



---

# LINMA2470 Stochastic Modelling

---

SIMON DESMIDT

Academic year 2024-2025 - Q2

# Contents

<b>1</b>	<b>Reminders</b>	<b>3</b>
1.1	General properties of probability . . . . .	3
1.2	Expectation and variance . . . . .	3
1.3	Law of large numbers . . . . .	4
1.4	Central limit theorem . . . . .	4
1.5	Exponential distribution . . . . .	4
1.6	Erlang distribution . . . . .	4
<b>2</b>	<b>Poisson Processes</b>	<b>5</b>
2.1	Distribution of $N(t)$ . . . . .	6
2.2	Non-homogenous Poisson processes . . . . .	7
2.3	Bernoulli process approximation . . . . .	7
2.4	Classification of queueing systems . . . . .	8
<b>3</b>	<b>Renewal Processes</b>	<b>9</b>
3.1	Strong law of large numbers . . . . .	9
3.2	Central limit theorem . . . . .	9
3.3	Stopping time . . . . .	10
3.4	Blackwell's renewal theorem . . . . .	10
3.5	Renewal reward process . . . . .	11
3.6	Little's Law . . . . .	13
<b>4</b>	<b>Finite State Markov Chains</b>	<b>15</b>
4.1	Definitions . . . . .	15
4.2	Transition probabilities . . . . .	16
4.3	Markov chains with rewards . . . . .	16
4.4	Markov decision processes . . . . .	17
4.5	Dynamic programming algorithm for the stationary optimal policy . . .	18
<b>5</b>	<b>Markov Decision Processes and Reinforcement Learning</b>	<b>19</b>
5.1	MDP and Policies . . . . .	19
5.2	Optimizing Policies . . . . .	21
5.3	Value Iteration Algorithm . . . . .	22
5.4	Linear programming approach . . . . .	22
5.5	Policy iteration algorithm . . . . .	23
5.6	Reinforcement Learning . . . . .	23

<b>6</b>	<b>Countable-state Markov Chains</b>	<b>25</b>
6.1	Definitions . . . . .	25
6.2	Birth-death Markov chains . . . . .	26
6.3	Processor sharing . . . . .	27
<b>7</b>	<b>Markov processses</b>	<b>29</b>
7.1	Semi-Markov processes . . . . .	29
7.2	Markov processes . . . . .	29
7.3	Markovian queueing models . . . . .	31
7.4	Reversibility . . . . .	33
7.5	More complex queues . . . . .	34
7.6	Above and beyond . . . . .	36
<b>8</b>	<b>Fluid approximation for non-stationary queueing systems</b>	<b>37</b>
8.1	Transient behaviour . . . . .	37
8.2	Steady-state approximation . . . . .	38

# Reminders

## 1.1 General properties of probability

- $P[A \cup B] = P[A] + P[B] - P[A \cap B]$ ;
- $P[A|B] = \frac{P[A \cap B]}{P[B]} = \frac{P[AB]}{P[B]}$ ;
- $A$  and  $B$  are independent iff  $P[AB] = P[A]P[B] \implies P[A|B] = P[A]$ ;
- $P[X \leq x] = F_X(x)$  is the distribution function, i.e. a monotone increasing function of  $x$  going from 0 to 1 when  $x$  goes from  $-\infty$  to  $+\infty$ .
- Its derivative is the density function  $f_X(x)$  such that  $f_X(x)\delta \approx P[x \leq X \leq x + \delta]$  for an infinitesimal  $\delta$ .
- A random variable  $X$  is said to be memoryless if  $\forall t, x > 0, P[X > t + x | X > t] = P[X > x]$ .
- Markov inequality (for a nonnegative random variable):  $P[Y \geq y] \leq \frac{\mathbb{E}[Y]}{y}$ ;
- Chebyshev inequality:  $P[|Z - \mathbb{E}[Z]| \geq \varepsilon] \leq \frac{\sigma_Z^2}{\varepsilon^2}$ ;

## 1.2 Expectation and variance

- For a discrete random variable,  $\mathbb{E}[X] = \sum_{n=-\infty}^{\infty} nP[X = n] = \sum_{n=-\infty}^{\infty} P(X > n)$ ;
- For a continuous random variable,  $\mathbb{E}[X] = \int_{-\infty}^{\infty} xf_X(x)dx$ ;
- $\mathbb{E}[X] = \int_0^{\infty} (1 - F_X(x))dx$ .
- $Var[X] = \sigma_X^2 = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$ ;

The moment generation function is

$$M_X(t) = \mathbb{E}[e^{tX}] = \sum_{i=0}^{\infty} \frac{t^i m_i}{i!} \quad (1.1)$$

where  $m_i = \mathbb{E}[X^i]$  is the  $i$ -th moment.

## 1.3 Law of large numbers

Let  $X_1, \dots, X_n$  be a series of independent and uniformly distributed (IID) random variables with expectation  $\bar{X}$  and finite variance  $\sigma_X^2$ . Let  $S_n = X_1 + \dots + X_n$ . Then,

- Weak version:

$$\lim_{n \rightarrow \infty} P \left[ \left| \frac{S_n}{n} - \bar{X} \right| \geq \varepsilon \right] = 0 \quad (1.2)$$

- Strong version:

$$\lim_{n \rightarrow \infty} P \left[ \sup_{m \geq n} \left( \frac{S_m}{m} - \bar{X} \right) > \varepsilon \right] = 0 \iff \lim_{n \rightarrow \infty} \frac{S_n}{n} = \bar{X} \quad \text{with probability 1} \quad (1.3)$$

## 1.4 Central limit theorem

$$\lim_{n \rightarrow \infty} P \left[ \frac{S_n - n\bar{X}}{\sqrt{n}\sigma} \leq y \right] = \int_{-\infty}^y \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx \quad (1.4)$$

## 1.5 Exponential distribution

- $f_X(x) = \lambda e^{-\lambda x}$ , for  $x \geq 0$ ;
- $F_X(x) = 1 - e^{-\lambda x}$ , for  $x \geq 0$ ;
- $\mathbb{E}[X] = 1/\lambda$ .

→ Note: the exponential distribution is memoryless.

## 1.6 Erlang distribution

The Erlang distribution  $Er(k, \lambda)$  is a sum of  $k$  independent exponential random variables with parameter  $\lambda$  each.

- $\mathbb{E}[X] = k/\lambda$ ;

# Poisson Processes

A Poisson process  $N(t)$  counts the number of arrivals with exponentially distributed inter-arrival times.

$$S_n = \sum_{i=1}^n X_i \quad X_i \sim \exp(\lambda) \quad (2.1)$$

$\forall n, t$ , we have the relation  $\{S_n \leq t\} = \{N(t) \geq n\}$ , where  $S_n$  is a random variable telling at which time the  $n$ -th occurrence appears.

→ Note: a Poisson process is memoryless:  $P[Z_1 > x] = e^{-\lambda x}$ , with  $Z_1$  be the duration of the time interval from  $t$  until the first arrival after  $t$ .

For a Poisson process of rate  $\lambda$ , and any given  $t > 0$ , the length of the interval from  $t$  until the first arrival after  $t$  is an exponentially distributed random variable. This random variable is independent of both  $N(t)$  and of the  $N(t)$  arrival epochs before time  $t$ . It is also independent of  $N(\tau)$ ,  $\forall \tau \leq t$ .

Let us consider the process after  $Z_1$ ,  $Z_m$ , the time until the  $m$ -th arrival after time  $t$ . It is independent of  $N(t)$  and of the entire previous history of the process.

Let us denote  $\tilde{N}(t, t') = N(t') - N(t)$ .

- Stationary increments property: It has the same distribution as  $N(t' - t)$ ,  $\forall t' \geq t$  (stationary increments property);
- Independent increments property: For any sequence of times  $0 < t_1 < \dots < t_k$ , the set  $\{N(t_1), \tilde{N}(t_1, t_2), \dots, \tilde{N}(t_{k-1}, t_k)\}$  is a set of independent random variables.

