^3 = 0;$
Class 2:	$\theta_2^1 = \nu \left(p_2 + \beta_1 \frac{p_1}{1-\alpha_1} \right);$	$\theta_2^2 = 0;$	$\theta_2^3 = 0;$
Class 3:	$\theta_3^1 = \nu \frac{1}{1-\alpha_3} \left(p_3 + \alpha_2 p_2 + \alpha_2 \beta_1 \frac{p_1}{1-\alpha_1} \right);$	$\theta_3^2 = 0;$	$\theta_3^3 = 0;$
Class 4:	$\theta_4^1 = 1;$	$\theta_4^2 = 1;$	$\theta_4^3 = 1.$

into

$$\theta_c^s = \sum_{s',c'} \theta_{c'}^{s'} q_{c',c}^{s',s} + \nu_c^s$$

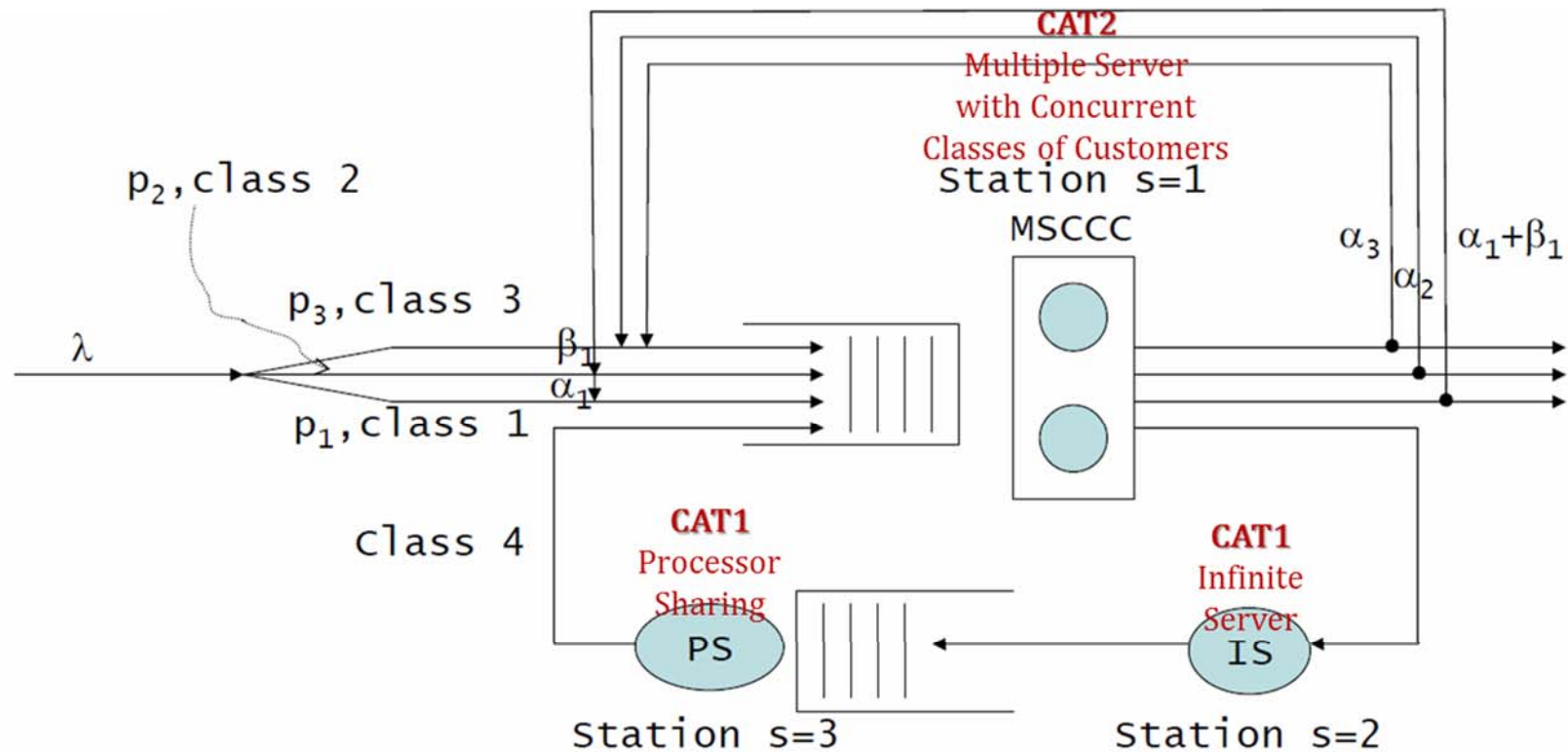


Figure 8.11: A Simple Product Form queuing network with 2 chains of customers, representing a machine with dual core processor. Chain 1 consists of classes 1, 2 and 3. Chain 2 consists of class 4.

