Queueing Systems

Alain Jean-Marie
INRIA/LIRMM, Université de Montpellier, CNRS
Alain.Jean-Marie@inria.fr

RESCOM Summer School 2019
June 2019

About the class

This class:

- Lesson 4: Queueing Systems (2h, now)
- Case Study 3: Queuing Systems (1h, after the break)

Companion classes:

- Lesson 5 and Case Study 4: Markov Decision Processes by E. Hyon (2h + 1h, tomorrow)

Common Lab:

- Lab 3: Queueing Systems and Markov Decision Processes (2h, Friday)

Related talk:

- Opening 1: Models of Hidden Markov Chains for Trace Analysis on the Internet by S. Vatton (1h30, tomorrow)

Preparation of the lab

Topic of the lab:

- programming the basic discrete-time queue with impatience using the library `marmoreCore` (C++ programming)
- programming the same queue with a control of the service using the library `marmoteMDP`
- finding the optimal control policy.

Preparation: two possibilities

- using a virtual machine with `virtualbox`
  - download VM + instructions from

`http://www-desir.lip6.fr/~hyon/Marmote/MachineVirtuelle/Rescom2019_TP.html`

  - copy it from USB drive, ask the technician!
- using the library (linux only)
  - tarball + instructions from

`https://marmotecore.gforge.inria.fr/dokuwiki/`

General Plan

Part 1: Basic features of queueing systems

Part 2: Metrics associated with queuing

Part 3: Modeling queueing systems with Markov chains

Part 4: Numerical solution and simulation

Part 1:
Basics of Queueing

Basics of Queueing: Plan

Table of contents

- customers, buffers, servers, arrival process, service time
- buffer capacity, blocking, rejection, impatience, balking, reneging
- customer classes, scheduling policy
- Kendall's notation
- networks of queues, routing, blocking

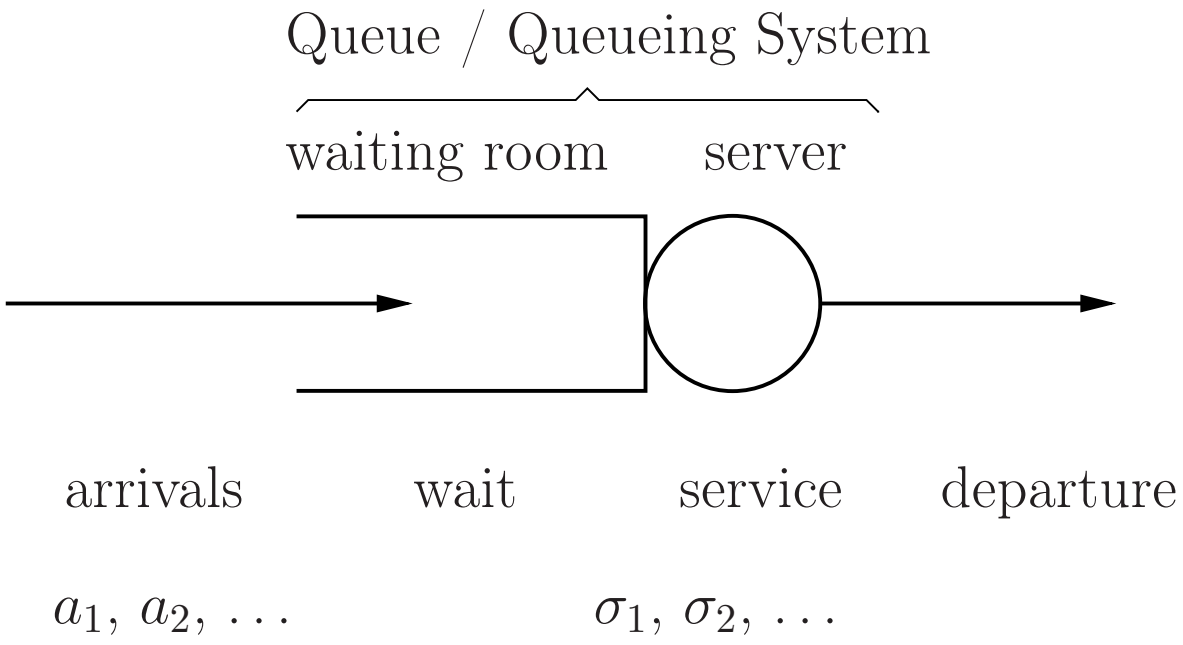
Why study queues?

Queues are everywhere...

... and in particular, a lot in networks.

A queue

A queue is an abstraction for several actors sharing a resource.



The usual representation of a queueing system

Components of a queue

The elements that form a queue are:

- one or several servers
  $\rightarrow$ manage the access to the resource(s)
- one waiting room
- (possibly) several classes of customers
- an arrival process per class
- a process of service times per class
  $\rightarrow$ the total duration a customer will need the resource
- a service discipline
- a departure process

The dynamics of queues

Queues are systems that evolve over time.

Fundamental quantities:

- $N(t)$ number of customers present in the queue
- $W(t)$ quantity of work (*workload*, backlog) to be done for these customers

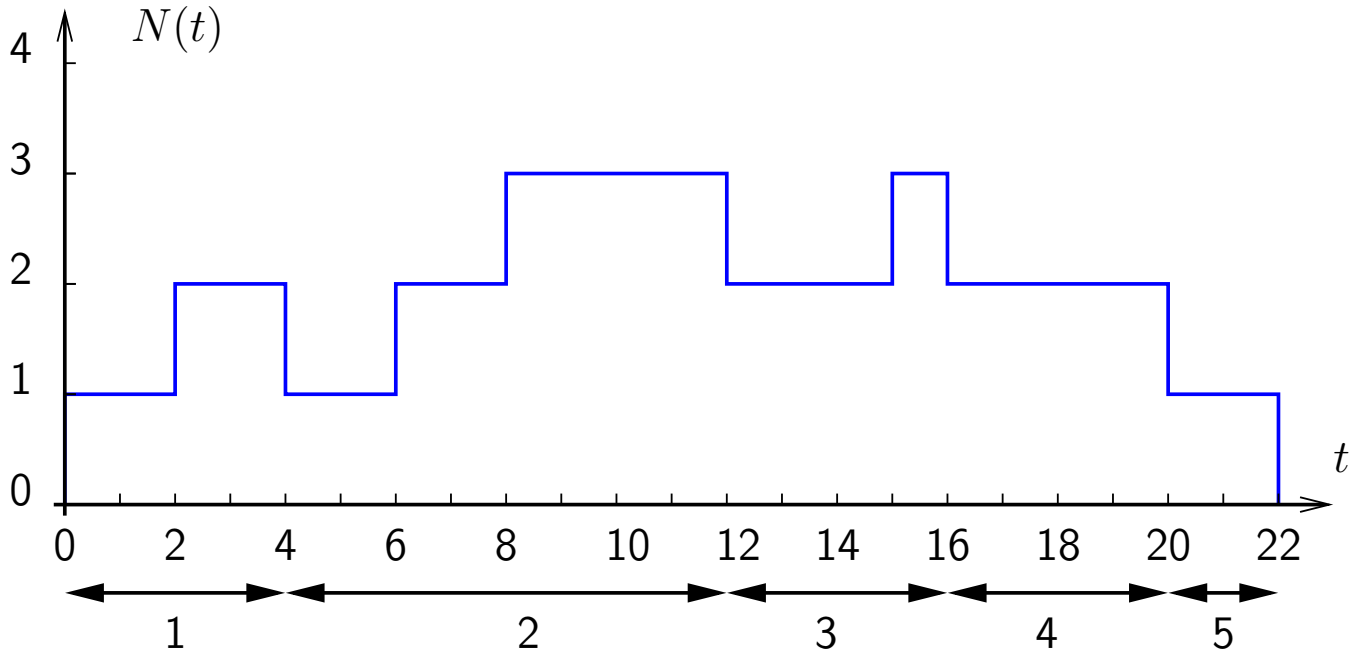
all this, at time t.