From the memoryless property, here is another definition of a Poisson process:

- A Poisson process is a counting process that has the stationary and independent increment properties and such that

$$\begin{aligned} P[\tilde{N}(t, t + \delta) = 0] &= 1 - \lambda\delta + o(\delta) \\ P[\tilde{N}(t, t + \delta) = 1] &= \lambda\delta + o(\delta) \\ P[\tilde{N}(t, t + \delta) \geq 2] &= o(\delta) \end{aligned} \quad (2.2)$$

## 2.1 Distribution of $N(t)$

$S_n$  is the sum  $n$  IID random variables and  $f_{S_n}$  is the convolution of  $n$  times  $f_X$ :

$$f_{S_n}(t) = \frac{\lambda^n t^n e^{-\lambda t}}{(n-1)!} \quad (2.3)$$

From this,

$$P[N(t) = n-1] = \frac{(\lambda t)^n e^{-\lambda t}}{(n)!} \quad (2.4)$$

and finally,

$$\mathbb{E}[N(t)] = \lambda t \quad \text{Var}[N(t)] = \lambda t \quad (2.5)$$

From equation (2.4), the Poisson process verifies the following probability conditions:

- $P[\tilde{N}(t, t+\delta) = 0] = 1 - \lambda\delta + o(\delta);$
- $P[\tilde{N}(t, t+\delta) = 1] = \lambda\delta + o(\delta);$
- $P[\tilde{N}(t, t+\delta) \geq 2] = o(\delta);$

where we use a first-order approximation of the exponential term, with  $o(\delta)$  its residual. As  $o(\delta)$  is negligible, we can approximate the Poisson process as a Bernoulli process.

### 2.1.1 Combining Poisson processes

Let  $N_1(t)$  and  $N_2(t)$  be two independent Poisson processes. Let the process  $N(t) = N_1(t) + N_2(t)$ . We can show using the three properties above that  $N(t)$  is a Poisson process with rate  $\lambda_1 + \lambda_2$ .

### 2.1.2 Subdividing a Poisson process

Let  $N(t)$  be a Poisson process with rate  $\lambda$ . We split the arrivals in 2 subprocesses  $N_1(t)$  and  $N_2(t)$ . Each arrival of  $N(t)$  is sent to  $N_1(t)$  with probability  $p$  and to  $N_2(t)$  with probability  $(1-p)$ , each split being independent from all others.

Then, the resulting processes  $N_1(t)$  and  $N_2(t)$  are two independent Poisson processes with respective rate  $p\lambda$  and  $(1-p)\lambda$ .

### 2.1.3 Conditional arrival distribution

The density probability function when we have  $n$  Poisson processes, under the condition that  $N(t) = n$ , is

$$f(s_1, \dots, s_n | N(t) = n) = \frac{n!}{t^n} \quad (2.6)$$

From the previous results, we can compute that

$$P[S_1 > \tau | N(t) = n] = \left( \frac{t-\tau}{t} \right)^n \quad (2.7)$$

and the expectation is

$$\mathbb{E}[S_1|N(t) = n] = \frac{t}{n+1} \quad (2.8)$$

And from this, we derive that

$$P[X_i > \tau|N(t) = n] = \left(\frac{t-\tau}{t}\right)^n \quad (2.9)$$

with expectation

$$\mathbb{E}[X_i] = \frac{t}{n+1} \quad (2.10)$$

And thus the density function is

$$f_{S_i}(x|N(t) = n) = \frac{x^{i-1}(t-x)^{n-i}n!}{t^n(n-i)!(i-1)!} \quad (2.11)$$

## 2.2 Non-homogenous Poisson processes

A non-homogenous Poisson process  $N(t)$  is a counting process with increments that are independent but not stationary, with

- $P[\tilde{N}(t, t+\delta) = 0] = 1 - \lambda(t)\delta + o(\delta);$
- $P[\tilde{N}(t, t+\delta) = 1] = \lambda(t)\delta + o(\delta);$
- $P[\tilde{N}(t, t+\delta) \geq 2] = o(\delta);$

where  $\tilde{N}(t, t+\delta) = N(t+\delta) - N(t)$ . The time-varying arrival rate  $\lambda(t)$  is assumed to be continuous and strictly positive.

## 2.3 Bernoulli process approximation

We can approximate the non-homogenous Poisson process with a Bernoulli process where the time is partitioned into increments of lengths inversely proportional to  $\lambda(t)$  (i.e. using a nonlinear time scale).

- $P[\tilde{N}(t, t+\epsilon/\lambda(t)) = 0] = 1 - \epsilon + o(\epsilon);$
- $P[\tilde{N}(t, t+\epsilon/\lambda(t)) = 1] = \epsilon + o(\epsilon);$
- $P[\tilde{N}(t, t+\epsilon/\lambda(t)) \geq 2] = o(\epsilon);$

Letting  $\epsilon$  tend to zero, we obtain

$$P[N(t) = n] = \frac{(m(t))^n e^{-m(t)}}{n!} \quad P[\tilde{N}(t, t') = n] = \frac{(m(t, t'))^n e^{-m(t, t')}}{n!} \quad (2.12)$$

with

$$m(t) = \int_0^t \lambda(\tau) d\tau \quad m(t, t') = \int_t^{t'} \lambda(\tau) d\tau \quad (2.13)$$



## 2.4 Classification of queueing systems

- We note  $A/B/k$  where  $A$  is the type of distribution for the arrival process,  $B$  for the service time and  $k$  the number of servers.

We suppose that the arrivals wait in a single queue. Commonly used letters are

- M: exponential distribution (for A) or Poisson process (for B);
- D: deterministic time intervals;
- E: Erlang distribution;
- G: general distribution.

# Renewal Processes

A renewal process is a counting process with IID interarrival intervals. We note  $X_i$  the interval between arrivals,  $\bar{X} = \mathbb{E}[X]$  is supposed to be finite with probability  $P[X_i > 0] = 1$ <sup>1</sup>,  $\sigma$  can be finite, and we denote  $S_n = \sum_{i=1}^n X_i$  the time of the  $n$ -th arrival.

## 3.1 Strong law of large numbers

Let  $\{N(t); t \geq 0\}$  be a renewal process, then

$$\lim_{t \rightarrow \infty} N(t) = \infty \quad \lim_{t \rightarrow \infty} \mathbb{E}[N(t)] = \infty \quad (3.1)$$

This implies that

$$\lim_{t \rightarrow \infty} \frac{N(t)}{t} = \frac{1}{\bar{X}} \text{ with probability } 1 \quad (3.2)$$

## 3.2 Central limit theorem

If the interarrival intervals of the renewal process  $N(t)$  have a finite standard deviation, then from the CLT for IID random variables, we have

$$\lim_{n \rightarrow \infty} P \left[ \frac{S_n - n\bar{X}}{\sqrt{n}\sigma} \leq \alpha \right] = \Phi(\alpha) \quad (3.3)$$

where  $\Phi(\alpha)$  is the integral the normal distribution.

$$\lim_{t \rightarrow \infty} P \left[ \frac{N(t) - t/\bar{X}}{\sigma\bar{X}^{-3/2}\sqrt{t}} < \alpha \right] = \Phi(\alpha) \quad (3.4)$$

→ Note: The reliability of the observed mean of successive results that are supposed to be IID depends a lot on the rule used to decide when we stop repeating the experiment.

---

<sup>1</sup>A probability of 1 means that the opposite can happen, but is so rare that the probability is 0.

### 3.3 Stopping time

Let  $N$  be the rv corresponding to the total number of experiments observed. Let  $I_n$  be a series of rv being the indicator function of  $\{N \geq n\}$ :

$$I_n = \begin{cases} 1 & \text{if the } n\text{-th experiment is observed} \\ 0 & \text{otherwise} \end{cases} \quad (3.5)$$

$N$  is a stopping time if  $I_n$  depends only on  $X_1, \dots, X_{n-1}$ . This means that stopping at 3pm, for example, is not a stopping time, because it can depend on  $X_n$ , depending if the  $n$ -th arrival is before or after 3pm.

