POLITECNICO DI MILANO

Computing Infrastructures



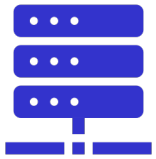


Performance modeling

Prof. Gianluca Palermo

Credits: Hilston, Marzolla, Mirandola, Gribaudo, Zahorian, Lazowska, Ardagna

The topics of the course: what are we going to see today?



HW Infrastructures:

System-level: Computing Infrastructures and Data Center Architectures, Rack/Structure;

Node-level: Server (computation, HW accelerators), Storage (Type, technology), Networking (architecture and technology);

Building-level: Cooling systems, power supply, failure recovery

SW Infrastructures:

Virtualization: Process/System VM, Virtualization Mechanisms (Hypervisor, Para/Full virtualization)

Computing Architectures: Cloud Computing (types, characteristics), Edge/Fog Computing, X-as-a service

Methods:

Reliability and availability of datacenters (definition, fundamental laws, RBDs)

Disk performance (Type, Performance, RAID)

Scalability and performance of datacenters (definitions, fundamental laws, queuing network theory)

Reference book

Quantitative System Performance

Computer System Analysis Using Queueing Network Models

Edward D. Lazowska, John Zahorjan,

G. Scott Graham, Kenneth C. Sevcik

Available on-line: http://homes.cs.washington.edu/~lazowska/qsp/

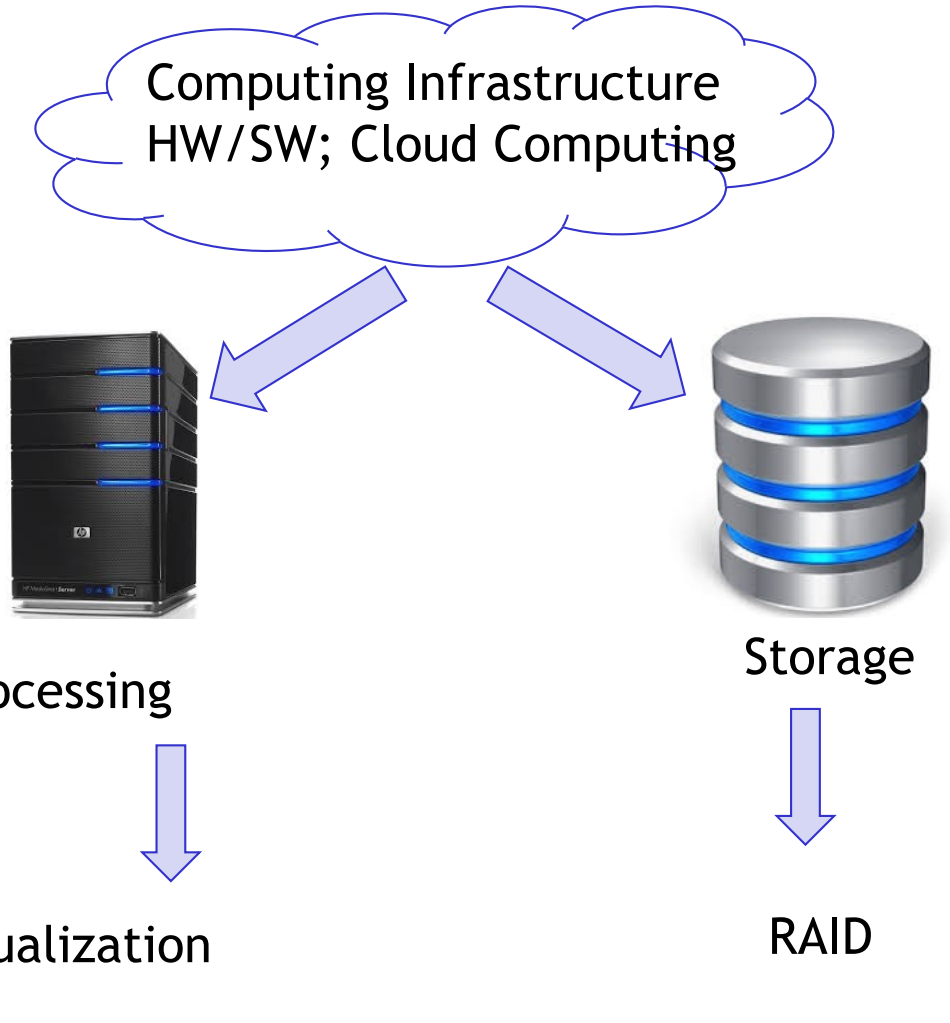
Big Picture

Performance

Average response time - Throughput

Dependability

MTTDL



Definitions: Performance

- **Computer performance:**
  - The total effectiveness of a computer system in terms
    - throughput
    - response time
    - availability
  - Can be characterized by the amount of useful work accomplished by a computer system or computer network compared to the time and resources used

System quality: a central issue

- Common practice:
  - system mostly validated versus “functional” requirements rather than versus quality ones
- However, computing systems are central in todays business and we need a deeper understanding on how they behave also from extrafunctional perspectives
- Little information related to quality is usually available early in the system lifecycle but understanding is of **great importance from the cost and performance point of view**
  - During design and system sizing
  - But also during system evolution
    - E.g. changes in workloads or amount of users

How can we evaluate system quality?

- Use of intuition and trend extrapolation
  - Unfortunately, those who possess these qualities in sufficient quantity are rare
- Pro: rapid and flexible
- Con: accuracy

- Experimental evaluation of alternatives
  - Experimentation is always valuable, often required, and sometimes the approach of choice
  - It also is expensive - often prohibitively so
  - A further drawback: an experiment is likely to yield accurate knowledge of system behavior under one set of assumptions, but not any insight that would allow generalization
- Pro: excellent accuracy
- Con: laborious and inflexible

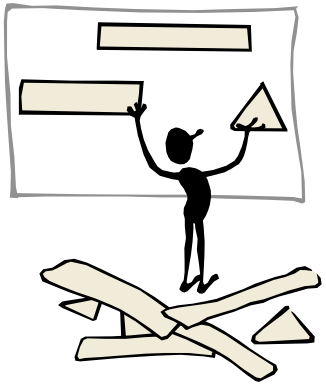
Solution: Model-Based Approach

Systems are complex so…

Abstraction of the systems: Models

"an attempt to distill, from the details of the system, exactly those aspects that are essentials to the system behavior"....
(E. Lazoswka)

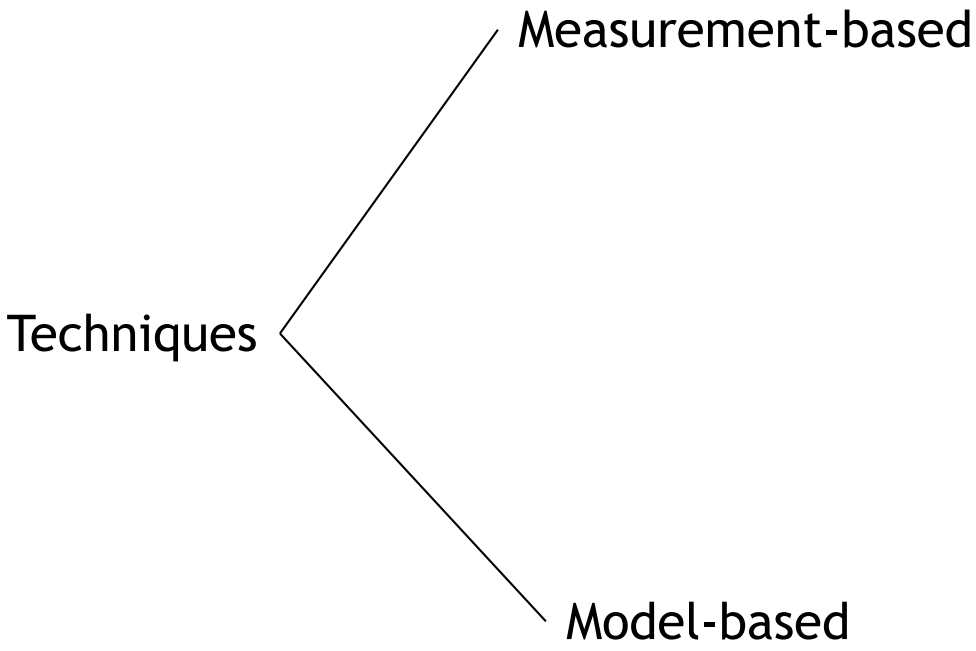
Often models are the only artifact to deal with!
e.g., design phase