Dynamics of a queue

Consider customers with the following characteristics:

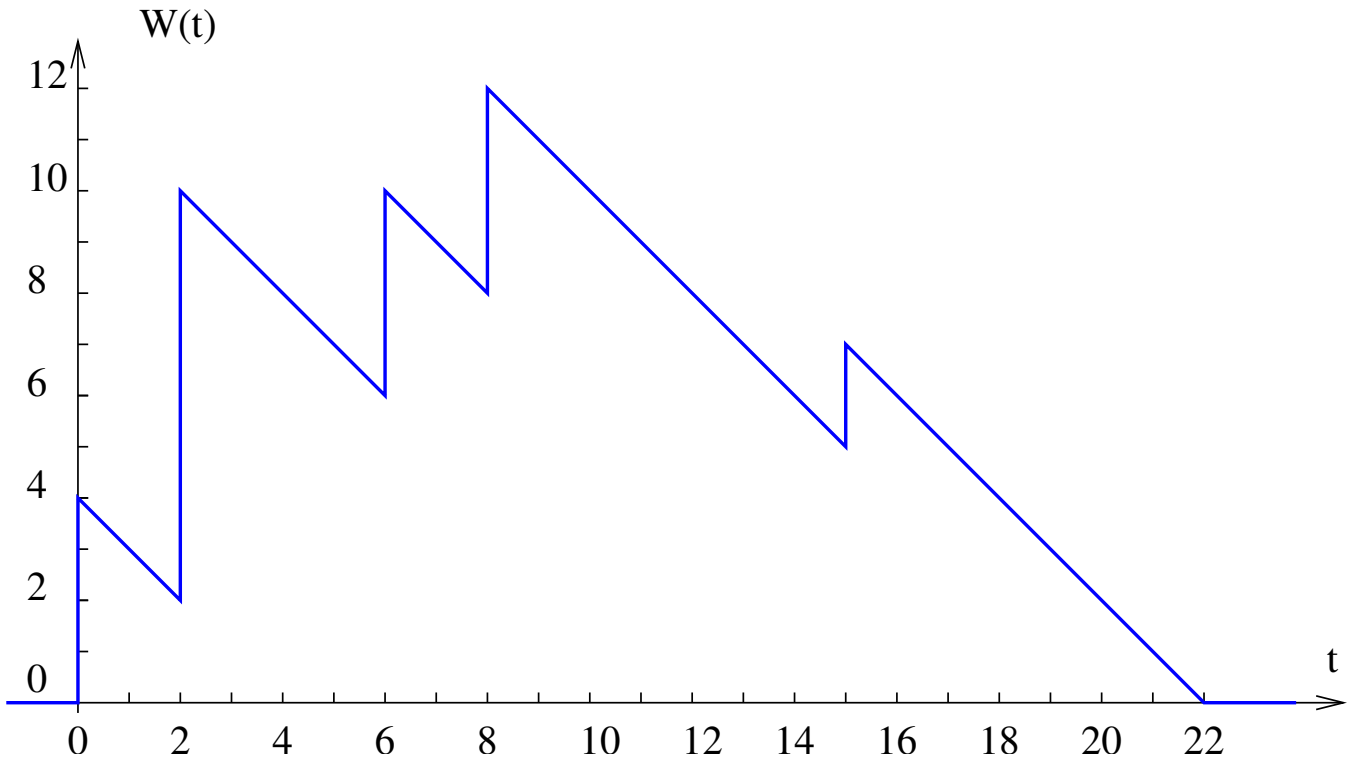
Number	1	2	3	4	5
Arrival	0	2	6	8	15
Service	4	8	4	4	2

Evolution of $N(t)$, the number of customers: if FIFO service,



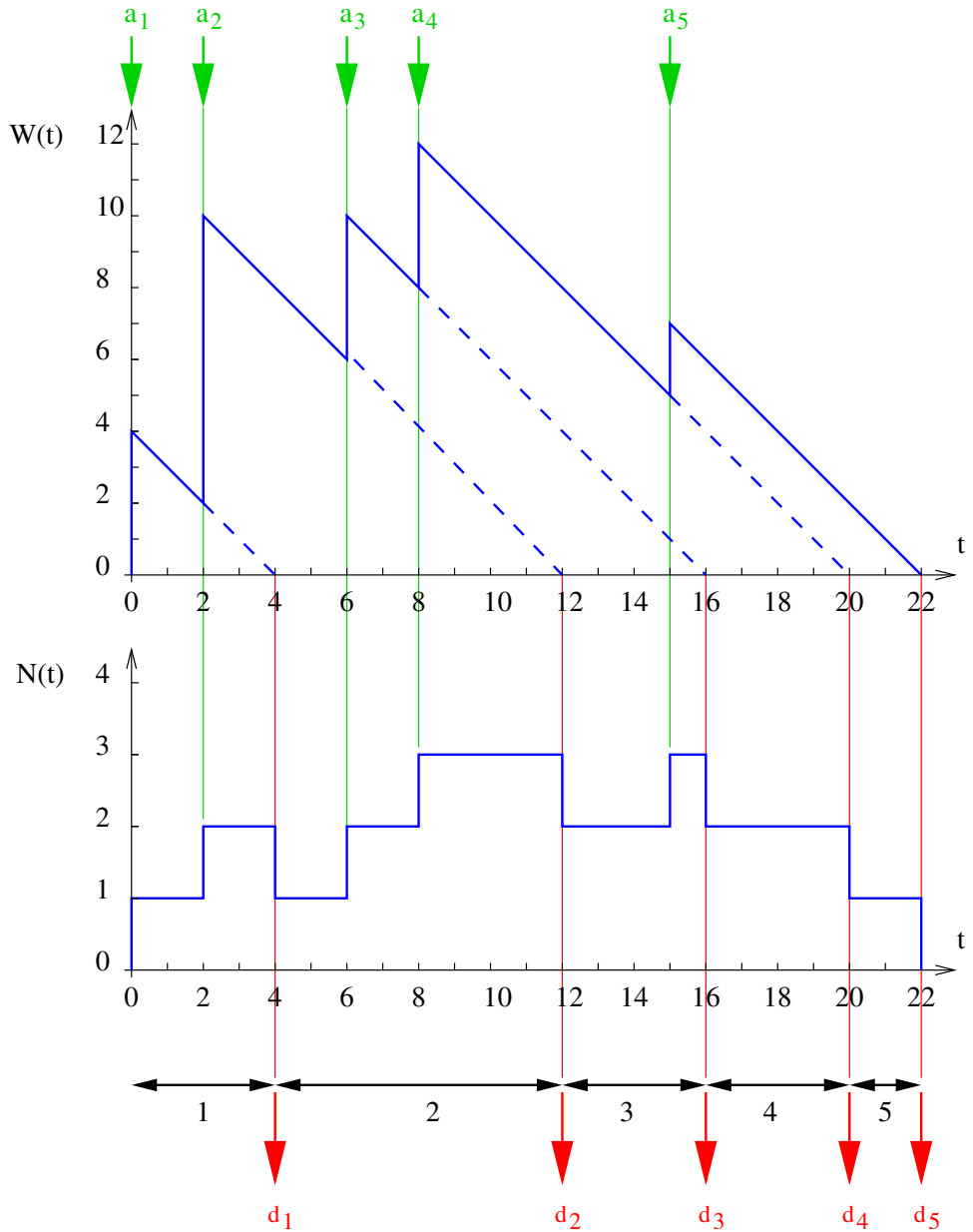
Dynamics of the workload

Evolution of $W(t)$, the workload:



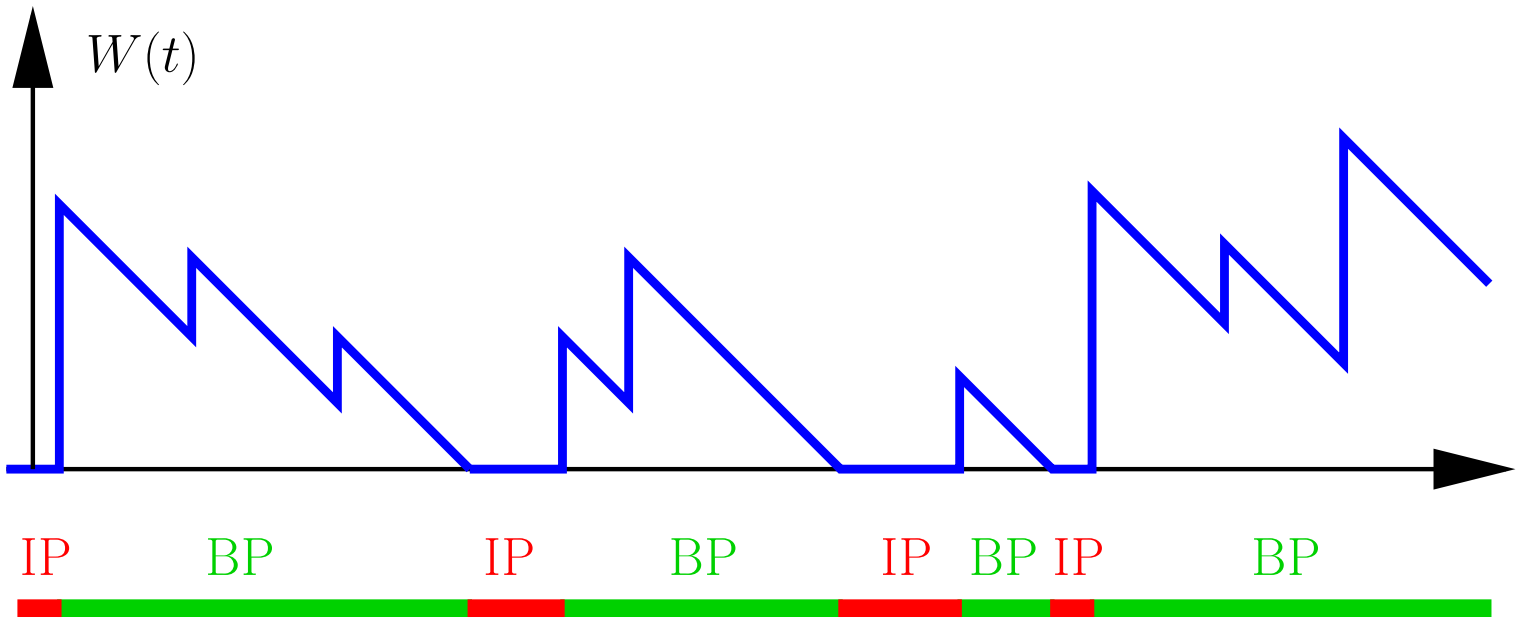
Jumps upwards at times $0, 2, 6, 8, 15$ with steps $4, 8, 4, 4, 2$.