#### 3.3.1 Wald's inequality

Let  $N$  be a stopping time for  $\{X_n; n \geq 1\}$ . Then,  $\mathbb{E}[S_N] = \mathbb{E}[N]\bar{X}$ .

### 3.4 Blackwell's renewal theorem

#### 3.4.1 Arithmetic distribution

If interarrival intervals can only have a length that is a multiple of some real number  $d$ , the interarrival distribution will be called an arithmetic distribution, and  $d$  the span of the distribution.

#### 3.4.2 Blackwell's inequality

If the interarrival distribution of a renewal process  $N(t)$  is not arithmetic, then

$$\lim_{t \rightarrow \infty} (m(t + \delta) - m(t)) = \frac{\delta}{\bar{X}} \quad \forall \delta \quad (3.6)$$

If the interarrival distribution is arithmetic with span  $d$ , then

$$\lim_{t \rightarrow \infty} (m(t + nd) - m(t)) = \frac{nd}{\bar{X}} \quad \forall n \geq 1 \quad (3.7)$$

where  $m(t) = \mathbb{E}[N(t)]$ .

#### 3.4.3 Relationship with a Poisson process

The sum of many renewal processes tends to a Poisson process: for a non-arithmetic renewal process with  $P[X_i = 0] = 0$ , we have

$$\begin{aligned} \lim_{t \rightarrow \infty} P[N(t + \delta) - N(t) = 0] &= 1 - \delta/\bar{X} + o(\delta) \\ \lim_{t \rightarrow \infty} P[N(t + \delta) - N(t) = 1] &= \delta/\bar{X} + o(\delta) \\ \lim_{t \rightarrow \infty} P[N(t + \delta) - N(t) \geq 2] &= o(\delta) \end{aligned} \quad (3.8)$$

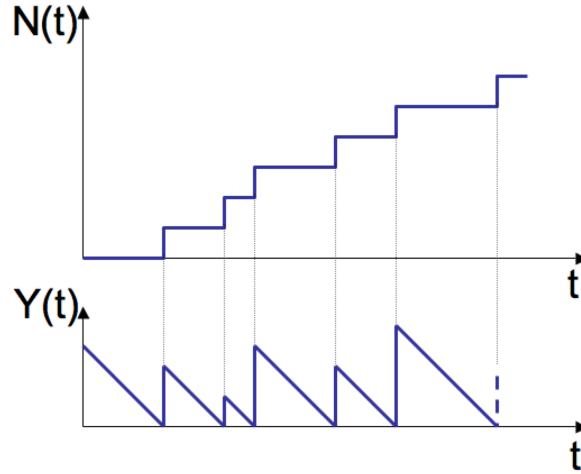
The increments are asymptotically stationary, but not independent.

### 3.5 Renewal reward process

Along to the renewal process  $N(t)$ , we can add a reward function  $R(t)$ . It models the rate at which the process is accumulating a reward or cost. It can however only depend on the current renewal but not the previous ones.

Let  $Y(t)$  be the residual life at time  $t$  for the current renewal:

$$R(t) = Y(t) = S_{N(t)+1} - t \quad (3.9)$$

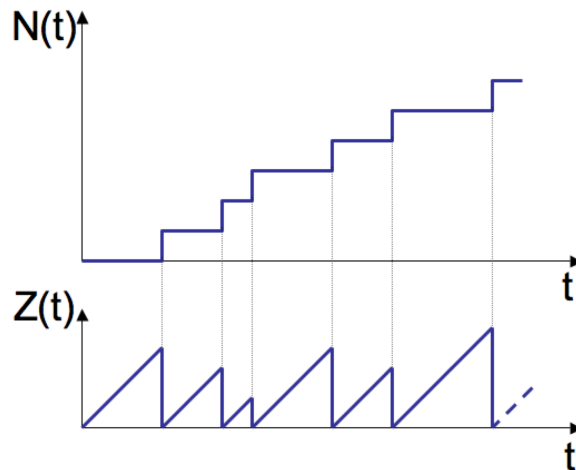


The time average residual life is  $\frac{1}{t} \int_0^t Y(\tau) d\tau$ . From the definition of  $Y(t)$ , we can calculate that

$$\lim_{t \rightarrow \infty} \frac{1}{t} \int_0^t Y(\tau) d\tau = \frac{\mathbb{E}[X^2]}{2\mathbb{E}[X]} = \frac{1}{2}\mathbb{E}[X] + \frac{\text{Var}(X)}{\mathbb{E}[X]} > \frac{1}{2}\mathbb{E}[X] \text{ with probability 1} \quad (3.10)$$

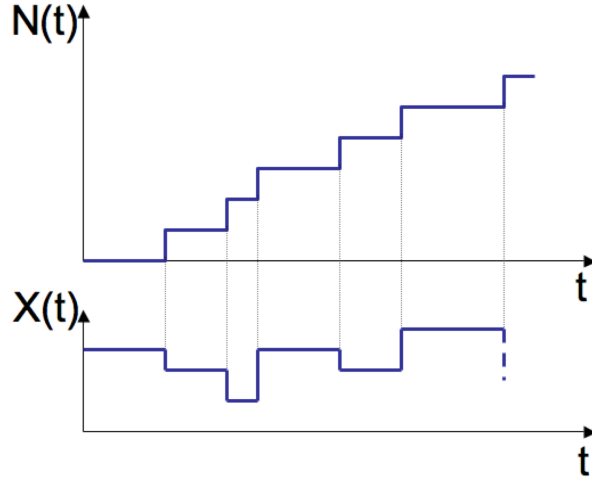
#### 3.5.1 Time average age

Let  $Z(t)$  be the age of the current renewal at time  $t$ :  $R(t) = Z(t) = t - S_{N(t)}$ . The time average age is  $\lim_{t \rightarrow \infty} \frac{1}{t} \int_0^t Z(\tau) d\tau = \frac{\mathbb{E}[X^2]}{2\mathbb{E}[X]}$ .



### 3.5.2 Time average duration

Let  $X(t)$  be the duration of the renewal containing time  $t$ :  $R(t) = X(t) = S_{N(t)+1} - S_{N(t)}$ . The time average duration is  $\lim_{t \rightarrow \infty} \frac{1}{t} \int_0^t X(\tau) d\tau = \frac{\mathbb{E}[X^2]}{\mathbb{E}[X]}$ .



### 3.5.3 General renewal reward functions

Let  $R(t)$  be a reward function for a renewal process with expected inter-renewal times  $\bar{X} < \infty$ , then with probability 1,

$$\lim_{t \rightarrow \infty} \frac{1}{t} \int_0^t R(\tau) d\tau = \frac{\mathbb{E}[R_n]}{\mathbb{E}[X]} \quad (3.11)$$

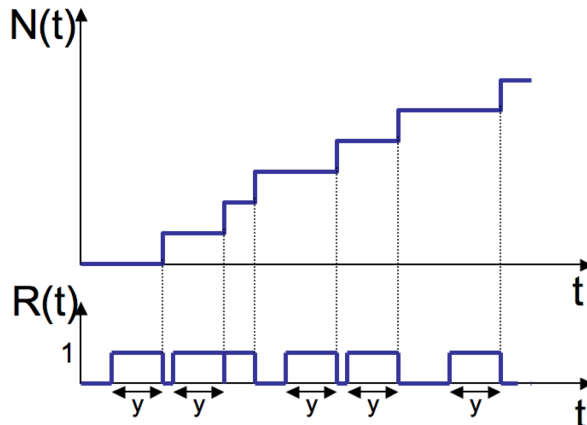
where  $R_n$  is defined as

$$R_n = \int_{S_n}^{S_{n+1}} R(\tau) d\tau \quad (3.12)$$

### 3.5.4 Distribution of residual life

We are interested in the fraction of time that  $Y(t) \leq y$ :

$$R(t) = I\{Y(t) \leq y\} \quad R_n = \min\{y, X_n\} \quad (3.13)$$



And we can calculate that

$$\begin{aligned}\mathbb{E}[R_n] &= \int_0^y P[X > x] dx \\ F_Y(y) &= \frac{1}{\mathbb{E}[X]} \int_0^y P[X > x] dx\end{aligned}\tag{3.14}$$