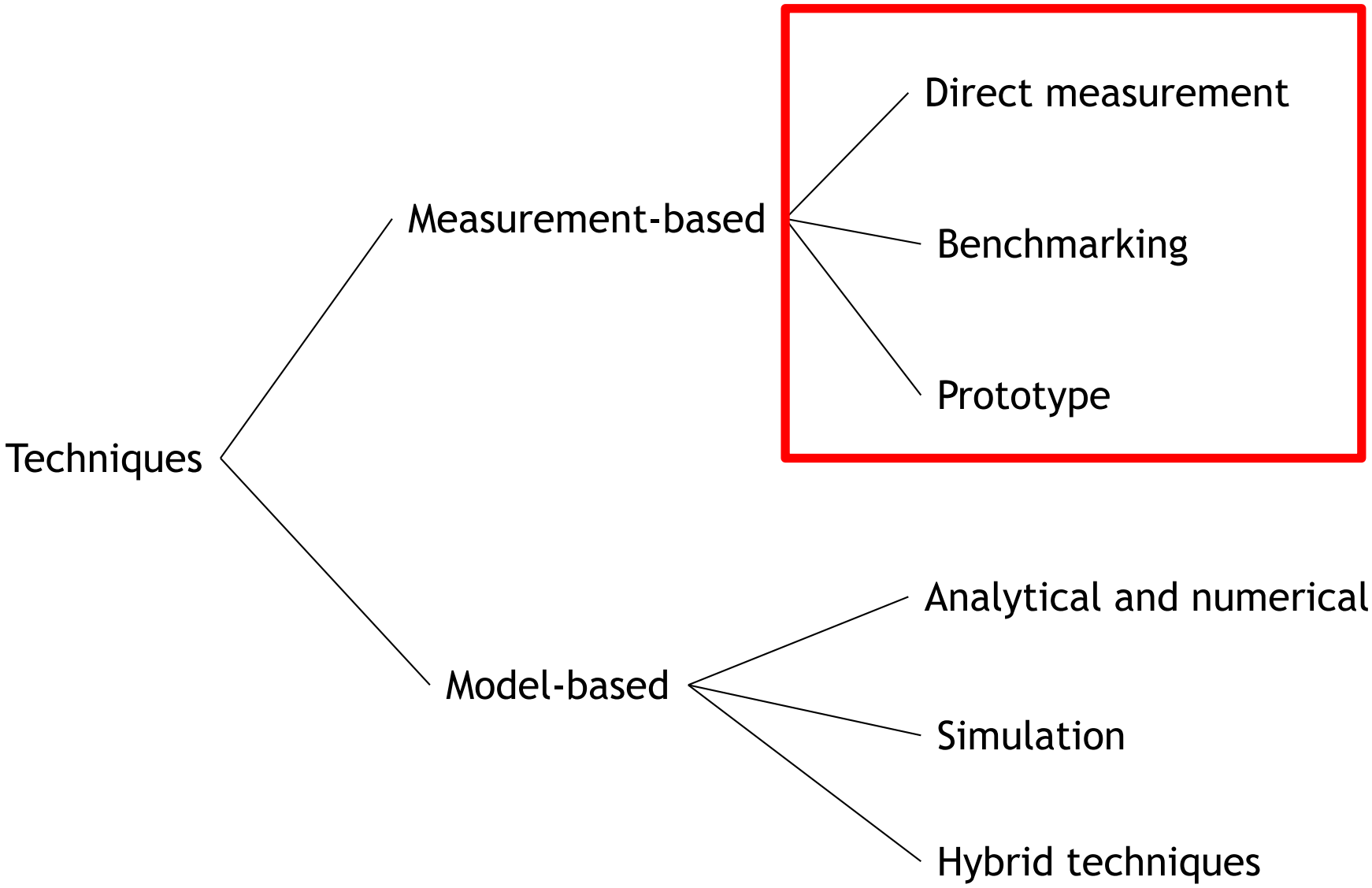
Models used to *drive* design decisions

- Which architecture ?
- How many resources to meet some performance/reliability goal?
- ...