Compared evolution of $N(t)$ and $W(t)$:



Idle periods, busy periods

In general, the typical evolution of $W(t)$ is of the form:



It alternates busy periods and idle periods.

Arrival process

The sequence of arrival times $a_1, a_2, \ldots, a_n, \ldots$ forms the **arrival process**.

It is often characterized by the sequence of **inter-arrival times**: $\tau_n = a_{n+1} - a_n$. For instance, for the Poisson process, the sequence $\{\tau_n; n \in \mathbb{N}\}$ is i.i.d. with distribution:

$$\mathbb{P}\{\tau_n \leq x\} \;=\; 1 \;-\; e^{-\lambda x}\;,$$

where $\lambda > 0$ is a parameter: the rate of the process.
If the process has i.i.d. inter-arrivals τ, then

$$\lambda \;=\; \frac{1}{\mathbb{E}\tau}.$$

In discrete time, the process can be characterized by the sequence $\{A_n; n \in \mathbb{N}\}$ where A_n is the number of arrivals in "slot" n.

A grouped arrival is called a batch.

Services

The service requirement of a customer is a random variable: σ_n for the n-th customer.

Typical distributions in queueing theory are:

- the exponential distribution:

$$\mathbb{P}\{\sigma_n \leq x\} \;=\; 1 \;-\; e^{-\mu x} \;,$$

  for some $\mu > 0$. The average service time is then $\mathbb{E}\sigma = 1/\mu$.
- the Erlang distribution (sum of exponential distributions)
- the Hypergeometric distribution (mixtures of exponentials)
- the deterministic distribution
- many others.

Number of servers

The queue may have one or several servers.

- single-server queue: the standard situation
- multi-server queue: one single queue for all servers
  - NOT to be confused with several single-server queues in parallel
  - servers may be heterogeneous in speed
- infinite-server queue
  - no waiting
  - think of a population of independent individuals

Service discipline

The way waiting customers enter service is specified by the **service discipline** (or **service policy**, **scheduling policy**).
Some usual service disciplines:

- FCFS (First-Come-First-Served): service in the order of arrival
- FIFO (First-In-First-Out): the common name for FCFS
- LIFO (Last-In-First-Out): the stack
- PS (Processor Sharing): customers get *simultaneously* some share of the resource
  Some variants, especially when several classes:
  - egalitarian PS
  - weighted PS
  - Head-of-the-Line PS
- Priorities: each class of customers has a priority level. Those with the highest priority are served first. FIFO among customers with same priority.

Service discipline (ctd.)

Other service discipines, whith some sort of priority

- Based on the service duration
  - SPT (Shortest Processing Time),
  - SRPT (Shortest Remaining Processing Time),
  - LPT, etc.
- Based on a deadline (real-time policies):
  - EDF (Earliest Deadline First), aka EDD (Earliest Due Date) and many more
  - LLF (Least Laxity First), etc.

Service discipline: preemption

In certain service disciplines such as:

- priority-based
- LIFO, SPT, SRPT, ...

a customer that arrives may have a higher priority than the customer currently in service.

non-préemption nothing happens until the end of current service

preemption the service is interrupted;
when interrupting tasks are over:

resume service will continue from where it has stopped

restart service will start over

kill! service will never start again!

Complications with service

Idle servers are costly so:

- servers may take vacations
  - once
  - repeated as long as nobody to serve
- servers may poll several waiting rooms according to several policies
  - visiting queues: cyclic, greedy, random, ...
  - serving queues: limited, exhaustive, gated, ...

Queueing capacity and admission/rejection policy

The size of the buffer may be finite of infinite.

When finite: **system capacity** (including servers) or **buffer capacity** (without servers). "**queue capacity**" is ambiguous, usually refers to system capacity.

In case of finite capacity, what happens to a customer which arrives when the queue is full?

- rejection of the newcomer
- eviction of the last in queue (push-out): the queue behaves as a stack
- eviction of a customer of lesser priority
- eviction at random
- ...

In case of *batch arrivals*:

- rejection of the whole batch
- rejection of customers which fit in queue

Other mechanisms

Impatience: customers do not like waiting

- behavior at the queue
  - balking: not entering the queue if too much waiting is expected
  - reneging: leaving the waiting room (or service) if too much waiting
- behavior after balking/reneging
  - permanent departure
  - resubmission after some "orbit"

Redundant jobs:

- submit the same service request to different queues
- cancel on start, cancel on completion

Jockeying...

Necessity of a notation

Queues are not simple in/out linear systems...

The quality of service in a queue may depend on

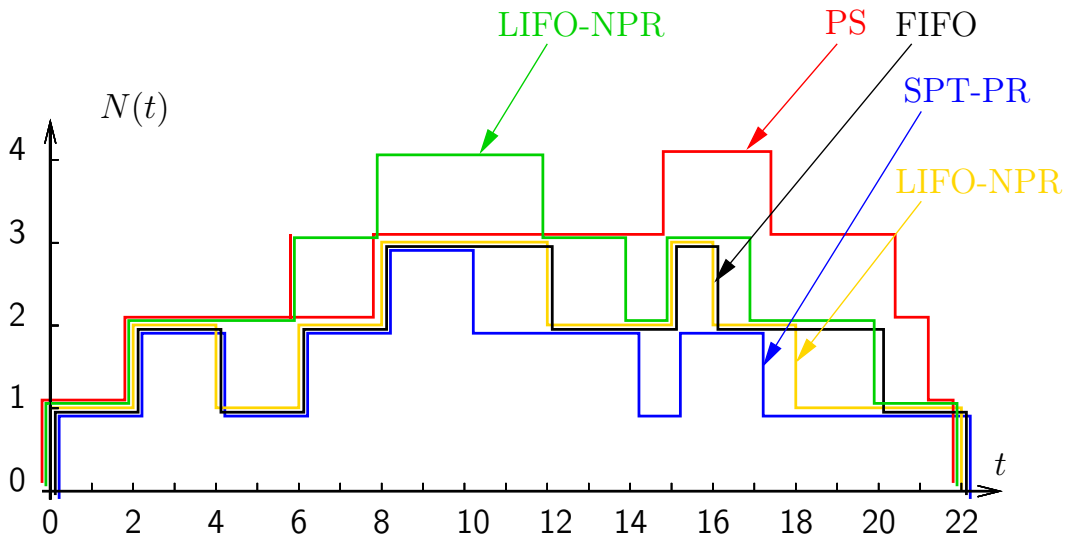
- the distribution of the arrival process
  $\rightarrow$ *not only* its average throughput
- the distribution of the service times,
  $\rightarrow$ *not only* its average, variance.
- the number of servers, the size of the waiting room, the service discipline...

$\Longrightarrow$ necessity to specify clearly the parameters of a queue.

Example: dependence on the service discipline

Illustration of this idea: with the data previously introduced.

Compared evolution of $N(t)$ for different service discipline + statistics



Policy	Avg	Var
FIFO	8,6	7,8
LIFO-PR	10,8	61,8
LIFO-NPR	8,2	21,8
SPT-PR	7,2	42,6
PS	11,3	26,1

$\Longrightarrow$ Necessity to specify clearly the queueing model.

Kendall's notation

This notation allows to identify certain queues among the variety of possibilities.
A queueing model is denoted by:

$$A/S/P/K/D$$

- A the inter-arrival distribution
- S the service time distribution
- P the number of servers
- K the system capacity (by default: ∞)
- D the discipline of service (by default: FIFO)

Code for Kendall's notation

Usual codes for distributions:

- D : deterministic/Dirac
- M : exponential distribution (Markov), Poisson process
- E : Erlang distribution
- H : Hyperexponential distribution
- G : general distribution
- GI : general distribution, i.i.d. sequence
- PH : phase-type distribution
- MMPP : Markov-modulated Poisson process
- BMAP : Batch Markov Arrival process
- ...