### 3.5.5 Key theorem

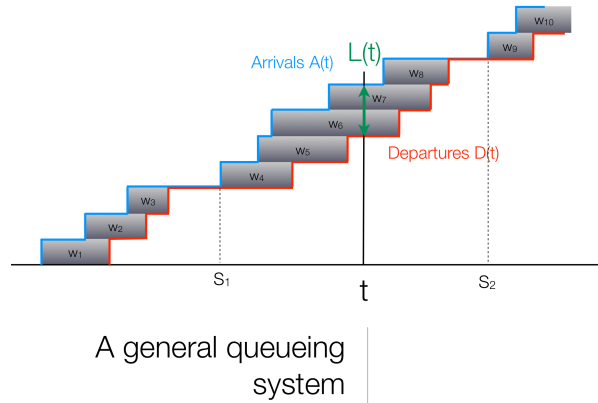
Let  $N(t)$  be a non-arithmetic renewal process, let  $R(z, x) \geq 0$  be such that  $r(z) = \int_{x=z}^{\infty} R(z, x) dF_X(x)$  is directly Riemann integrable. Then,

$$\lim_{t \rightarrow \infty} \mathbb{E}[R(t)] = \frac{\mathbb{E}[R_n]}{\bar{X}}\tag{3.15}$$

## 3.6 Little's Law

Let a queueing system be such that

- $A(t)$  is the number of arrivals between 0 and  $t$ ;
- $D(t)$  is the number of departures between 0 and  $t$ ;
- $L(t) = A(t) - D(t)$  is the number of customers in the system at time  $t$ ;
- $w_i$  the time the  $i^{th}$  customer spends in the system;
- $N(t)$  is the renewal process counting the number of busy periods of the system (each time a customer arrives when the system is empty).



Let us use  $L(t)$  as a reward function for the renewal process  $N(t)$ . This implies

$$\begin{aligned}\sum_{n=1}^{N(t)} R_n &\leq \int_0^t L(\tau) d\tau \leq \sum_{i=1}^{A(t)} w_i \leq \sum_{n=1}^{N(t)+1} R_n \\ \lim_{t \rightarrow \infty} \frac{1}{t} \int_0^t L(\tau) d\tau &= \frac{\mathbb{E}[R_n]}{\mathbb{E}[X]}\end{aligned}\tag{3.16}$$

Putting all this together, we can show that  $\bar{L} = \lim_{t \rightarrow \infty} \frac{1}{t} \int_0^t L(\tau) d\tau = \bar{W}\lambda$ .

### 3.6.1 M/G/1 queue

Let  $R(t)$  be the remaining time for the customer being served. Let  $U(t)$  be the time an arrival at time  $t$  would have to wait before being served. Let  $L_q(t)$  be the number of customers in queue at time  $t$ , independent of the  $Z_i$ . We define

$$U(t) = \sum_{i=1}^{L_q(t)} Z_i + R(t) \implies \mathbb{E}[U(t)] = \mathbb{E}[L_q(t)]\mathbb{E}[Z] + \mathbb{E}[R(t)] \quad (3.17)$$

We can show that

$$\int_0^{S_N(t)} R(\tau) d\tau \leq \int_0^{S_N(t)+1} R(\tau) d\tau \quad (3.18)$$

And from Little's Law,

$$\lim_{t \rightarrow \infty} \mathbb{E}[L_q(t)] = \lambda \bar{W}_q \implies \lim_{t \rightarrow \infty} \mathbb{E}[U(t)] = \lambda \bar{W}_q \mathbb{E}[Z] + \lambda \frac{\mathbb{E}[Z^2]}{2} \quad (3.19)$$

Poisson arrival process implies that arrivals occur with identical probability at any moment, this implies independence with  $U(t)$ . Hence  $\mathbb{E}[W_q(t)] = \mathbb{E}[U(t)]$ . Hence  $\bar{W}_q = \lambda \bar{W}_q \mathbb{E}[Z] + \lambda \frac{\mathbb{E}[Z^2]}{2}$ . And we can isolate  $\bar{W}_q$ :

$$\bar{W}_q = \frac{\lambda(\mathbb{E}[Z]^2 + \sigma^2)}{2(1 - \lambda\mathbb{E}[Z])} \quad (3.20)$$

And we remember  $\bar{W} = \bar{W}_q + \mathbb{E}[Z]$ .

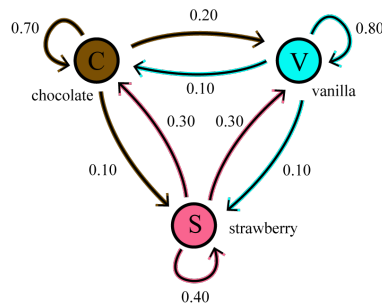
# Finite State Markov Chains

## 4.1 Definitions

A Markov chain is a stochastic process with fixed intervals  $\{X_n, n \geq 0\}$  such that each random variable  $X_n, n \geq 1$  depends on the past only through the most recent random variable  $X_{n-1}$ :

$$P[X_n = j | X_{n-1} = i, X_{n-2} = k, \dots, X_0 = m] = P[X_n = j | X_{n-1} = i] = P_{ij} \quad (4.1)$$

The rv  $X_n$  is called the state of the Markov chain, and the set of possible sample values for the states lie in a countable set. A Markov chain can be represented under a graph or matrix form:



$$P = \begin{pmatrix} 0.70 & 0.20 & 0.10 \\ 0.10 & 0.80 & 0.10 \\ 0.30 & 0.30 & 0.40 \end{pmatrix} \quad (4.2)$$

- We say that a state  $j$  is accessible from  $i$  ( $i \rightarrow j$ ) if there exists a path in the graph from  $i$  to  $j$ :  $i \rightarrow j$  iff  $P_{ij}^n = P[X_n = j | X_0 = i] > 0$  for some  $n$ .
- Two distinct states  $i, j$  communicate ( $i \leftrightarrow j$ ) if  $i$  is accessible from  $j$  and vice versa.
- A class  $C$  of states is a non-empty set of states such that for each  $i \in C$ , each state  $j \neq i$  satisfies  $j \in C$  if  $i \leftrightarrow j$  and  $j \notin C$  if  $i \nleftrightarrow j$ .
- A state  $i$  is recurrent if it is accessible from all states that are accessible from  $i$ . A transient state is a state that is not recurrent.

→ Note: all states of a same class are of the same type.

- A finite-state Markov chain has at least one recurrent class.
- The period of a state  $i$ , denoted  $d(i)$ , is the greatest common divisor of all  $n$  such that  $P_{ii}^n > 0$ . A state is aperiodic if  $d(i) = 1$ .

→ Note: All states of a class have the same periodicity.



- If a class has a period  $d > 1$ , then there exists a partition  $\{C_i\}_{i=1}^d$  of the states of the class such that all the transitions from a state of  $C_n$  go to a state of class  $C_{n+1}$  and all transitions from  $C_d$  go to a state of  $C_1$ , i.e. we make a cycle of subclasses.
- A class is called ergodic if it is aperiodic and recurrent.
- A matrix is stochastic iff it is square, non negative and each row sums to 1, i.e.  $P\mathbb{1}_n = \mathbb{1}_n$ .

## 4.2 Transition probabilities

We can calculate that

$$P[X_{n+2} = j | X_n = i] = P_{ij}^2 = \sum_{k=1}^J P_{ik} P_{kj} \implies P^2 = P \cdot P \implies P^n = P \cdot \dots \cdot P \quad (4.3)$$

More generally,  $P_{ij}^{n+m} = \sum_{k=1}^J P_{ik}^n P_{kj}^m$ .