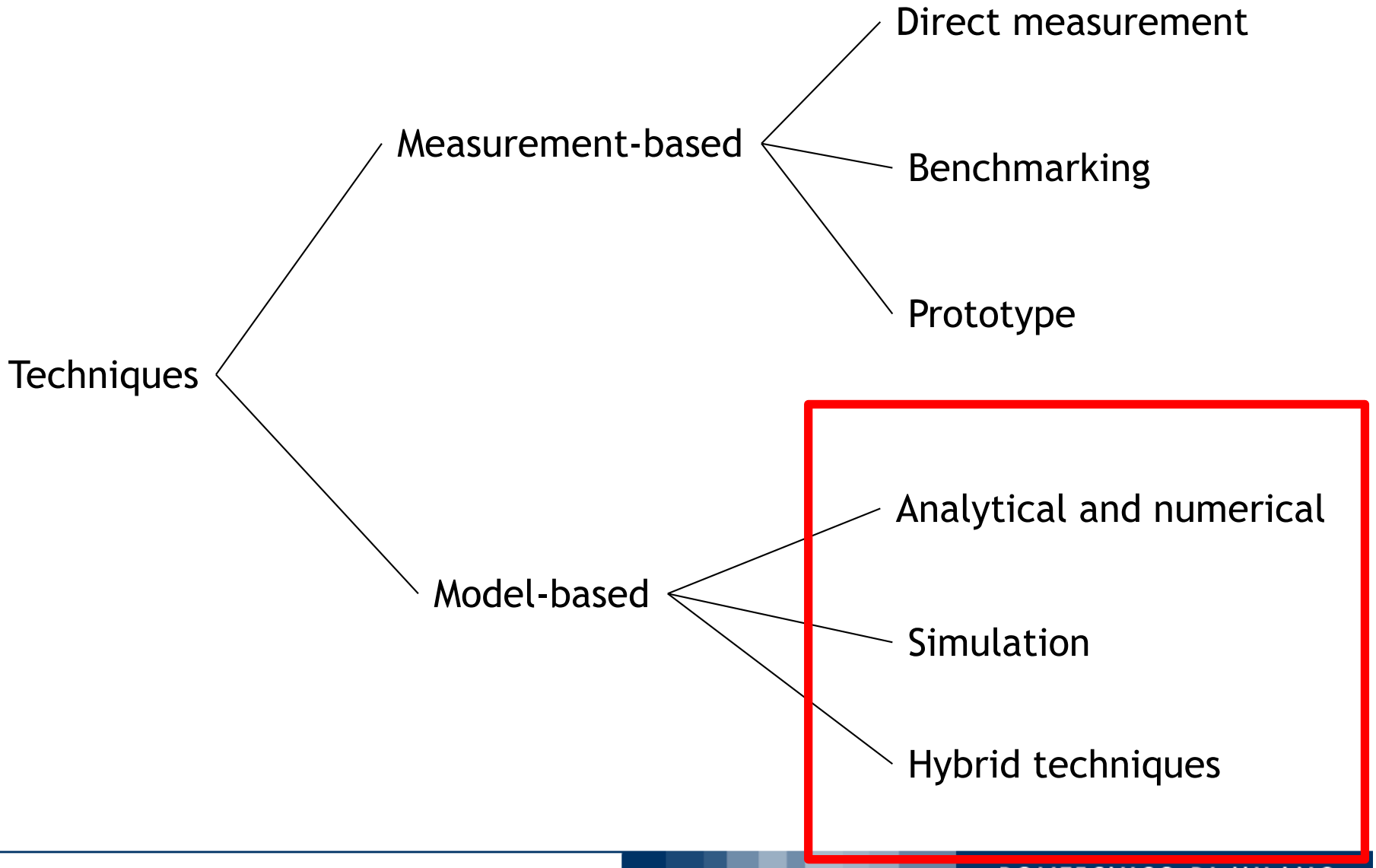
Quality Evaluation techniques



Quality Evaluation techniques



Quality Evaluation techniques



Evaluation techniques: summary

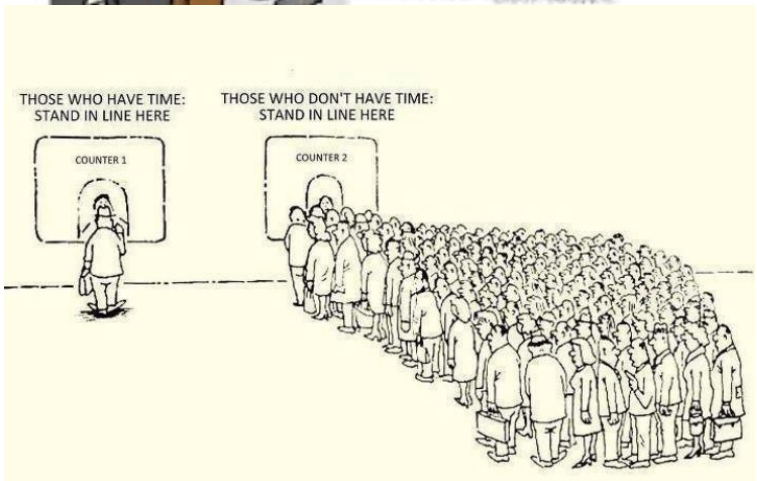
- Analytical and Numerical techniques are based on the application of mathematical techniques, which usually exploit results coming from the theory of probability and stochastic process
  - They are the most efficient and the most precise, but are available only in very limited cases
- Simulation techniques are based on the reproduction of traces of the model.
  - They are the most general, but might also be the less accurate, especially when considering cases in which rare events can occur.
  - The solution time can also become really large when high accuracy is desired
- Hybrid techniques combine analytical/numerical methods with simulation

Reference Models of this course: Queueing Networks

Queueing theory is the theory behind what happens when you have a lot of jobs, scarce resources, and so long queue and delays.

Queueing network modelling is a particular approach to computer system modelling in which the computer system is represented as a *network of queues*

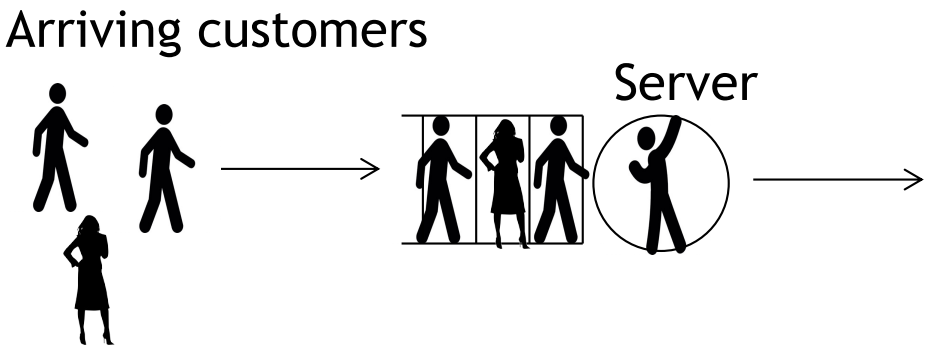
A network of queues is a collection of *service centers*, which represent system **resources**, and *customers*, which represent **users** or **transactions**



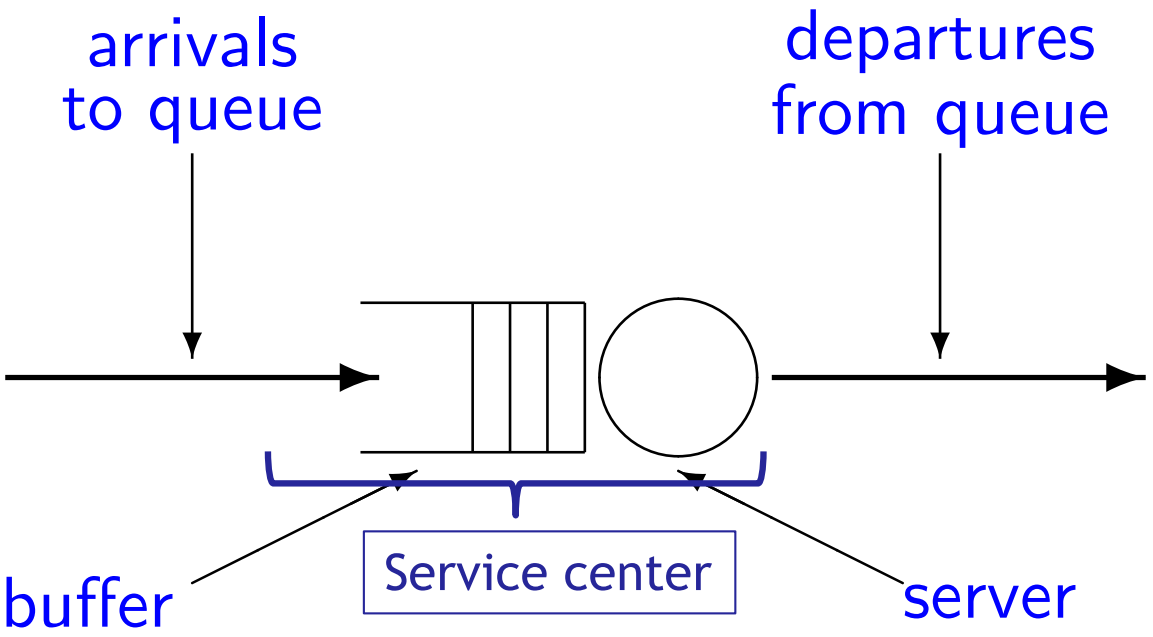
Queueing Networks

- Queueing theory applies whenever queues come up
- Queues in computer systems:
  - CPU uses a time-sharing scheduler
  - Disk serves a queue of requests waiting to read or write blocks
  - A router in a network serves a queue of packets waiting to be routed
  - Databases have lock queues, where transactions wait to acquire the lock on a record
- Predicting performance e.g. for capacity planning purposes
- Queueing theory is built on an area of mathematics called stochastic modelling and analysis

Success of queueing network: low-level details of a system are largely irrelevant to its high-level performance characteristics



QN: graphical representation



Service center

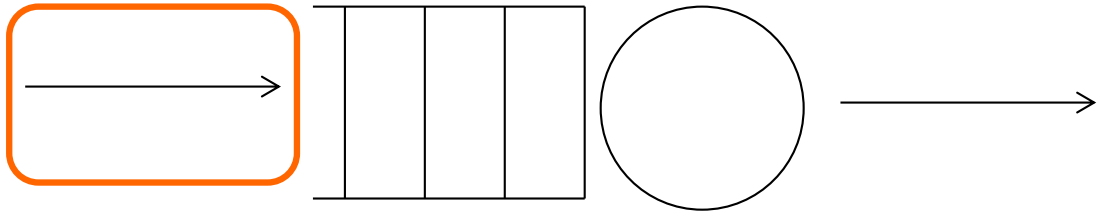
Different aspects characterize queuing models:

- Arrival
- Service
- Queue
- Population

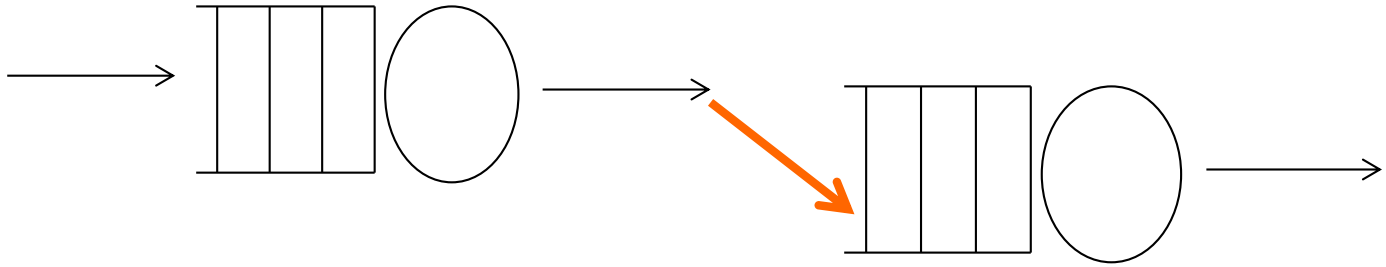
Arrival of customers

Arrivals represent jobs entering the system: they specify how fast, how often and which types of jobs does the station service.