Common examples of queues in Kendall's notation

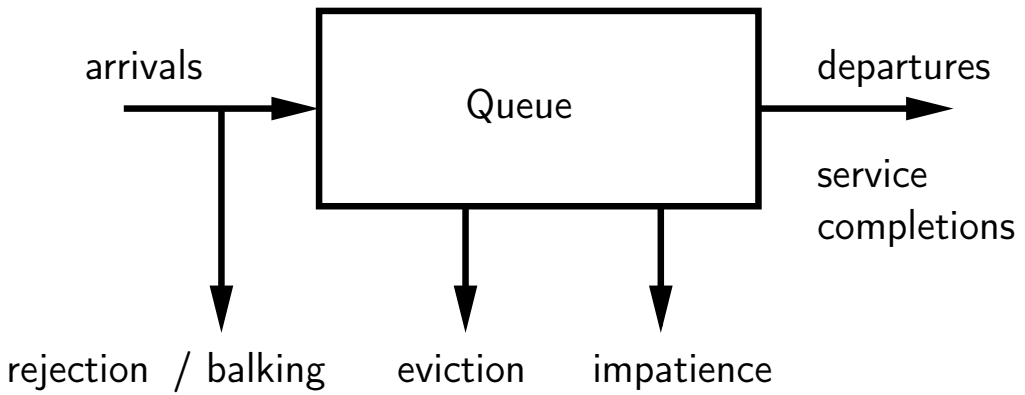
The most useful queues:

- M/M/1 (abbreviation of M/M/1/∞/FIFO)
- M/M/1/K (equivalent to M/M/1/K/FIFO)
- M/GI/1
- M/M/K/K
- GI/M/1
- M/GI/∞

Networks of Queues

Most interesting systems have several resources $\rightarrow$ several queues.

Each queue is an input/ouput system for customers: possibility to interconnect them.



Specifying customer movement: two main ways

- probabilistic routing, possibly with change of class
- deterministic routing depending on the class

Jackson Networks

Jackson Networks are networks of queues with probabilistic routing.

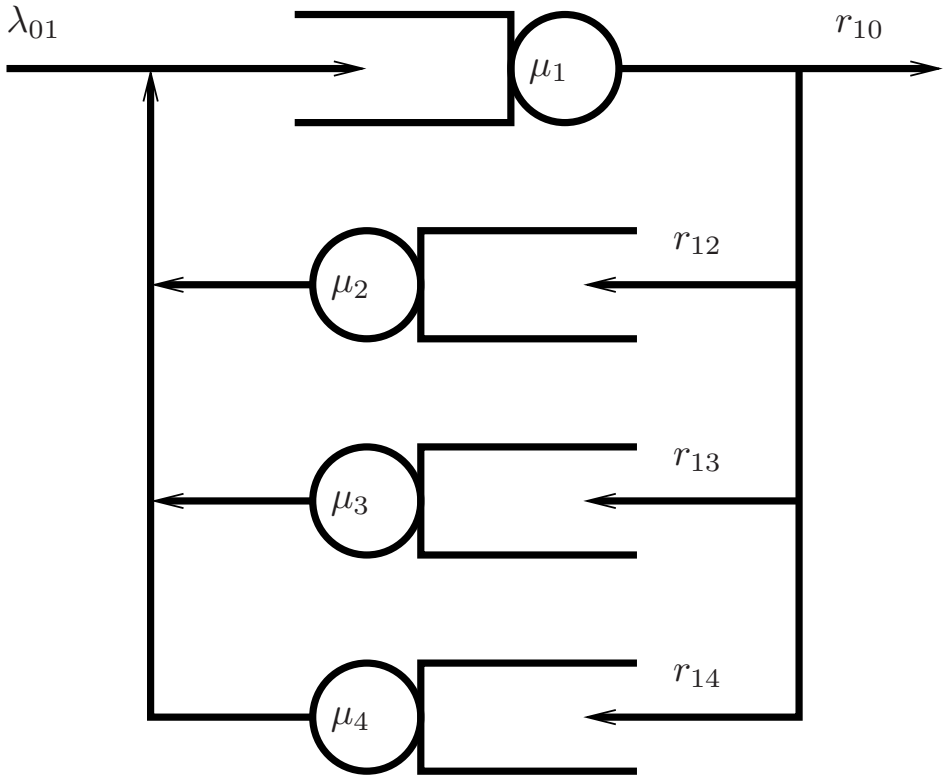
Specification of a Jackson Network:

- N "nodes" which are single-server queues with a duration of service $\sim$ Exp,
- one single class of customers
- the service rates at each node $(\mu_1, \ldots, \mu_N)$
- a vector of arrival rates $\boldsymbol{\lambda}_0 = (\lambda_{0,1}, \ldots, \lambda_{0,N})$ for Poisson processes of customers at each node
- a square matrix $N \times N$ of routing probabilities $\mathbf{R}$:

$$r_{i,j} = \mathbb{P}\{\text{a customer which exits node } i \text{ goes to } j\} \ .$$

Example of Jackson Network

The "central server" model:



Kelly Networks

Specification of a Kelly Network

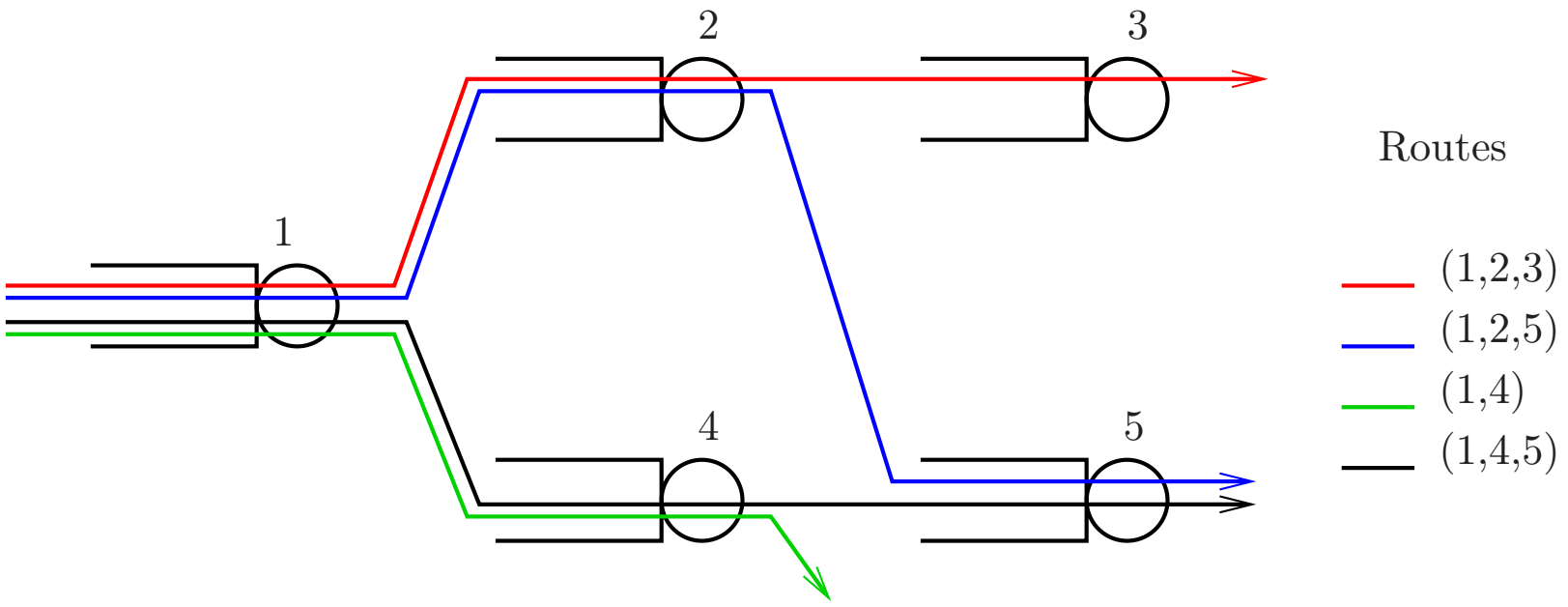
- customers belong to different classes
- to each cass corresponds a *route* in the network:

$$r_k = (r_k^1, \ldots, r_k^{n_k}) \ .$$

- customers arrive at route r_k according to a Poisson process with rate λ_k customers/s
- the service duration in node j is $\sim Exp(\mu_j)$

Example of a Kelly network

Example with 5 nodes (queues) and 4 routes.



This route topology is feed-forward: no loops.