Because of ergodicity, all rows converge to the same value, and we store those values in a row vector called  $\pi$ . Then,  $\pi = \pi P$  and the sum of all values of  $\pi$  is 1.

From this, we induce that  $\pi$  is a left eigenvector of  $P$  for the eigenvalue 1, and the number of linearly independent solutions corresponds to the multiplicity of the eigenvalue 1. There will be one independent solution for each recurrent class of  $P$ . Moreover, if  $P$  is ergodic, then  $\lim_{n \rightarrow \infty} P^n = \mathbb{1}_m \pi$ , else  $\pi$  will be the average over the different subclasses.

## 4.3 Markov chains with rewards

Let  $r_i$  be the reward associated with state  $i$ . In the case where the reward is on the edge from node  $i$  to  $j$  and not the states, the reward is  $r_{ij}$  and we have  $r_i = \sum_j P_{ij} r_{ij}$ . For an ergodic chain, we observe that the average reward per period will be

$$g = \sum_i r_i \pi_i \quad (4.4)$$

where  $\pi_i$  is the steady state probability.

### 4.3.1 Expected reward over multiple transitions

Let  $X_m$  be the state at time  $m$  and let  $R_m = R(X_m)$  be the reward at time  $m$ . Under the condition  $X_m = i$  (starting point), the aggregate expected reward  $v_i(n)$  over  $n$  periods from  $X_m$  to  $X_{m+n-1}$  is

$$v_i(n) = \mathbb{E}[R(X_m) + \dots + R(X_{m+n-1}) | X_m = i] = r_i + \sum_j P_{ij} r_j + \dots + \sum_j P_{ij}^{n-1} r_j \quad (4.5)$$

And in vector notation

$$v(n) = r + [P]r + \dots + [P^{n-1}]r = \sum_{h=0}^{n-1} [P^h]r \quad (4.6)$$

### 4.3.2 Relative gain vector

Assuming the Markov chain is an ergodic unichain, i.e. it has a single ergodic class with possibly some transient classes, we know that  $\lim_{n \rightarrow \infty} [P^n] = \mathbb{1}_m \pi$  and thus

$$\lim_{n \rightarrow \infty} [P^n]r = \mathbb{1}_m \pi r = g \mathbb{1}_m \quad (4.7)$$

This means that the expected reward per period converges to  $g$ . From this, we can evaluate the transient effect and define the relative-gain vector, denoted by  $w$ :

$$w = \lim_{n \rightarrow \infty} (v(n) - n g \mathbb{1}_m) = \lim_{n \rightarrow \infty} \sum_{h=0}^{n-1} [P^h - \mathbb{1}_m \pi]r \quad (4.8)$$

It can also be computed by solving the following equations instead of calculating the limit:

$$w + g \mathbb{1}_m = [P]w + r \quad \pi w = 0 \quad (4.9)$$

The second equation means that the sum (weighted by the probabilities) of all gains is 0.

## 4.4 Markov decision processes

Suppose that in each state  $i$ , we can choose between  $K_i$  different possibilities with rewards  $r_i^{(1)}, \dots, r_i^{(K_i)}$ . This means that we choose between different Markov chains that have the same states but not necessarily the same edges. We want to find the optimal (stationary or dynamic) policy for this problem.

### 4.4.1 Dynamic programming algorithm for the dynamic optimal policy

This section aims to find a dynamic programming algorithm for the dynamic optimal policy. We assume here that for any policy  $A$ , the resulting Markov chain with matrix  $[P^A]$  is an ergodic unichain.

Let  $v(0)$  be the final reward vector. Through a recurrence method, we can show that

$$v_i^*(n) = \max_k \{r_i^{(k)} + \sum_j P_{ij}^{(k)} v_j^*(n-1)\} \quad (4.10)$$

or in vector form:

$$v^*(n) = \max_A \{r^A + [P^A]v^*(n-1)\} \quad (4.11)$$

for a policy  $A$ , i.e. a decision  $k_i$  for each state  $i$ . From equations (4.9), we know that

$$w^A = r^A - g^A \mathbb{1}_m + [P^A]w^A \quad (4.12)$$

and thus

$$v^A(n) = n g^A \mathbb{1}_m + w^A + [P^A]^n (v(0) - w^A) \quad (4.13)$$

If  $v(0) = w^B$  for a policy  $B$  such that

$$r^B + [P^B]w^B \geq r^A + [P^A]w^A \quad \forall A \quad (4.14)$$

then  $B$  is the dynamic optimal policy for each time period, and

$$v^*(n) = w^B + ng^B \mathbf{1}_m \quad (4.15)$$

This relation is an equivalence.

#### 4.4.2 Policy improvement algorithm

---

##### Algorithm 1 Policy improvement algorithm

---

- 1: **Step 1:** Choose an arbitrary policy  $B$ ;
  - 2: **while**  $\exists A : r^B + [P^B]w^B \stackrel{\leq}{\neq} r^A + [P^A]w^B$  **do**
  - 3:     **Step 2:** Compute  $w^B$ ;
  - 4:     **Step 3:** Find  $A$  such that  $r^A + [P^A]w^B \stackrel{\neq}{\geq} r^B + [P^B]w^B$ ;
  - 5:     **Step 4:**  $B \leftarrow A$ ;
  - 6: **end while**
- 

**Theorem 4.1.** Assuming for any policy  $A$  that the Markov chain  $[P^A]$  is an ergodic unichain, if  $B$  is an optimal stationary policy, then

$$\lim_{n \rightarrow \infty} v^*(n) - ng^B \mathbf{1}_m = w^B + (\beta - \pi^B w^B) \mathbf{1}_m \quad (4.16)$$

What is  $\beta$ ?

### 4.5 Dynamic programming algorithm for the stationary optimal policy

---

##### Algorithm 2 Dynamic programming algorithm for the stationary optimal policy

---

- 1: **Step 1:** Fix an arbitrary vector  $v(0)$ ;
- 2: **while**  $l < u$  **do**
- 3:     **Step 2:** Compute

$$l = \min_i [v_i^*(n) - v_i^*(n-1)] \quad u = \max_i [v_i^*(n) - v_i^*(n-1)] \quad (4.17)$$

- 4: **end while**
  - 5:  $l = u = g^A$  and  $A$  is the optimal stationary policy.
-

# Markov Decision Processes and Reinforcement Learning

## 5.1 MDP and Policies

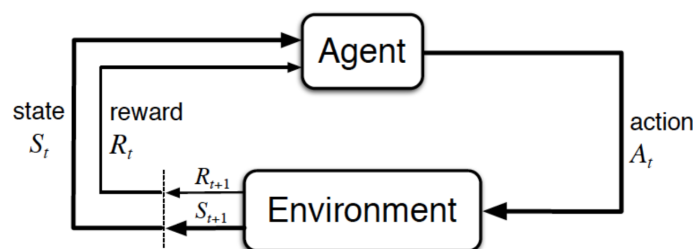
### 5.1.1 Markov Decision Process

A Markov Decision Process models a sequential decision-making process under uncertainty, where moving to the next stage only depends on the current action-state pair.

**Definition 5.1.** A Markov Decision Process (MDP) is defined by

- a set of system states  $S$ ;
- a set of actions  $\mathcal{A}$ ;
- a set of rewards  $R$ ;
- probabilities  $p(s', r|s, a)$  of getting a reward  $r$  and moving to state  $s'$  if action  $a$  is taken in state  $s$ .

The MDP is finite if the three sets are finite.



From the definition, we can derive 3 quantities:

- Transition probabilities:  $p(s'|s, a) = \sum_r p(s', r|s, a)$ ;
- Reward probabilities:  $p(r|s, a) = \sum_{s'} p(s', r|s, a)$ ;
- Expected reward knowing  $s, a$ :  $r(s, a) = \mathbb{E}[R|s, a] = \sum_r r \sum_{s'} p(s', r|s, a)$ .