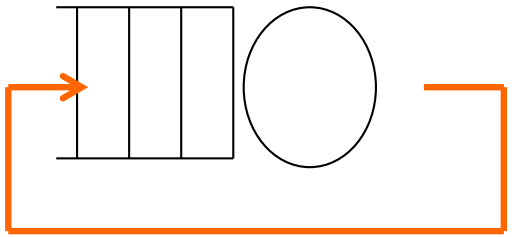
Arrival can come from an external source:



Arrival can come from another queue:



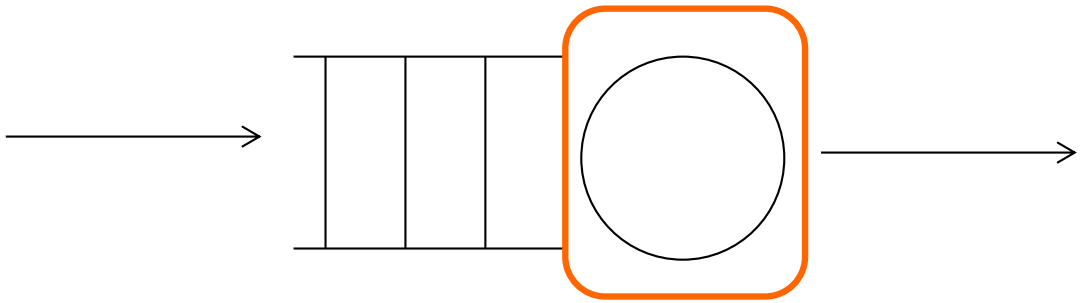
or even from the same queue, through a loop-back arc



Service center

Different aspects characterize queuing models:

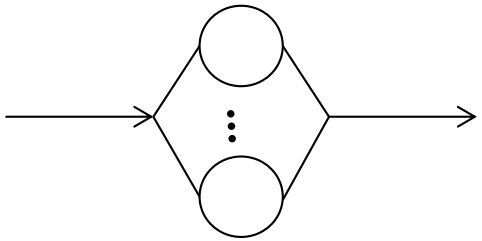
- Arrival
- Service
  - represents the time a job spends being served.
- Queue
- Population



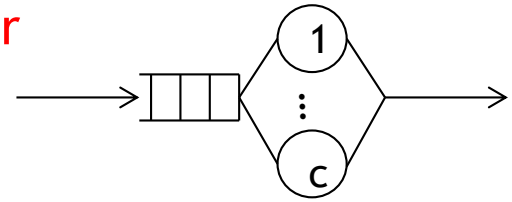
Number of servers: Possible Situations

- Single server:
  - capability to serve one customer at a time;
  - waiting customers remains in the buffer until chosen for service;
  - the next customer is chosen depending on the service discipline

- Infinite server:
  - there are always at least as many servers as there are customers,
    - each customer can have a dedicated server
  - There is no queueing, (and no buffer) in such facilities



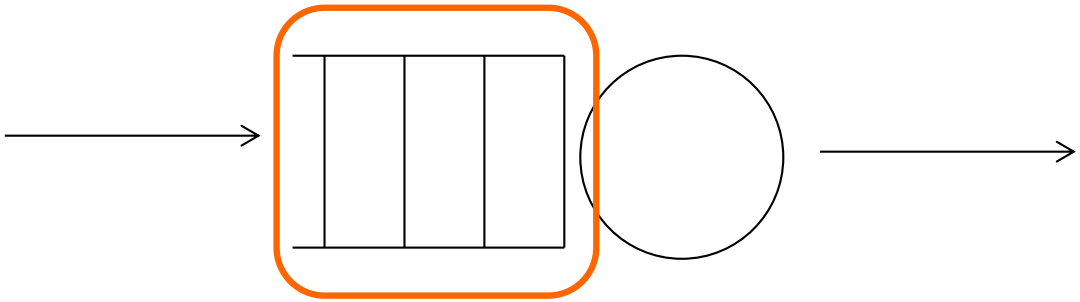
- Multiple servers:
  - Fixed number of c servers, each of which can service a customer at any time
  - If the number of customers in the facility is less than or equal to c there will no queueing
  - If there are more than c customers, the additional customers will have to wait in the buffer



Service center

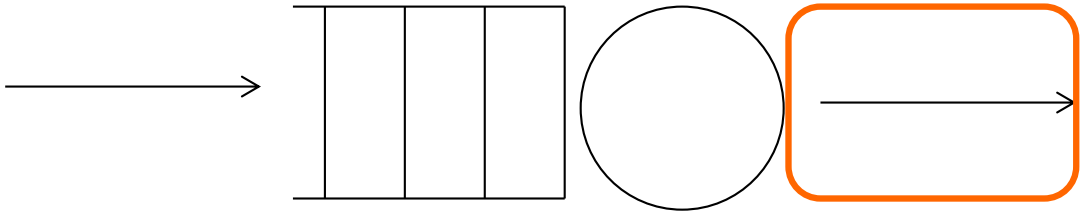
Different aspects characterize queuing models:

- Arrival
- Service
- Queue
  - If jobs exceed the capacity of parallel processing of the system, they are forced to wait *queueing* in a *buffer*
- Population

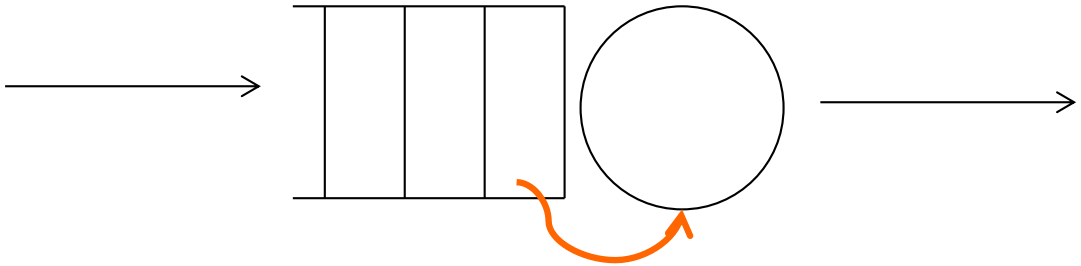


Queue

When the (one of the) job(s) currently in service leaves the system, one of the job in the queue can enter the now free service center



Service discipline/queuing policy determines which of the job in the queue will be selected to start its service



Service center

Different aspects characterize queuing models:

- Arrival
- Service
- Queue
- Population

Population

- Ideally, members of the population are indistinguishable from each other
- When this is not the case we divide the population into classes whose members all exhibit the same behaviour
- Different classes differ in one or more characteristics, for example, arrival rate, service demand
- Identifying different classes is a workload characterisation task

Example

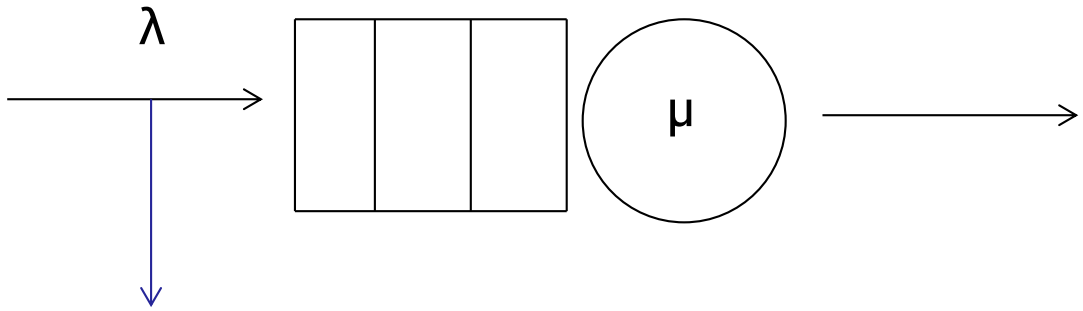
- Consider a wireless access gateway:
- Measurements have shown that packets arrive at a mean rate of 125 packets per second, and are buffered
- The gateway takes 2 milliseconds on average to transmit a packet
- The buffer currently has 13 places, including the place occupied by the packet being transmitted and packets that arrive when the buffer is full are lost
- **Goal of the modelling and analysis:**
  - We wish to find out if the buffer capacity which is sufficient to ensure that less than one packet per million gets lost