Constraints on Stations

for 'Product Form Networks'

- Stations must belong to a restricted catalog of stations
- See Section 8.4 for comprehensive description
- We will give commonly used examples
- **Category 1**
- Example 1: *Global Processor Sharing Station*
 - ▶ One server
 - ▶ Rate of server is shared equally among all customers present
 - ▶ Service requirements (time) for customers of class c are drawn iid from a distribution which **depends** on the class (and the station)
- Example 2: *Delay Station (also called as Infinite Server)*
 - ▶ Infinite number of servers (comprises the station)
 - ▶ Service requirements for customers of class c are drawn iid from a distribution which **depends** on the class (and the station)
 - ▶ **No queuing**, service time = service requirement = residence time

■ **Category 2** (MSCCC station): *MSCCC with B servers*

▶ B servers

▶ FIFO queuing with constraints

For each class, no. of customers in service is upper bounded.

▶ Service requirements for customers of class c are drawn iid from an **exponential** distribution, **independent** of the class (but may depend on the station)

■ Example 3 : *FIFO with B servers*

▶ B servers

▶ FIFO queuing discipline

▶ Service requirements for customers of class c are drawn iid from an **exponential** distribution, **independent** of the class (but may depend on the station)

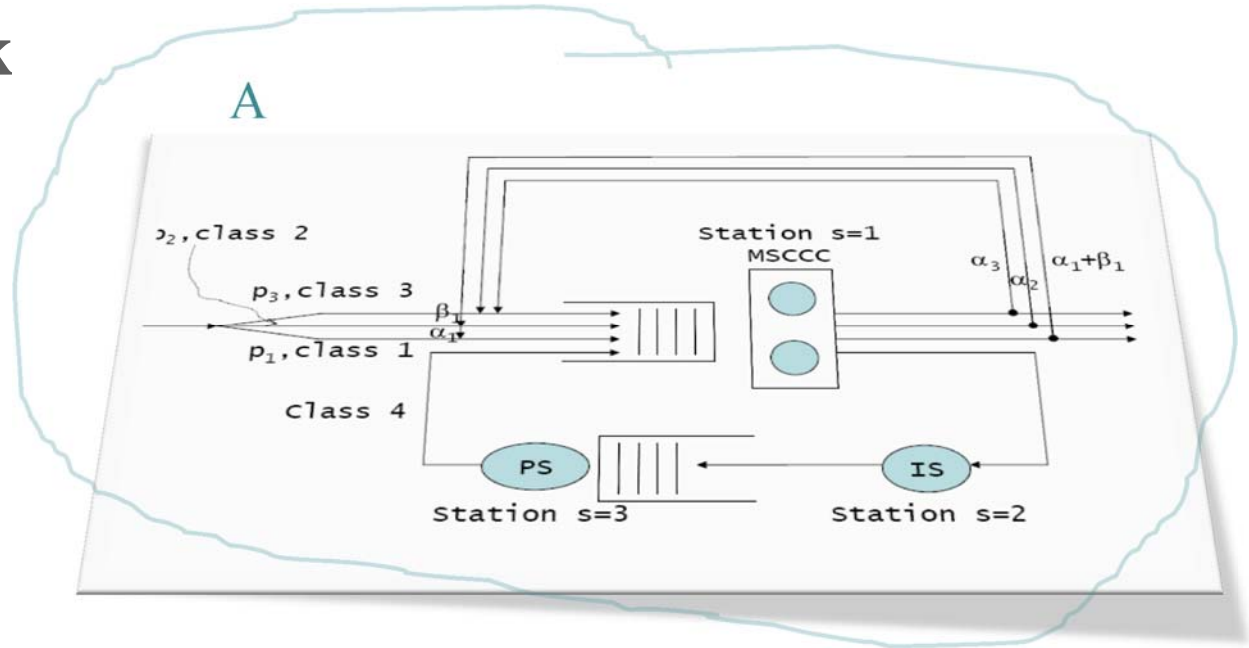
■ Examples 1 and 2 are **insensitive** (to service distribution)

▶ **service time not restricted**, but the types of stations are uncommon and, **no FIFO**