Extensions

Other networks of queues

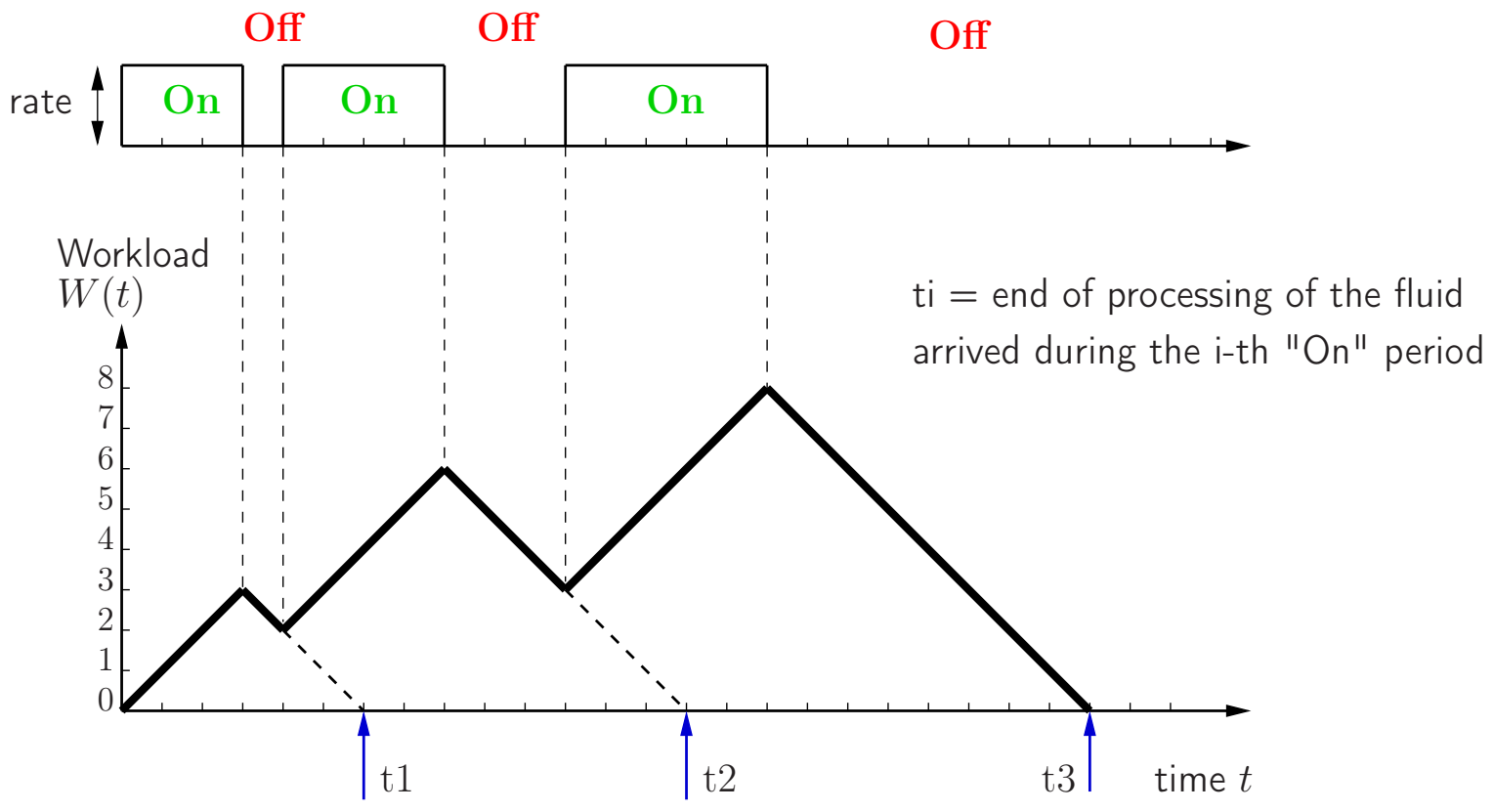
- BCMP Networks
  - multiple classes
  - multiple servers, infinite servers
  - processor sharing discipline
- Whittle Networks
- G-Networks with negative customers
- ...

Other sorts of queues

Fluid queues: no more individual customers, but a "fluid" of work

- arriving with an instantaneous rate $r(t)$ (variable)
- being served at an instantaneous rate $C(t)$.

Example: an ON/OFF process with constant rates:



Part 2:
Metrics associated with queueing

Metrics associated with queuing

Table of contents:

- stability of queues
- throughputs and their different kinds
- conservation laws for throughputs
- waiting times, response times
- occupancy: distribution and average, Little's formula

Metrics

General metrics apply to queues or network of queues

capacity	related to stability
throughputs	débit d'arrivée, débit de sortie, débit offert, débit efficace, débit perdu
times	time spent in the system, time spent waiting
loss rates/probabilities	in case of finite capacities, or impatiences
utilization	of the server, of the waiting room

Stability

Stability refers to the system "not exploding".

Mathematically: from some sequences of random variables (see w_n or r_n below), or the processes $\{N(t); t \in \mathbb{R}\}$ $\{W(t); t \in \mathbb{R}\}$: they have a proper limit when n or t goes to infinity.

Principal stability result

The $G/G/1$ queue is stable if and (almost) if

$$\boxed{\mathbb{E}\sigma \quad < \quad \mathbb{E}\tau}$$

Consequences:

- the queue has a service capacity

$$C_{max} \quad = \quad \frac{1}{\mathbb{E}\sigma} \text{ customers/s}$$

- it is stable if and only if

$$\lambda \quad = \quad \frac{1}{\mathbb{E}\tau} \quad < \quad C_{max} \ .$$

Throughputs

The throughput is a number associated with a stream of "events" occurring at discrete times: $t_1, t_2, \ldots, t_n, \ldots$.

Mathematically, it is defined for **stationary** processes as:

$$\boxed{\theta \;=\; \frac{\mathbb{E}N(0,T)}{T} \;=\; \mathbb{E}N(0,1)\;,}$$

where $N(0,T)$ is the (random) number of events in the interval $[0,T)$.

Empirically, it is measured as:

$$\theta \;\sim\; \frac{N(0,T)}{T}\;.$$

Throughput is akin to a speed, a frequency. Measured in customers/s, events/s, etc. Sometimes called "rate".

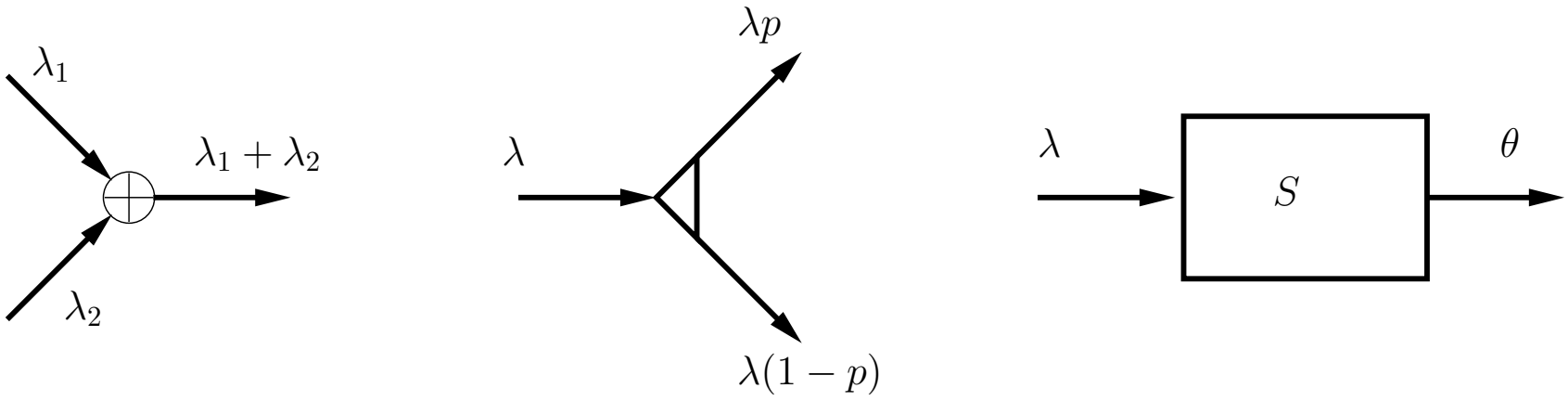
Many sort of throughputs

In a queue

- offered throughput, arrival rate
- departure rate
  - global
  - of successful customers: goodput
- rejection rate, reneging rate, impatience rate
- ...

Conservation of throughputs

Conservation law: throughputs are conserved



Laws of conservation of throughputs: merge, split and I/O

Input/Output law: for System S:

- if S is stable, input throughput and output throughput coincide: $\theta = \lambda$;
- otherwise, customers accumulate at rate $\lambda - \theta$
- $\theta = \max\{\lambda, C_{max}\}$.