### 5.1.2 Policies

A policy defines the decision making in each state;

**Definition 5.2.** A policy is a mapping  $\pi : s \in S \rightarrow \pi(a|s)$ , where  $\pi(a|s)$  represents the probability of taking action  $a$  in state  $s$ .

### 5.1.3 Policy values and state-action values

The policy value  $v_\pi(s)$  at a given state  $s$  corresponds to the expected rewards collected over time by applying policy  $\pi$  starting from state  $s$ .

$$v_\pi(s) = \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s \right] \quad (5.1)$$

where  $\gamma \in [0, 1[$  is a discounted factor. The state-action value function is the same idea, but has a dependance on the action we intend to take.

$$q_\pi(s, a) = \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a \right] \quad (5.2)$$

We can show that

$$v_\pi(s) = \mathbb{E}_{A \sim \pi(a|s)} [q_\pi(s, A)] \quad (5.3)$$

Those definitions induce the Bellman equations, a linear system in  $v_\pi(s)$ :

$$\forall s \in S, \quad v_\pi(s) = \sum_a \pi(a|s) \sum_{s'|r} p(s', r | s, a) [r + \gamma v_\pi(s')] \quad (5.4)$$

### 5.1.4 Policy evaluation

Given the policy  $\pi$  and the MDP probabilities  $p$ , we can rewrite the Bellman equations as

$$V = R + \gamma P V \iff (I - \gamma P) V = R \quad (5.5)$$

where  $I - \gamma P$  is invertible as  $\|P\|_\infty \leq 1$  and  $\gamma < 1$ .

Another way to solve this is by an iterative process: introducing the linear operator  $L : V \rightarrow \gamma P V + R$ , we want to find  $V$  such that  $V \approx L(V)$ . This approach is guaranteed to converge since  $L$  is  $\gamma$ -Lipschitz (affine application). Defining  $V^* = \lim_{t \rightarrow \infty} L^t(V_0)$ , with  $V_0$  the first iterate, we can show that

$$\|V^* - V_{n+1}\|_\infty \leq \frac{\gamma}{1 - \gamma} \|V_{n+1} - V_n\|_\infty \quad (5.6)$$

---

**Algorithm 3** Policy Evaluation Algorithm

---

```
1: Input:  $\pi$  the policy to be evaluated, and  $\theta$  the guaranteed accuracy of estimation
2: Initialization:  $V(s)$  the arbitrary initial value for all  $s$ 
3: while  $\Delta \geq \theta$  do
4:    $\Delta = 0$ 
5:   for each state  $s$  do
6:      $v = V(s)$ 
7:      $V(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma V(s')]$ 
8:      $\Delta = \max(\Delta, |v - V(s)|)$ 
9:   end for
10: end while
```

---

## 5.2 Optimizing Policies

### 5.2.1 Policy Improvement Theorem

**Theorem 5.3.**

$$[\forall s \in S, q_\pi(s, \pi^{new}(s)) \geq q_\pi(s, \pi(s))] \implies v_{\pi^{new}}(s) \geq v_\pi(s) \quad (5.7)$$

A strict inequality on the left implies a strict one on the right too. This theorem means that a new policy will be at least as good as a given policy  $\pi$  if changing any action of  $\pi$  by the corresponding action of  $\pi^{new}$  yields a better total gain at the end.

### 5.2.2 Bellman optimality conditions

**Definition 5.4.** A policy  $\pi^*$  is optimal if for any state  $s \in S$  and any other policy  $\pi$ ,  $v_{\pi^*}(s) \geq v_\pi(s)$ .

**Theorem 5.5.** A policy  $\pi$  is optimal iff for any state action pair  $(s, a)$  with a positive probability to be selected by the policy, i.e.  $\pi(a|s) > 0$ , we have

$$a \in \arg \max_{a' \in A} q_\pi(s, a') \quad (5.8)$$

Meaning that any action with a nonzero probability to be taken maximizes the gain of the state at which it is taken.

It can be shown that this implies that any finite MDP admits an optimal policy which is deterministic.

This theorem yields the two following equations for optimal policies:

$$\begin{aligned} v_*(s) &= \max_a q_{\pi^*}(s, a) = \max_{a \in A(s)} \sum_{s',r} p(s',r|s,a)(r + \gamma v_*(s')) \\ q^*(s, a) &= \sum_{s',r} p(s',r|s,a)(r + \gamma \max_{a'} q^*(s', a')) \end{aligned} \quad (5.9)$$

## 5.3 Value Iteration Algorithm

In the same idea as we did in the evaluation section, let us define the operator  $\Phi : V \rightarrow \Phi(V)$  such that

$$\Phi(V)(s) := \max_{a \in A(s)} \sum_{s', r} p(s', r|s, a)(r + \gamma V(s')) \quad (5.10)$$

And thus the Bellman optimality equation ((5.9)) is equivalent to  $V = \Phi(V)$ , which can be solved iteratively from the starting point  $V_0$ . This method is guaranteed because  $\Phi$  is a contracting operator ( $\sim$  L-Lipschitz function).

---

### Algorithm 4 Value Iteration Algorithm

---

```

1: Input: the guaranteed accuracy of estimation  $\theta$ ;
2: Initialization:  $V(s) :=$  an arbitrary initial value for all  $s$ ;
3:    $\Delta \geq \theta$ ;
4: while  $\Delta \geq \theta$  do
5:    $\Delta = 0$ ;
6:   for each state  $s$  do
7:      $v = V(s)$ 
8:      $V(s) = \max_a \sum_{s', r} p(s', r|s, a)(r + \gamma V(s'))$ ;
9:      $\Delta = \max(\Delta, |v - V(s)|)$ ;
10:  end for
11: end while

```

---

## 5.4 Linear programming approach

The problem consists in computing the values of  $v(s)$  for some arbitrary weights  $\alpha(s) > 0$ .

$$\begin{aligned} \min_{v(s)} \quad & \sum_s \alpha(s) v(s) \\ v(s) \geq \quad & \sum_{s', r} p(s', r|s, a)[r + \gamma v(s')] \quad \forall s \in S, a \in A \end{aligned} \quad (5.11)$$

And the dual is

$$\begin{aligned} \max_{x(s, a) \geq 0} \quad & \left[ \sum_{s' \in S, r \in R} p(s', r|s, a) r \right] x(s, a) \\ \sum_{a \in A} x(s, a) = \quad & \alpha(s) + \gamma \sum_{s' \in S, a \in A, r \in R} p(s, r|s', a) x(s', a) \quad \forall s \in S \end{aligned} \quad (5.12)$$

The variables  $x(s, a)$  represent the total discounted probabilities of being in state  $s$  and take action  $a$ .

---

**Algorithm 5** Policy Iteration Algorithm

---

```
1: Input: the guaranteed accuracy of estimation  $\theta$ ;  
2: Initialization:  $A(s) :=$  arbitrary action for all  $s$ ;  
3:  $V(s) :=$  an arbitrary initial value for all  $s$ ;  
4:  $\Delta \geq \theta$ ;  
5: while  $\Delta \geq \theta$  do  
6:    $\Delta = 0$ ;  
7:   for each state  $s$  do  
8:      $v = V(s)$ ;  
9:     OldAction :=  $A(s)$ ;  
10:     $V(s) = \max_a \sum_{s',r} p(s', r|s, a)(r + \gamma V(s'))$ ;  
11:     $A(s) = \arg \max_a \sum_{s',r} p(s', r|s, a)(r + \gamma V(s'))$ ;  
12:     $\Delta = \max(\Delta, |v - V(s)|)$ ;  
13:   end for  
14:   Update the  $V(s)$  by evaluating the new policy.  
15: end while
```

---

## 5.5 Policy iteration algorithm

## 5.6 Reinforcement Learning

### 5.6.1 Temporal Difference Learning

Temporal Difference Learning is used when we do not know the probabilities  $p(s', r|s, a = \pi(s))$ . In that case, we rather use a temporal difference error, measuring the difference between the current estimate  $V(s)$  and  $r + \gamma V(s')$  (see blue term).