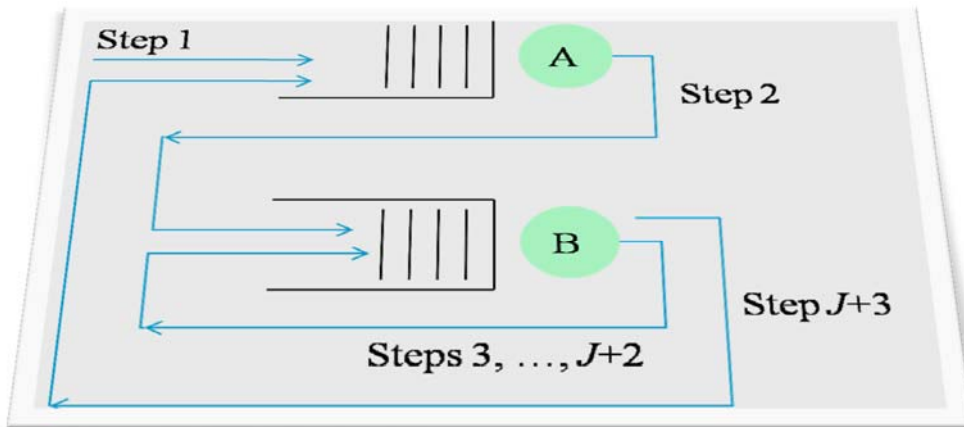
■ Example 3 is not (service time must be exponential, same for all)

▶ FIFO is commonly used queuing discipline

■ Say which network satisfies the hypotheses for product form



B (FIFO, Exp)



Similar to Category 2 but class-dependent service time

Violation!

C (Prio, Exp)

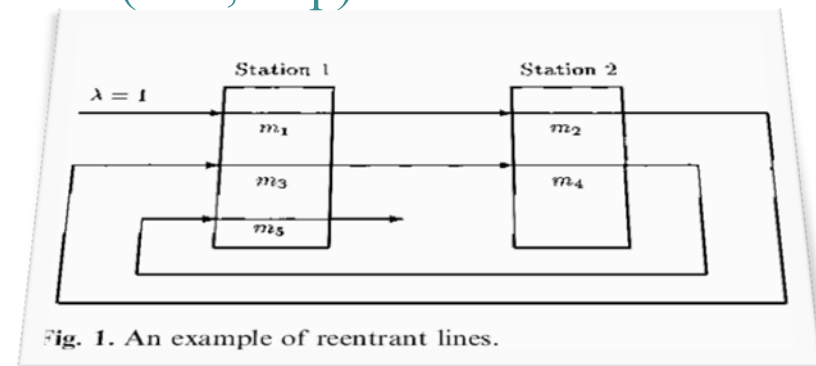


Fig. 1. An example of reentrant lines.

Similar to Category 2 but 'Priority' queuing discipline

Violation!

The Product Form Theorem

Natural Stability Condition Finally Holds!

- If a network satisfies the «Product Form» conditions given earlier
 - ▶ The stationary distribution of numbers of customers can be written **explicitly**
 - ▶ It is a product of terms, where each term *depends only on the station*
 - ▶ Efficient algorithms exist to compute performance metrics for even very large networks
 - ▶ For PS and Delay stations, **service time distribution does not matter** other than through its mean (*insensitivity*)
 - ▶ **The natural *stability* condition holds**
 - ▶ See Theorem 8.5.4 for the simplest form

Two Representative Cases

8.3.3 THE PROCESSOR SHARING QUEUE, M/GI/1/PS

$$\mathbb{P}(N(t) = k) = (1 - \rho)\rho^k$$

M/M/B QUEUE

For more specific system, one can say more. A frequently used system is the M/M/B queue, i.e. the system with Poisson arrivals, B servers, exponential service times and FIFO discipline. The system can be studied directly by solving for the stationary probability. Here when $\rho < 1$ there is a unique stationary regime, which is also reached asymptotically when we start from arbitrary initial conditions; for $\rho \geq 1$ there is no stationary regime.

When $\rho < 1$ the stationary probability is given by

$$\mathbb{P}(N(t) = k) = \begin{cases} \eta \frac{(B\rho)^k}{k!} & \text{if } 0 \leq k \leq B \\ \eta \frac{B^B \rho^k}{B!} & \text{if } k > B \end{cases} \quad (8.21)$$

Conclusions

- Be aware of the state-of-the-art and **demystify** it.
- **Poisson** arrival is reasonable for many problems: Don't reject it outright!
 - ▶ M/M/1, M/GI/1, M/G/1/PS and variants have **closed forms**
- **Bottleneck analysis** and worst case analysis are usually very simple and often give good **insights** (far more desirable than nothing)
- Queuing networks may be very complex to analyze except if they are **product form**! If you manage to **adapt** your system to a product-form network with reasonable assumptions, there are plentiful theory.
 - ▶ For computational aspects (exacerbated by product form per se), see Ch. 8.6.
- The most **critical limitation** of queueing network is the absence of "**conditional branch**" between stations.
 - ▶ How to circumvent this absence?