Flow conservation in networks of queues

Application to networks of queues: traffic equations and stability conditions.

In both Jackson and Kelly networks, as long as no customers are lost and no accumulation occurs, it is possible to write traffic equations using the laws of conservation.

For Jackson Networks: let $\hat{\lambda}_i$ be the arrival rate to node i: and $\boldsymbol{\lambda} = (\hat{\lambda}_1, \ldots, \hat{\lambda}_N)$.

- The traffic equations write as:

$$\boldsymbol{\lambda} = \boldsymbol{\lambda}_0 + \boldsymbol{\lambda}\,\mathbf{R}$$

$$\hat{\lambda}_i = \lambda_{i,0} + \sum_{j=1}^{N} \lambda_j r_{ji} \ .$$

- The stability condition for the network is:

$$\forall 1 \leq i \leq N, \qquad \hat{\lambda}_i < \mu_i.$$

Traffic equations in Kelly networks

In a Kelly Network with routes $(r_1, \ldots, r_K)$:

- the throughput of customers of class/route k entering node i is:

$$\hat{\lambda}_{ik} \;=\; \lambda_k \;\times\; (\text{numberof } i \text{ in } r_k).$$

- the total arrival rate to node i:

$$\hat{\lambda}_i \;=\; \sum_{i \ni k} \lambda_k \;\times\; (\text{nombre de } i \text{ in } r_k)\ .$$

- the stability condition:

$$\forall 1 \leq i \leq N, \qquad \hat{\lambda}_i \;<\; \mu_i.$$

Waiting times, Response times

Definitions:

response time r_n = total time spent in the system, between arrival time and departure time

waiting time w_n = total time spent in the waiting room (not being served)

Those are related by:

$$r_n = w_n + \sigma_n$$

General results: Little's formula

The following relationship holds for general systems.

Little's formula

The average response (sojourn) time R and the average number of customers N in a stable system where customers arrive at rate λ, are related by the formula:

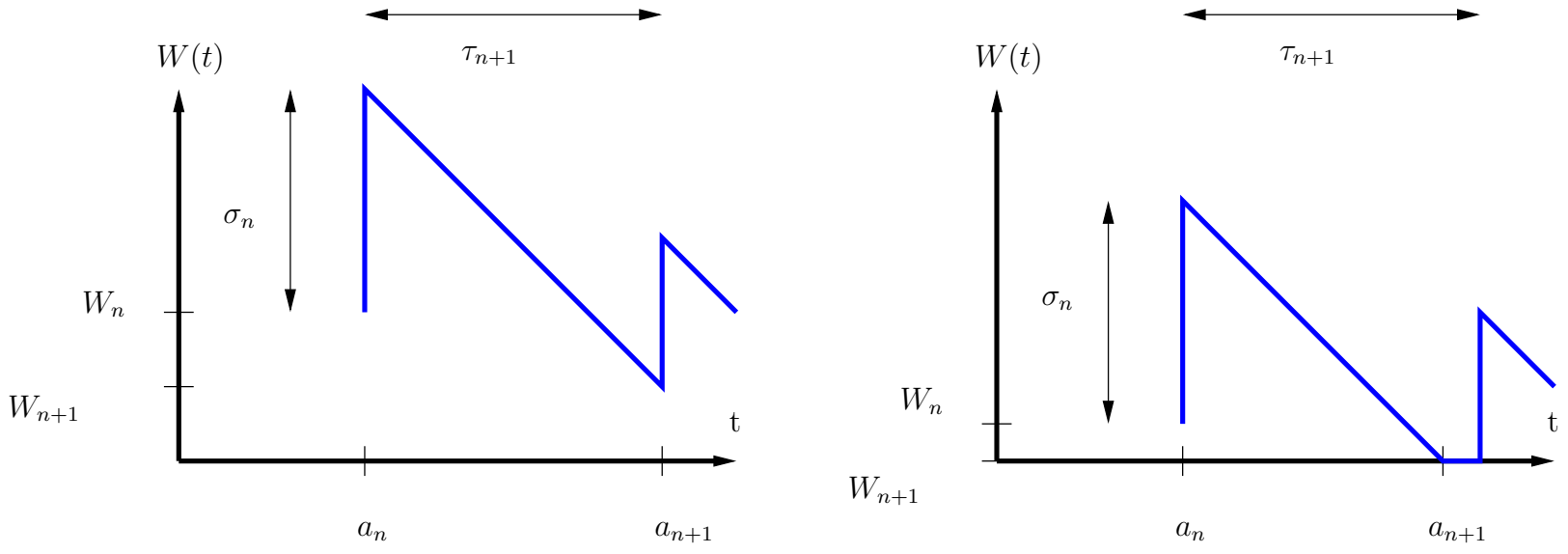
$$\boxed{\lambda\, R \;=\; N}$$

Waiting time: the FIFO case

w_n: waiting time of customer number n before entering service
σ_n: service duration

Lindley's equation

$$W_{n+1} = [W_n + \sigma_n - \tau_{n+1}]^+.$$



Virtual waiting time, real waiting time

If a_n is the arrival time of customer n and if FIFO:

$$W_n \;=\; W(a_n^-)\;.$$

$\Longrightarrow$ $W(t)$ is also called: virtual waiting time.

Warning! W_n and $W(t)$ do not necessarily have the same distribution!

PASTA Property (Poisson Arrivals See Time Averages)

If arrivals are Poisson, the stationary distributions of W_n and $W(t)$ do coincide.

Utilization

Utilization of a resource is the fraction of the time some resource is used.

Let $U(0, T)$ be the total quantity of resource used in interval $[0, T)$.
Then the utilization of the resource is:

$$\rho = \text{limsup}_{T \to \infty} \frac{U(0, T)}{T} .$$

Consquence of Little's formula:

Utilization in the $G/G/1$ queue

The utilization of the server in a stable $G/G/1$ queue is:

$$\rho = \frac{\lambda}{C_{max}} = \lambda \, \mathbb{E}\sigma$$

Other metrics

Other metrics are sometimes of interest:

jitter : a measure of the variation of response times $\{r_n\}$

relaxation time : a measure of the time it takes for the system to becom stationary

busy periods : distribution, average, variance of the length of busy periods

Part 3:
Modeling queueing systems with Markov chains

Modeling queueing systems with Markov chains

Table of contents:

- basic modeling of queues (in discrete time): system state, evolution equations
- crash course on discrete-time Markov chains
  - definition, transition matrix, state probabilities
  - transient and stationary distributions: path and matrix formulas, equilibrium equations
  - classification of states
  - Markov Reward Processes and the evaluation of values

Modeling queueing systems with Markov chains

Table of contents (ctd.)

- modeling example: the single-server queue with iid arrivals
  - construction of the Markov chain
  - equilibrium equations
  - solution via generating functions
  - variations on the model

$\rightarrow$ to be developed in the Case Study after the pause.

Back to the dynamics of queues

The evolution of queues can sometimes conveniently be represented with equations. We have seen the recurrence on waiting times for FIFO queues:

$$w_{n+1} = [w_n + \sigma_n - \tau_{n+1}]^+ .$$

Consider a discrete-time queue, and Q_n the number of customers in queue at slot n. The natural evolution equation is:

$$Q_{n+1} = Q_n + A_n - D_n ,$$

with A_n the number of arrivals and D_n the number of departures. This depends on the scheduling policy and the amount of customers.

Such equations generate a stochastic *process* $\{Q_n; n \in \mathbb{N}\}$. Computing its properties is "solving the queueing problem".
When this process is a *Markov process*, this is easier.

Discrete Time Markov chains

Consider a sequence of random variables $\{X(n)\} \subset \mathcal{E}$, where $\mathcal{E}$ is some discrete set.

Definition of Discrete-Time Markov Chains

$\{X(n), n \in \mathbb{N}\}$ is a homogeneous **discrete time Markov chain** if:

i/ (Markov property) $\forall t \in \mathbb{N}$, et $\forall \ (j_0, j_1, \ldots, j_t, j_{t+1}) \in \mathcal{E}^{t+2}$:

$$\begin{aligned} &\mathbb{P}\{X(t+1) = j_{t+1} | X(t) = j_t, \ldots, X(0) = j_0\} \\ &= \ \mathbb{P}\{X(t+1) = j_{t+1} | X(t) = j_t\} \ ; \end{aligned}$$

ii/ (homogeneity) $\forall t \in \mathbb{N}$, et $(i, j) \in \mathcal{E} \times \mathcal{E}$,

$$\mathbb{P}\{X(t+1) = j | X(t) = i\} \ = \ P_{i,j} \ .$$

$P_{i,j}, (i, j) \in \mathcal{E} \times \mathcal{E}$: transition probabilities

P *transition matrix*.