---

**Algorithm 6** TD(0) for estimating  $v_\pi$ 

---

```
1: Input:  $\pi$  the policy to be evaluated;  
2:  $\alpha$ : algorithm step size with  $0 < \alpha \leq 1$ ;  
3: Initialization:  $V(s) :=$  an arbitrary initial value for all  $s$ ;  
4:  
5: for do each episode  
6:    $s \leftarrow$  initial state;  
7:   while  $s$  is not terminal do  
8:     for do each time step  $t$  in the episode  
9:        $a \leftarrow \pi(s)$ ;  
10:      Take action  $q$  and observe  $r, s'$ ;  
11:       $V(s) \leftarrow V(s) + \alpha[r + \gamma V(s') - V(s)]$ ;  
12:       $s \leftarrow s'$ ;  
13:     end for  
14:   end while  
15: end for
```

---



### 5.6.2 Q-learning

Q-Learning works just like TD Learning, but using  $Q(s, a)$  instead of  $V(s)$ . The policy is then computed from the final iterate using the usual formula. The colored lines are

---

**Algorithm 7** Q-learning for estimating  $\pi^*$ 

---

```
1: Input:  $\pi$ : policy to be optimized;
2:    $\alpha$ : algorithm step size with  $0 < \alpha \leq 1$ ;
3: Initialization:  $Q(s, a) :=$  an arbitrary initial value for all  $(s, a)$ ;
4: for do each episode
5:    $s \leftarrow$  arbitrary initial state;
6:   while  $s$  is not terminal do
7:     for do each time step  $t$  in the episode
8:       Choose action  $a$  according to the learning policy, e.g.  $\epsilon$ -greedy;
9:       Take action  $a$  and observe  $r, s'$ ;
10:       $Q(s, a) = Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$ ;
11:       $s \leftarrow s'$ ;
12:    end for
13:  end while
14: end for
```

---

for the comparison with the next algorithm.

### 5.6.3 SARSA

---

**Algorithm 8** SARSA for estimating  $\pi^*$ 

---

```
1: Input:  $\pi$ : policy to be optimized;
2:    $\alpha$ : algorithm step size with  $0 < \alpha \leq 1$ ;
3: Initialization:  $Q(s, a) :=$  an arbitrary initial value for all  $(s, a)$ ;
4: for do each episode
5:    $s \leftarrow$  arbitrary initial state;
6:   Choose  $a$  according to the learning policy, e.g.  $\epsilon$ -greedy;
7:   while  $s$  is not terminal do
8:     for do each time step  $t$  in the episode
9:       Take action  $a$  and observe  $r, s'$ ;
10:      Choose  $a'$  from  $s'$  according to the learning policy derived from  $Q$ , e.g.  $\epsilon$ -greedy;
11:       $Q(s, a) = Q(s, a) + \alpha[r + \gamma Q(s', a') - Q(s, a)]$ ;
12:       $s \leftarrow s'$ ;
13:       $a \leftarrow a'$ ;
14:    end for
15:  end while
16: end for
```

---

# Countable-state Markov Chains

A countable-state Markov chain contains an infinite but countable number of states, e.g.  $\mathbb{N}$ .

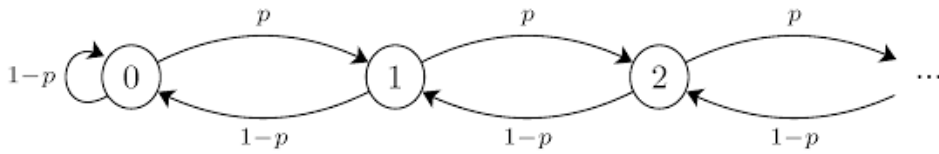


Figure 6.1: Example of countable-state Markov chain

In the example in figure 6.1, the first state is transient if  $p < 0.5$  and recurrent otherwise.

## 6.1 Definitions

- $f_{ij}(n)$  is the probability that the first passage of the Markov chain by state  $j$  happens at time  $n$ , given that  $X_0 = i$ . Mathematically,

$$f_{ij}(n) = P[X_n = j, X_k \neq j \quad \forall k < n | X_0 = i] \quad (6.1)$$

There exists a recurrence relation for  $f_{ij}(n)$ :  $f_{ij}(n) = \sum_{k \neq j} p_{ik} f_{kj}(n-1)$ , and  $f_{ij}(1) = p_{ij}$ .

- We denote  $F_{ij}(n)$  the probability that the first passage of the Markov chain by state  $j$  happens at time  $n$  or before, given that  $X_0 = i$ . Mathematically,

$$F_{ij}(n) = \sum_{m=1}^n f_{ij}(m) = P[X_k = j \quad \text{for some } k \leq n | X_0 = i] \quad (6.2)$$

- A state is recurrent if  $F_{jj}(\infty) = 1$ ;
- A state is transient if  $F_{jj}(\infty) < 1$ ;
- We define the random variable  $T_{ij}$  as the time until the first passage from  $i$  to  $j$ . Hence  $T_{jj}$  is the time needed to come back to state  $j$  after leaving it.
- The mean of  $T_{ij}$  is

$$\bar{T}_{ij} = 1 + \sum_{n=1}^{\infty} (1 - F_{ij}(n)) \quad (6.3)$$

▲ Theorem: all states from a same class are of the same type.

▲ Theorem: Let  $j$  be recurrent and aperiodic,  $i$  be a state in the same class, then

$$\lim_n P[X_n = j | X_0 = i] = \frac{1}{\bar{T}_{jj}} \quad (6.4)$$

• A Markov chain is said to be irreducible if all states communicate.

▲ Theorem: For an irreducible Markov chain, if the system

$$\begin{cases} \pi_j = \sum_i \pi_i p_{ij} & \forall j \\ \sum_j \pi_j = 1 \\ \pi_j \geq 0 & \forall j \end{cases} \quad (6.5)$$

has a solution, then  $\pi_i = 1/\bar{T}_{ii}$  and all states are positive-recurrent, i.e.  $f_{ii}(\infty) = 1$  and  $\bar{T}_{ii} < \infty$ . Moreover, if all states are positive-recurrent, then the system has a solution.

## 6.2 Birth-death Markov chains

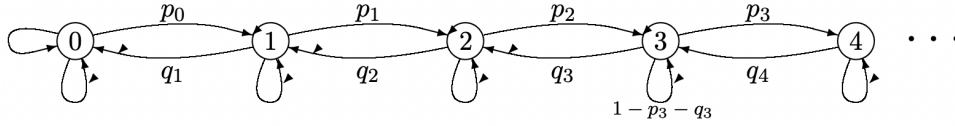


Figure 6.2: Example of birth-death Markov chain

This kind of Markov chain is used to model the evolution of a population or a queueing system. If the states are positive recurrent, we must have that

$$\pi_i p_i = \pi_{i+1} q_{i+1} \quad (6.6)$$

And so by recursion,

$$\begin{aligned} \pi_0 &= \frac{1}{1 + \sum_{i=1}^{\infty} \prod_{j=0}^{i-1} \rho_j} & \rho_j &= \frac{p_j}{q_{j+1}} \\ \pi_i &= \pi_0 \prod_{j=0}^{i-1} \rho_j \end{aligned} \quad (6.7)$$

→ Note: if  $\rho_i = \rho < 1$  for all  $i$ , then a solution exists and all states are positive recurrent.

### 6.2.1 Reversibility

If the Markov chain is in steady state, then the backward chain is homogeneous, i.e.

$$P[X_{n-1} = j | X_n = i] = P[X_n = i | X_{n-1} = j] \frac{\pi_j}{\pi_i} \quad (6.8)$$

and the backward transition probabilities  $p_{ij}^*$  are defined as

$$p_{ij}^* = p_{ji} \frac{\pi_j}{\pi_i} \iff \pi_i p_{ij}^* = \pi_j p_{ji} \quad (6.9)$$

A Markov chain is said to be reversible if  $p_{ij}^* = p_{ij}$ , or equivalently,  $\pi_i p_{ij} = \pi_j p_{ji}$ .