Example

Making exponential assumptions about the arrival rate and the service rate we would model the gateway as:

A single queue center with:

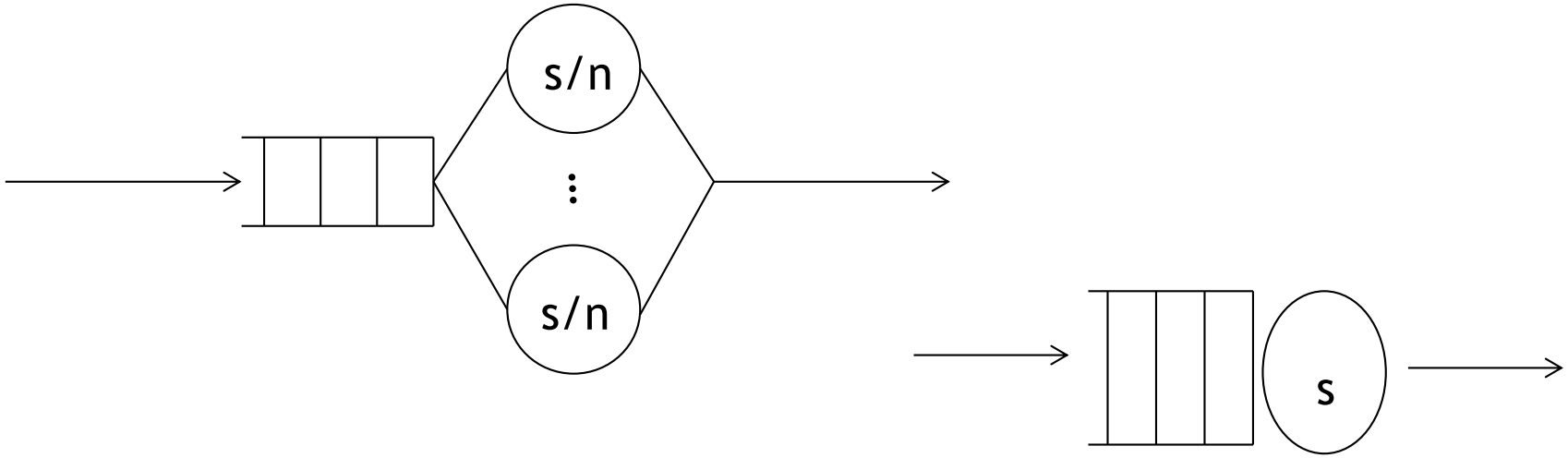
- Finite queue capacity=13
- FCFS service discipline
- Exponential arrival distribution
  - rate λ = 125 req/s
- Exponential service distribution
  - μ = 1/(2ms) = 500 req/s



Example

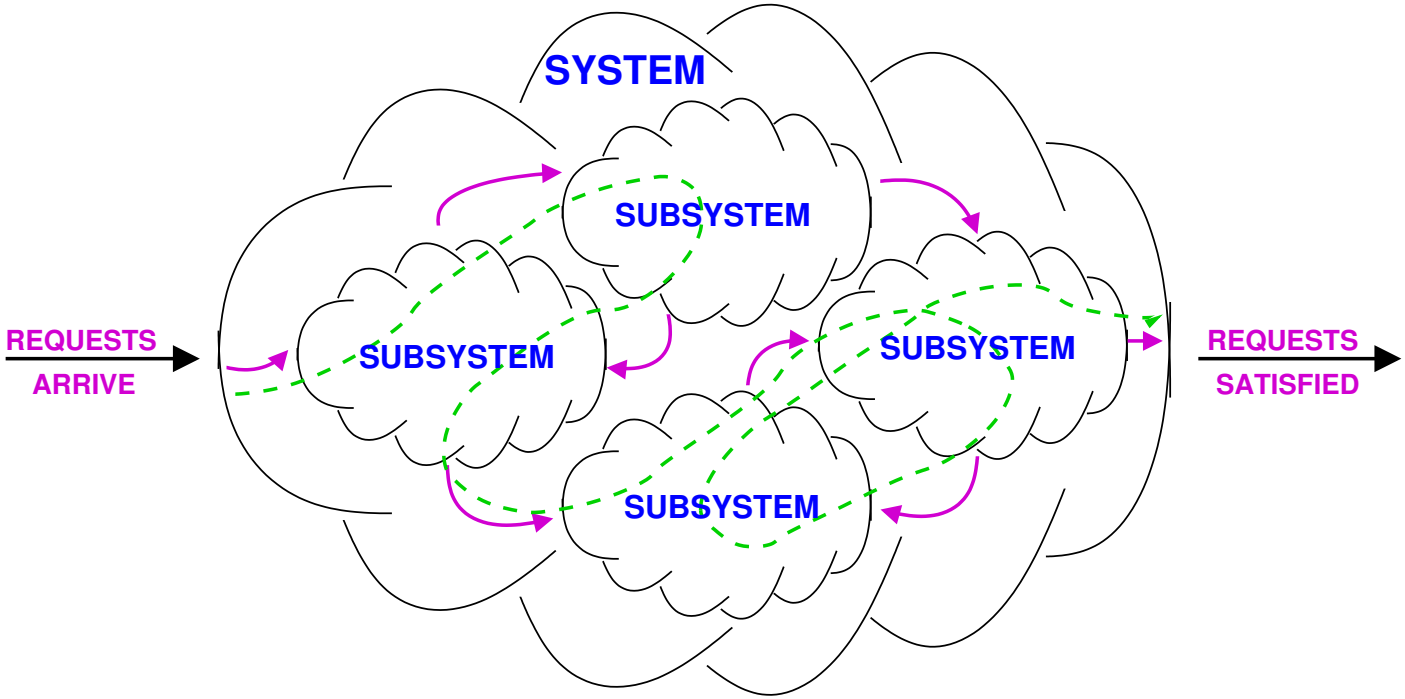
- You are given a choice between one fast CPU of speed s or n slow CPU each of speed s/n. Your goal is to minimize mean response time

- Question: Which is the better choice?
  - Arrival rate? Job type?

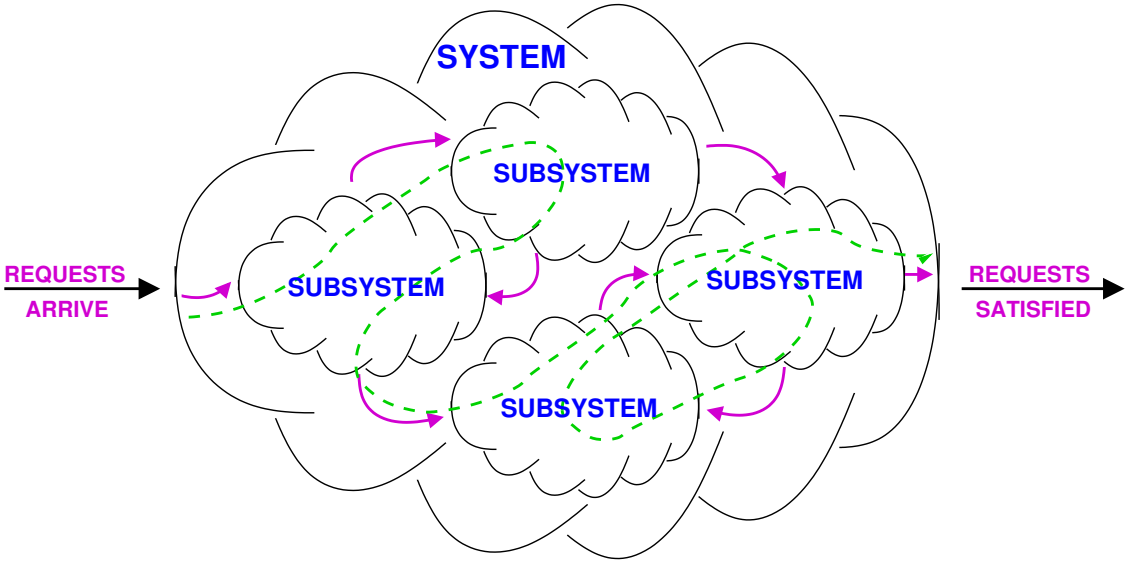


Queueing Networks

For many systems we can adopt a view of the system as a collection of resources and devices with customers or jobs circulating between them