Dynamics of probabilities

One looks for *transition probabilities at n steps*:

$$p(i,j;n) \;=\; \mathbb{P}\{X(n)=j \mid X(0)=i\}\ ,$$

Let $P(n)$ be the matrix of $p(i,j;n)$. Then:

$$\boxed{P(n) \;=\; \mathbf{P}^n.}$$

Let now, for $n \in \mathbb{N}$ and $j \in \mathcal{E}$,

$$\pi_n(j) \;=\; \mathbb{P}\{X(n)=j\}.$$

Then:

$$\pi_n(j) \;=\; \sum_{i\in\mathcal{E}} \pi_0(i)\ p(i,j;n)\ .$$

Algebraic form: for any $n \in \mathbb{N}$:

$$\boxed{\boldsymbol{\pi}_n \;=\; \boldsymbol{\pi}_0\ \mathbf{P}^n.}$$

The graph of a Markov Chain

To every probability transition matrix $\mathbf{P} = ((p_{ij}))_{\mathcal{E}\times\mathcal{E}}$, is associated a weighted (valued) directed graph $G = (V, E, W)$:

- $V = \mathcal{E}$
- $E = \{(i,j) \in V \times V, p_{ij} > 0\}$
- $W : E \to \mathbb{R}$, $(i,j) \mapsto p_{ij}$.

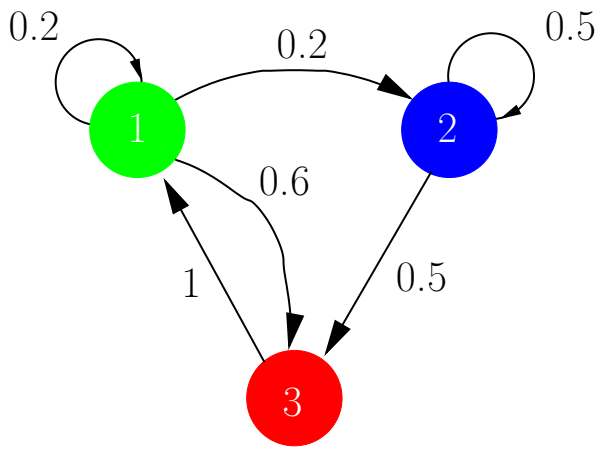
Dynamics of probabilities (ctd.)

Another view of the formula $P(n) = \mathbf{P}^n$: the path formula.

Let G be the graph of $\mathbf{P}$.
For all $n \in \mathbb{N}$, $(i,j) \in \mathcal{E} \times \mathcal{E}$, we have:

$$\begin{aligned} p(i,j;n) &= \sum_{(i_1,\ldots,i_{n-1}) \in \mathcal{E}^{n-1}} p_{i,i_1}\ p_{i_1,i_2}\ \cdots\ p_{i_{n-1},j} \\ &= \sum_{(i,i_1,\ldots,i_{n-1},j) \text{ path in } G} p_{i,i_1}\ p_{i_1,i_2}\ \cdots\ p_{i_{n-1},j}\ . \end{aligned}$$

Example of Markov chain



Graph, or transition diagram

Transition Matrix:

$$\mathbf{P} = \begin{pmatrix} 0.2 & 0.2 & 0.6 \\ 0 & 0.5 & 0.5 \\ 1 & 0 & 0 \end{pmatrix} .$$

Example of Markov chain (ctd.)

Probability vectors:

$$\begin{array}{lcl}
\pi_0 & = & (1, 0, 0) \\
\pi_1 & = & (0.2, \quad 0.2, \quad 0.6) \\
\pi_2 & = & (0.64, \quad 0.14, \quad 0.22) \\
\pi_3 & = & (0.348, \quad 0.198, \quad 0.454) \\
\pi_4 & = & (0.5236, \quad 0.1686, \quad 0.3078) \\
\vdots & \vdots & \vdots \\
\pi_\infty & = & (5/11, 2/11, 4/11).
\end{array}$$

Is this behavior common to all Markov chains?

Equilibrium equations

We have:

$$\boldsymbol{\pi}_{n+1} \;=\; \boldsymbol{\pi}_n \, \mathbf{P}.$$

If $\lim_n \boldsymbol{\pi}_n = \boldsymbol{\pi}$ exists, then:

$$\boxed{\boldsymbol{\pi} \;=\; \boldsymbol{\pi} \, \mathbf{P} \; .}$$

These equilibrium equations are written: $\forall i \in \mathcal{E}$,

$$\pi(i) \;=\; \sum_{j \in \mathcal{E}} \pi(j) \; P_{j,i} \; .$$

They define the stationary probability.
The computation of stationary probabilities is reduced to the solution of a linear system!

Example of Markov chain (ctd.)

In the example:

$$\mathbf{P} = \begin{pmatrix} 0.2 & 0.2 & 0.6 \\ 0 & 0.5 & 0.5 \\ 1 & 0 & 0 \end{pmatrix} .$$

Indeed,

$$(5/11, 2/11, 4/11) \cdot \begin{pmatrix} 0.2 & 0.2 & 0.6 \\ 0 & 0.5 & 0.5 \\ 1 & 0 & 0 \end{pmatrix} = (5/11, 2/11, 4/11).$$

It is the only solution to $\pi\mathbf{P} = \pi$ that is a probability vector.

Classification of states

The behavior of $p(i,j,n)$ as $n \to \infty$ depends primarily on the properties of the graph $G = (V, E)$ (*without* the weights).

Let

- $V = C_1 \cup C_2 \cup \ldots \cup C_q$ be the partition in strongly connected components
- $\widetilde{G} = \{\widetilde{V}, \widetilde{E}\}$ the quotient graph induced by this partition: it is acyclic.

Then

Classification of states

- if C_k is a leaf in $\widetilde{G}$, states of C_k are called recurrent
- othewise, states of C_k are called transient and $p(i,j,n) \to 0$ as $n \to \infty$, for all i and all $j \in C_k$.

Further concepts for Markov Chains

A Markov chain is said to be:

irreducible if its graph has only one strongly connected component

aperiodic if the greatest common divisor of all cycle lengths in its graph is 1.

Asymptotic behavior

The set $\mathcal{E}$ is finite.
Let $\mathbf{P}$ be a finite stochastic matrix, irreducible and aperiodic. Let $\boldsymbol{\pi}$ be the left eigenvector of $\mathbf{P}$ for the eigenvalue 1, such that $\boldsymbol{\pi}.\mathbf{1} = 1$. Then

$$\lim_{n\to\infty} \mathbf{P}^n \;=\; \mathbf{1}.\boldsymbol{\pi} \;.$$

Consequently, for any vector π_0, one has:

$$\begin{aligned} \lim_{n\to\infty} \boldsymbol{\pi}_n &= \boldsymbol{\pi} \\ \lim_{n\to\infty} p(i,j,n) &= \boldsymbol{\pi}(j) \quad \forall i,j \in \mathcal{E} \;. \end{aligned}$$

The convergence occurs geometrically fast:

$$||\boldsymbol{\pi}_n - \boldsymbol{\pi}|| \;=\; \mathcal{O}(\rho^n)$$

for any ρ such that $|\lambda_2| < \rho < 1$, where λ_2 is the eigenvalue of $\mathbf{P}$ of largest modulus after 1.

Markov reward processes

In general, the Markov chain $\{X(n); n \in \mathbb{N}\}$ is not what is of interest in itself. In addition, there is a mechanism of accumulation of costs or rewards.

To each state of the Markov chain $\{X(n), n \in \mathbb{N}\}$ is associated a (possibly random) reward $R(n)$ with the rules:

- the distribution of $R(n)$, conditioned on $\{X(n) = i\}$, is some $F_i(\cdot)$
- the rewards at different steps are conditionnally independent.

Markov Reward Process

The process $(X(n), R(n))$ is a Markov Reward Process.

Statistics of Markov Reward Processes

We are interested in:

- the expected discounted reward: for each $i \in \mathcal{E}$,

$$V(i) = \mathbb{E}\left[\sum_{n=0}^{\infty} \beta^n R(n) \middle| X(0) = i\right]$$

- the reward rate or average reward:

$$\rho(i) = \lim_{T\to\infty} \frac{1}{T}\, \mathbb{E}\left[\sum_{n=0}^{T-1} R(n) \middle| X(0) = i\right]$$

Computation of Markov Reward statistics

Use the column vector notation:

$$R = ((\mathbb{E}R|X = i))_{i\in\mathcal{E}} \qquad V = ((V(i)))_{i\in\mathcal{E}}.$$

Discounted rewards

The vector $V = ((V(i)))_{i\in\mathcal{E}}$ is the solution to the linear system:

$$V \;=\; R + \beta\, \mathbf{P} V \;.$$

Average rewards

If the matrix $\mathbf{P}$ is irreducible, the quantity $\rho(i)$ is independent on i and is given by:

$$\rho \;=\; \boldsymbol{\pi}.R$$

where $\boldsymbol{\pi}$ is the stationary distribution of the chain.

Again linear algebra problems.

Part 4:
Numerical solution and simulation

Numerical solution and simulation

Table of contents:

- Numerical methods: direct solution of linear systems, iterative solutions: power method, Jacobi-like methods
- Monte-Carlo Simulation: pseudo-random numbers, simulation of generic Markov Chains, simulation via evolution equations
- Application to sampling from complicated distributions

Numerical computation of metrics

As observed above:

- the computation of stationary probabilities is reduced to the solution of a linear system

Also:

- many performance metrics (throughputs, average queue size, average waiting time, ...) can be deduced from the stationary distribution because they are some sort of Markov reward
- other performance metrics, such as hitting times, are also computed with linear systems.

Linear systems are easy! Problem solved?
Some remaining issues:

- the system is often very large
- it may even be infinite!

How to get distributions of a DTMC?

Two main approaches to obtain a stationary distribution from the matrix $\mathbf{P}$:

1. numerical solution of the system $\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{P}$
2. Monte-Carlo simulation of the Markov Chain + statistics

Numerical solutions

Solving the system: $\boldsymbol{\pi}\mathbf{P} = \mathbf{P}$ and $\pi.\mathbf{1} = 1$.

- Direct Gaussian Elimination
  $\rightarrow$ adapted to small state spaces
- Iterative methods:
  $\rightarrow$ adapted to medium state spaces
  - power method: $\boldsymbol{\pi}_{n+1} = \boldsymbol{\pi}_n.\mathbf{P}$
  - relaxation method:

$$\pi^{(k+1)}(i) = \sum_{j<i} \pi^{(k+1)}(j) P_{ji} + \sum_{j \geq i} \pi^{(k)}(j) P_{ji} \ .$$

  - Jacobi-style methods: $\mathbf{P} = \mathbf{M} + \mathbf{N} \implies \boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{N}(\mathbf{I} - \mathbf{M})^{-1}$
    $\rightarrow$ typically faster convergence
  - many others...

Monte-Carlo simulation

Monte-Carlo simulation consists in generating trajectories of the Markov-Chain $\{X_n; n \in \mathbb{N}\}$.

Then statistics are performed on the sample. Obvious example: the empirical distribution of one sample.

$$\hat{p}_i = \frac{1}{N+1} \sum_{n=0}^{N} \mathbf{1}_{\{X(n)=i\}} .$$

Other possibility, with many independent samples and some large N:

$$\hat{p}_i = \frac{1}{K} \sum_{k=1}^{K} \mathbf{1}_{\{X^{(k)}(N)=i\}} .$$

Main issues:

- computation time
- convergence to the stationary regime: when can one assume that $X(n)$ is a sample from the stationary distribution?

Pseudo-random numbers

Principle: given that $X_n = i \in \mathcal{E}$, the distribution of X_{n+1} is given by $\{P_{ij}; j \in \mathcal{E}\}$.
Sampling from a discrete distribution is the basic step. The following (naive) algorithm does it. It just needs a `Unif([0,1])` pseudo-random numbers generator.

Algorithm: Function `DiscreteSample`

Data: A discrete distribution: array of n probabilities p_i and values $\mathrm{v}[i]$
Result: A random value equal to $\mathrm{v}[k]$ with probability p_k
begin
  $U \leftarrow \mathtt{Unif}([0,1])$
  $j \leftarrow 1$
  while $U > p[j]$ **do**
    $U \leftarrow U - \mathrm{p}[j]$
    $j \leftarrow j + 1$
return $\mathrm{v}[j]$

Generic simulation algorithm

The basic algorithm returns a trajectory of predefined length.

Data: A transition matrix $\mathbf{P}$, in the form of $|\mathcal{E}|$ distributions $\mathbf{p}_i$, $i \in \mathcal{E}$
Data: An initial distribution $\boldsymbol{\pi}_0$
Data: A time horizon $N \in \mathbb{N}$
Result: A sequence of $N+1$ states $x[i]$, $0 \leq i \leq N$
begin
  $x[0] \leftarrow \texttt{DiscreteSample}(\boldsymbol{\pi}_0)$
  for n **from** 1 **to** N **do**
    $nextDist \leftarrow \mathbf{p}_{x[n-1]}$
    $x[n] \leftarrow \texttt{DiscreteSample}(nextDist)$
    // insert here any processing on sample $x[n]$
  // insert here any processing on the sequence $x[0:N]$
return $x[0:N]$

Applications of Monte-Carlo simulation

It was claimed earlier that sampling from a discrete distribution is easy... but this is not always so. For instance: too many states...
In that case, the MCMC (Markov chain Monte-Carlo) helps. Assume one wants to sample from distribution $f(\cdot)$.

- construct a Markov Chain that has f a a stationary distribution
- simulate the Markov Chain for N steps
- return the state of the Markov Chain.

The transition probabilities of the Markov chains, denoted with $q(y|x)$ are called instrumental distributions.

The (generic) Hastings-Metropolis algorithm

Data: A distribution function $f(\cdot)$
Data: A family of instrumental distributions $q(\cdot|x)$
Data: A time horizon $T \in \mathbb{N}$
Result: A sample of the distribution $f(\cdot)$
begin
  $x \leftarrow$ `InitialSample()`
  for n **from** 1 **to** T **do**
    $y =$ `DiscreteSample`$(q(\cdot|x))$
    $r \leftarrow \rho(x, y)$ // using (1)
    $u \leftarrow$ `Unif`$([0, 1])$
    if $u < r$ **then**
      $x \leftarrow y$ // keep sample y
    else
      $x \leftarrow x$ // reject sample y, keep x
return x

$$\rho(x, y) = \min\left\{\frac{f(y)}{f(x)}\frac{q(x|y)}{q(y|x)}, 1\right\} . \quad (1)$$

Tuning the Hastings-Metropolis algorithm

The instrumental distributions must be chosen such that:

1. it must be algorithmically easy to draw samples from the distribution $q(\cdot|x)$ (small requirement in memory, small computational time);
2. the function $\rho(x, y)$ must be easy to calculate;
3. the resulting Markov chain must be irreducible and aperiodic;
4. rejections do not happen too frequently;
5. convergence do the stationary distribution is fast.

Two solutions comply with requirements 1, 2 and 3:

- independent sampling: $q(y|x) = q(y)$
- random walk sampling: $q(x + 1|x) = q(x - 1|x) = 1/2$
- random walk + teleportation: mixture of both (see the PageRank algorithm)

The (generic) Gibbs sampling algorithm

When the distribution has several dimensions, and each one-dimensional marginal is "easily" simulated, Gibbs' algorithm can be used.

Data: A multidimensional distribution function $f(x_1, \ldots, x_p)$
Data: A time horizon $T \in \mathbb{N}$
Result: A sample of the distribution $f(\cdot)$
begin
  $(x_1, \ldots, x_p) \leftarrow$ `InitialSample()`
  for n **from** *1* **to** T **do**
    for k **from** *1* **to** p **do**
      $x_k \leftarrow$ `DiscreteSample`$(f(x_1, \ldots, x_{k-1}, x_{k+1}, \ldots, x_p))$
return $(x_1, \ldots, x_p)$

Gibbs sampling (ctd.)

Principle:

- each coordinate of the vector $(x_1, \ldots, x_p)$ is updated in sequence
- the updated coordinates are used (cf. relaxation)
- the new values are always accepted

The resulting process $x^{(1)}, {}^{(2)}, \ldots, x^{(T)}$ is a Markov chain with stationary distribution f: if at somme time t the vector $x^{(t)}$ is distributed according to f, each sample of the following loop has the same distribution.

Uniform sampling of discrete structures

A particular class of matrices: the bistochastic matrices

$$\mathbf{1}'\mathbf{P} = \mathbf{1}' .$$

Let

- $\mathcal{E}$ be a set of objects from which we need uniform samples
- $\mathcal{S} = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$ a set of *bijections* of $\mathcal{E}$.
- $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_k)$ a probability distribution over $\mathcal{S}$

Sampling algorithm by random walk

Data: A set $\mathcal{S}$ of bijections of $\mathcal{E}$, and a distribution α on the set $\mathcal{S}$
Data: A time horizon $N \in \mathbb{N}$
Result: A sample of the uniform distribution on $\mathcal{E}$
begin
  $x \leftarrow$ `InitialSample()`
  for n **from** 1 **to** N **do**
    $p = \texttt{DiscreteSample}(\alpha)$
    $x = \sigma_p(x)$
return x

Principle: this realizes a Markov chain which has a bistochastic matrix. The uniform distribution is its stationary distribution.