Queueing Networks

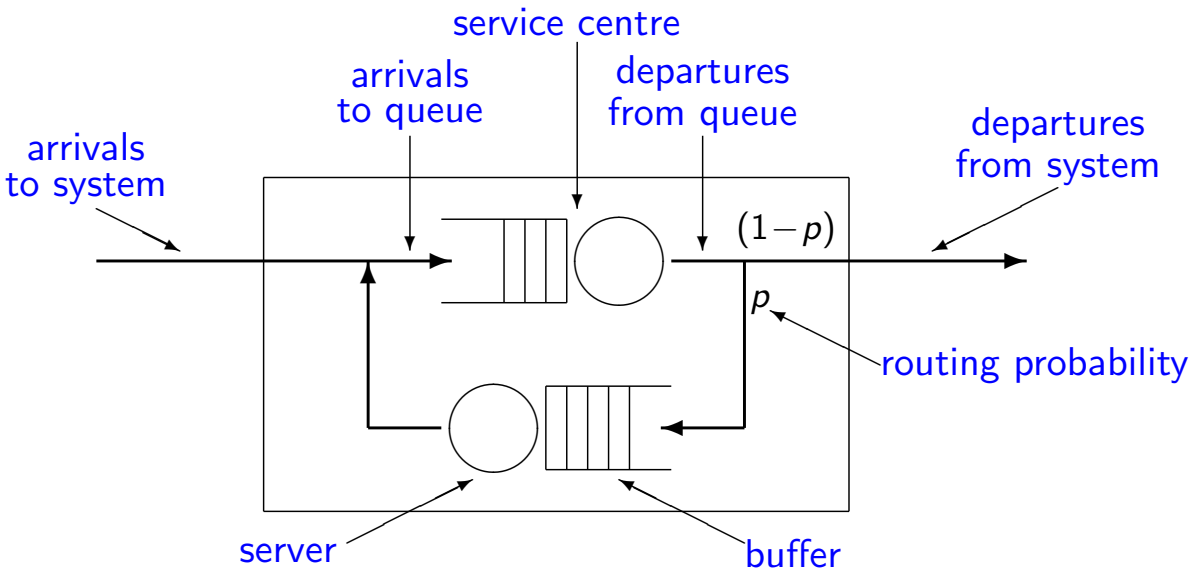


- We can associate a service center with each resource in the system and then route customers among the service centres
- After service at one service centre a customer may progress to other service centres, following some previously defined pattern of behaviour, corresponding to the customer's requirement

Queueing Networks

A queueing network can be represented as a graph where nodes represent the service centers k and arcs the possible transitions of users from one service center to another

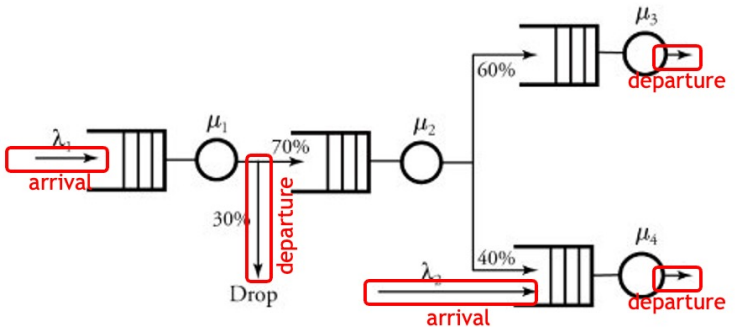
Nodes and arcs together define the network topology

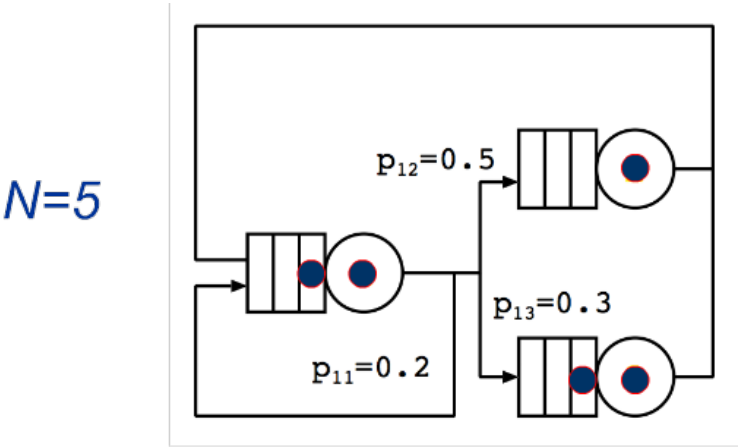


Queueing Networks

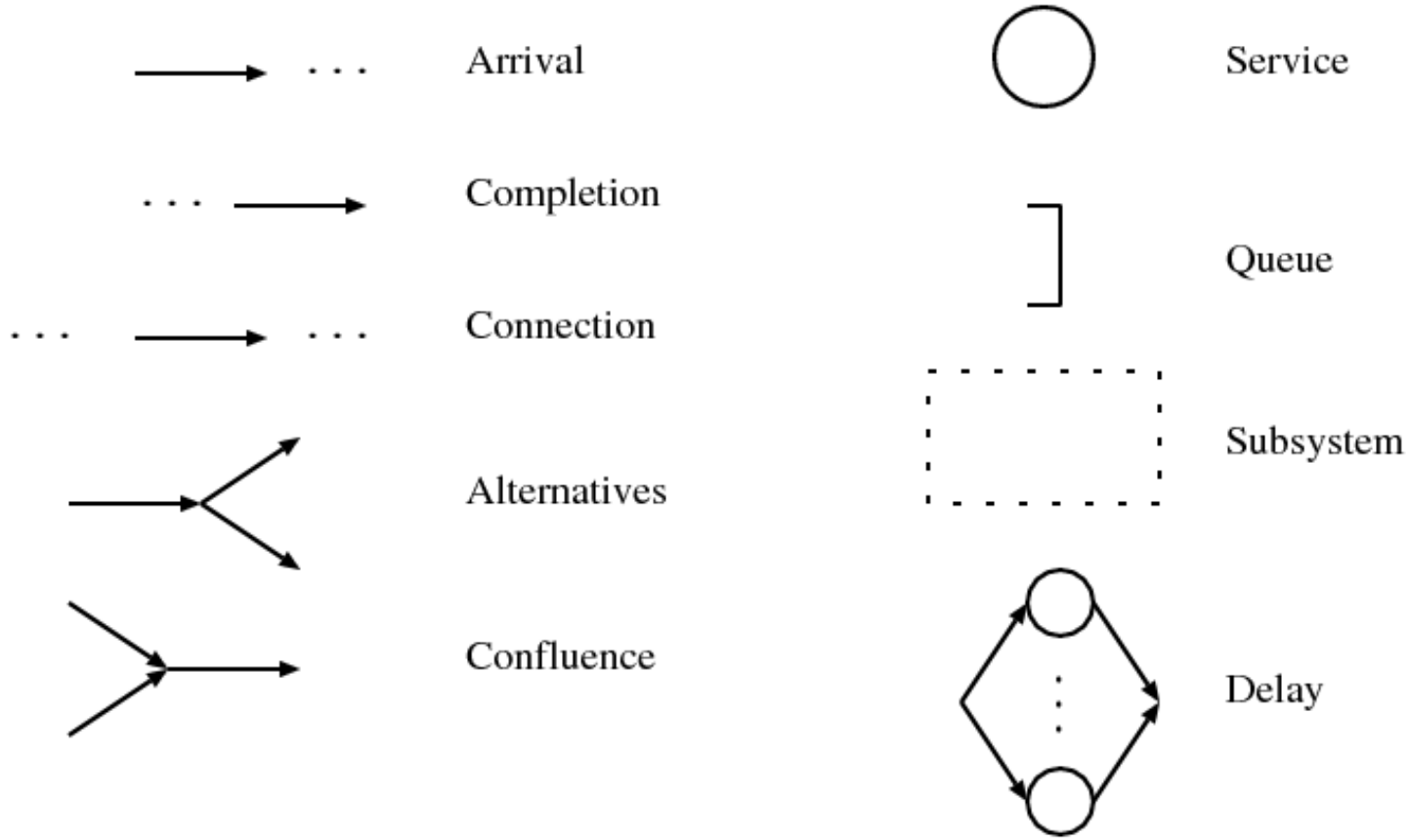
A network may be:

- Open: customers may arrive from, or depart to, some external environment
- Closed: a fixed population of customers remain within the system
- Mixed: there are classes of customers within the system exhibiting open and closed patterns of behaviour respectively





Graphical notation

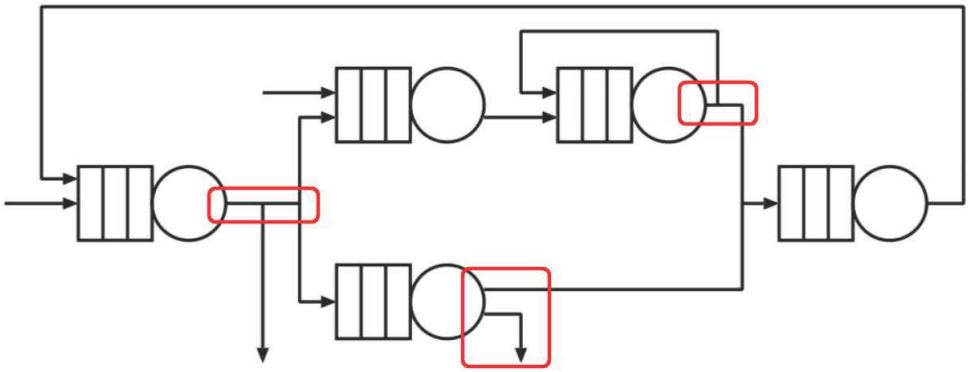


- Graphical notation is not unique, but it usually corresponds to a graph where edges denotes the flow of customers in the network

Queueing networks

Different aspects characterize queuing models:

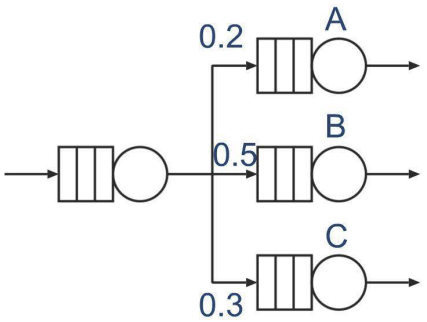
- Arrival
- Service
- Queue
- Population
- (NEW!) Routing
  - Whenever a job, after finishing service at a station has several possible alternative routes, an appropriate selection policy must be defined



Main Routing Algorithms

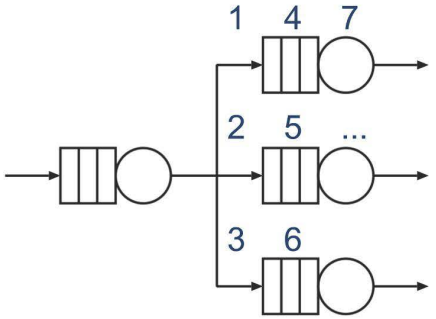
Probabilistic

- each path has assigned a probability of being chosen by the job that left the considered station



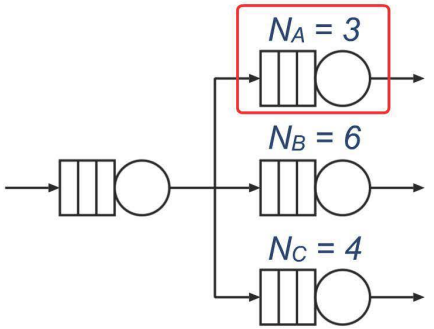
Round robin

- the destination chosen by the job rotates among all the possible exits

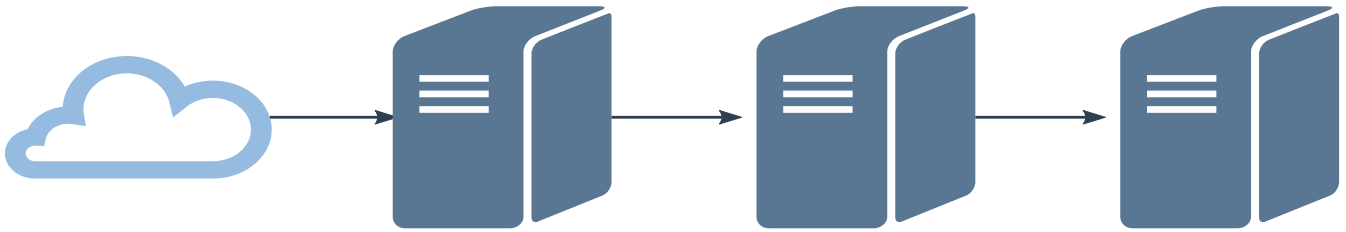


Join the shortest queue

- jobs can query the queue length of the possible destinations, and chose to move to the one with the smallest number of jobs waiting to be served



Open Networks: Examples



A client server system, dealing with external arrivals

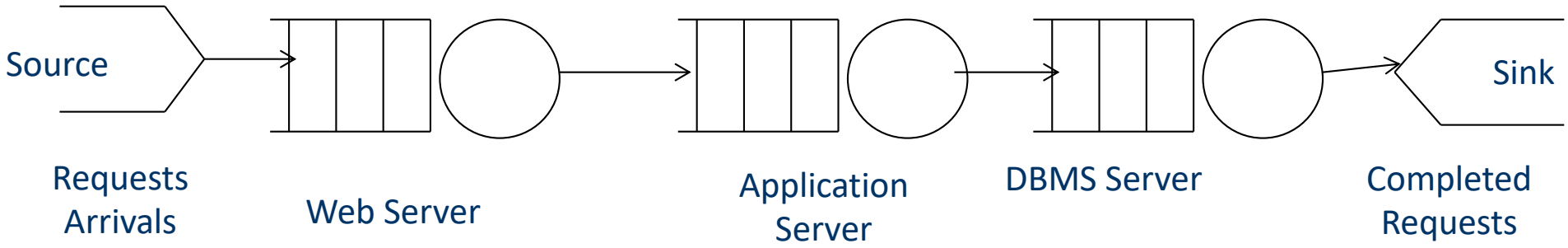
- Classical three tier architecture

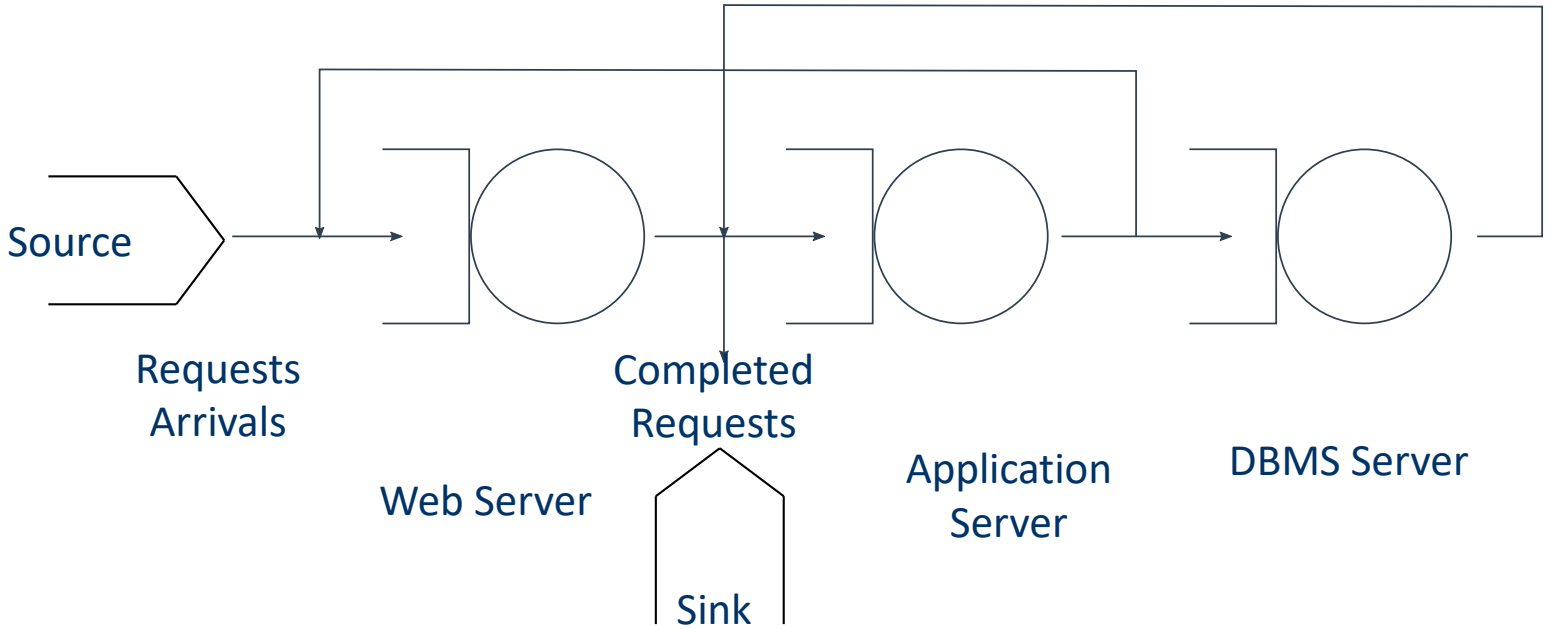
Provide a QN model of the system and evaluate the overall throughput considering that the network delay is negligible with respect to the other devices and two different cases:

1) The only thing we know is that each server should be visited by the application

2) In the second case we know that the application **after visiting the web server** requires some operations at **the application server** and then **can go back** to **the web server** and **leave** the system **or** can require service at the **DBMS** and then **go back** to the **application server**

Open Networks

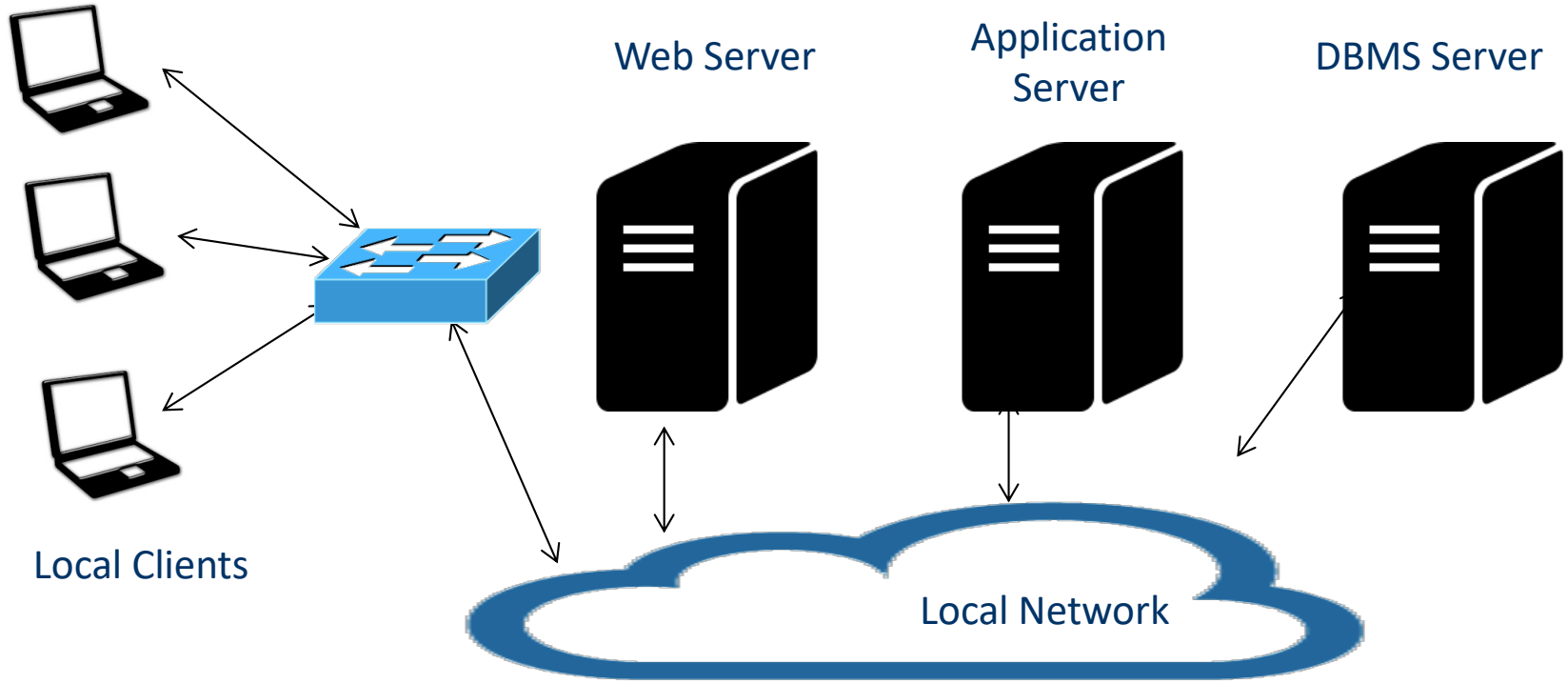




Scenario 1: Tandem networks

Tandem queuing networks are used for example to model production lines, where raw parts enter the systems, and after a set of stages, the final product is completed (and leaves)

Closed Networks: Examples



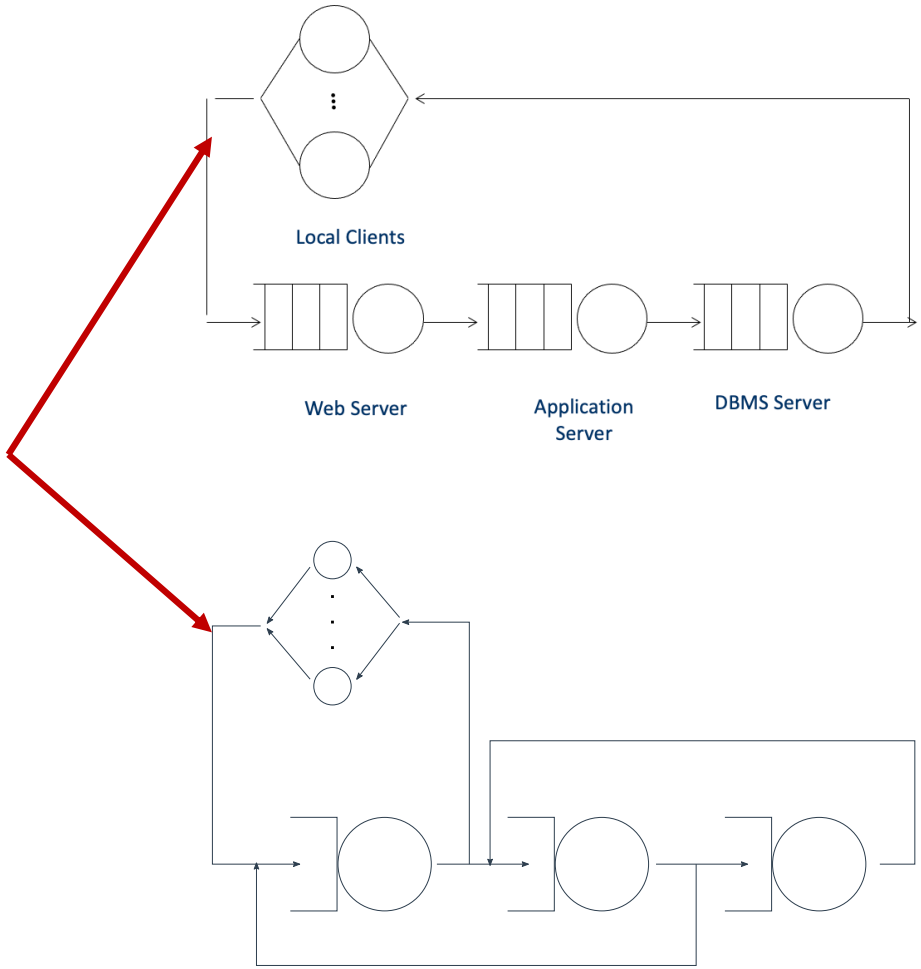
A client server system, **with a finite number of customers**:

- Classical three tier architecture
- Not accessible from outside, e.g. local network of a company

Provide a QN model of the system and evaluate the system throughput considering that Network delay is negligible with respect to the other devices. Model the two different cases previously described.

Close Networks

Users “think” before submitting new requests



Scenario 1

Scenario 2

Level of Detail