**Theorem 6.1.** Every birth-death chain with a steady-state probability distribution is reversible, due to the way the steady-state probabilities are computed (see equation (6.6)).

**Theorem 6.2.** Assume that an irreducible Markov chain has transition probabilities  $p_{ij}$ . Suppose  $\{\pi_i\}$  is a set of positive numbers summing to 1 and satisfying  $\pi_i p_{ij} = \pi_j p_{ji}$  for all  $i, j$ . Then,  $\{\pi_i\}$  is the steady-state distribution for the chain and the chain is reversible.

## 6.3 Processor sharing

The system considered here consists in  $m$  customers  $c_1, \dots, c_m$  where they are each served for an increment of time  $\delta$ . After  $c_m$  has been served, the server returns and start again. It is a cyclic serving order. When the service of  $c_i$  is completed, the client leaves and  $m$  is reduced, and when a client arrives,  $m$  increases.

→ Note: processor sharing is the limit of round-robin service as the increment  $\delta$  goes to 0.

### 6.3.1 Model

- Assume a Bernoulli arrival process in which the probability of an arrival in an interval  $\delta$  is  $\lambda\delta$ ;
- Assume the service time of customers to be IID arithmetic (i.e. multiple of  $\delta$ ) random variables  $Z_i$  with span  $\delta$ .

Let  $f(j) = P[Z_i = j\delta]$  and  $\bar{F}(j) = P[Z > j\delta]$ . Let  $g(j)$  be the probability that the customer that was served during  $j$  increments departs after one more increment of service:  $g(j) = \frac{f(j+1)}{\bar{F}(j)}$ .

Here is the dynamical system:

- A new customer arrives with probability  $\lambda\delta$  and goes to first position;
- The customer in first position is served for an increment  $\delta$ ;
- The customer that was served departs if service is complete;
- Otherwise, the customer goes to the back of the queue.

The states of the system are  $s = (m, z_1, \dots, z_m)$ , and we can show that the transition probabilities are given by

$$\begin{aligned}
 p_{s,r(s)} &= (1 - \lambda\delta)(1 - g(z_1)) \\
 p_{s,d(s)} &= (1 - \lambda\delta)g(z_1) \\
 p_{s,a(s)} &= \lambda\delta(1 - f(1)) \\
 p_{s,s} &= \lambda\delta f(1)
 \end{aligned} \tag{6.10}$$

where  $r(s) = (m, z_2, \dots, z_m, z_1 + 1)$  (=rotation),  $d(s) = (m - 1, z_2, \dots, z_m)$  (=departure),  $a(s) = (m + 1, 1, z_1, \dots, z_m, 1)$  (=arrival).

# Markov processes

## 7.1 Semi-Markov processes

### 7.1.1 Definitions

- A semi-Markov process is a Markov chain with holding times between transitions that are random variables that depend only from the starting and ending state of the transition. We denote  $S_1 < S_2 < \dots$  the times of the successive transitions, and  $X_0, X_1, \dots$  the successive states. Then,  $X(t) = X_n$  for all  $S_n \leq t < S_{n+1}$ .
- The embedded Markov chain is formed by the successive states of the semi-Markov process:  $P[X_n = j | X_{n-1} = i] = P_{ij}$ .
- $U_n := S_n - S_{n-1}$  is a random variable depending only on  $X_{n-1}$  and  $X_n$ :

$$P[U_n \leq u | X_{n-1} = i, X_n = j] =: G_{ij}(u) \quad \mathbb{E}[U_n | X_{n-1} = i, X_n = j] =: \bar{U}(i, j) \quad (7.1)$$

### 7.1.2 Stationary probabilities

**Theorem 7.1.** If the embedded Markov chain is irreducible and positive recurrent, then the stationary probability  $p_i$  of being in state  $i$  is given by

$$p_i = \frac{\pi_i \bar{U}(i)}{\sum_j \pi_j \bar{U}(j)} \quad (7.2)$$

where  $\bar{U}_i = \sum_j P_{ij} \bar{U}(i, j)$ .

## 7.2 Markov processes

### 7.2.1 Definitions

- A Markov process is a semi-Markov process where the holding times between transitions are exponentially distributed. Hence the holding time depends only on the current state:

$$P[U_n \leq x | X_{n-1} = i, X_n = j] = 1 - e^{-v_i x} \quad (7.3)$$

where  $v_i$  is the exponential parameter [transition/hour].

### 7.2.2 Transition rates

→ Note: the transition rates are not transition probabilities and can thus be bigger than 1.

Let  $Y(t)$  be the time until the next transition.

$$P[Y(t) \leq x, X(t + Y(t)) = j | X(t) = i, \{X(\tau); \tau < t\}] = p_{ij}e^{-v_i x} \quad (7.4)$$

where the term  $p_{ij}$  is the transition probability of the embedded Markov chain and the second term is the exponential distribution. Taking  $x$  infinitesimal, we find the transition rate between state  $i$  and  $j$ :

$$q_{ij} = v_i p_{ij} \quad v_i = \sum_j q_{ij} \quad (7.5)$$

### 7.2.3 Equivalences

As a summary of what has been seen until now, here is an equivalence of different stochastic processes:

- A semi-Markov process with exponentially distributed holding times.
- $N$  Poisson processes with rates  $v_i$ . If  $X(t) = i$ , the next transition will occur when an arrival occurs for the Poisson process  $i$  and the choice of the next state  $j$  is made with probability  $p_{ij}$ .
- $N^2$  Poisson processes with rates  $q_{ij}$ .

### 7.2.4 Stationary probabilities

The steady state probabilities  $\{\pi_i\}$  are found solving the system  $\pi_j = \sum_i \pi_i p_{ij}$ , and so we can find

$$p_i = \frac{\pi_i / v_i}{\sum_k \pi_k / v_k} \quad (7.6)$$

And if the sum at the denominator is finite,

$$\pi_i = \frac{p_i v_i}{\sum_k p_k v_k} \quad (7.7)$$

→ Note: A Markov process is irreducible iff the embedded Markov chain is irreducible.

### 7.2.5 Chapman-Kolmogorov differential equation (what for?)

Let  $P_{ij}(t) = P[X(t) = j | X(0) = i]$ . Then

$$\frac{dP_{ij}(t)}{dt} = \sum_{k \neq i} q_{ik} P_{kj}(t) - v_i P_{ij}(t) \quad (7.8)$$

### 7.2.6 Transient equation

Let us define the matrix  $Q$ :

$$Q_{ij} = q_{ij} \quad Q_{ii} = -\sum_{j \neq i} q_{ij} = -v_i \quad (7.9)$$

Then, Chapman-Kolmogorov differential equation can be written as

$$\frac{dP(t)}{dt} = QP(t) \quad (7.10)$$

And, with  $P(0) = I$ ,

$$P(t) = e^{Qt} = \sum_{n=0}^{\infty} \frac{Q^n}{n!} t^n \quad (7.11)$$

**Theorem 7.2.** Let  $Q$  be the transition rate matrix associated to an irreducible Markov process with  $n$  states. Then,  $Q$  has an eigenvalue equal to 0 with the corresponding right eigenvector being  $\mathbb{1}_n^T$  and left eigenvector  $p = (p_1, \dots, p_n) > 0$ . All other eigenvalues have strictly negative real part.

Defining  $\lambda_i$  as the eigenvalues of  $Q$ ,

$$P(t) = \sum_{i=0}^n v_i e^{\lambda_i t} \pi_i \quad (7.12)$$

where  $v_i$  and  $\pi_i$  are respectively the right and left eigenvector associated to  $\lambda_i$ .

## 7.3 Markovian queueing models

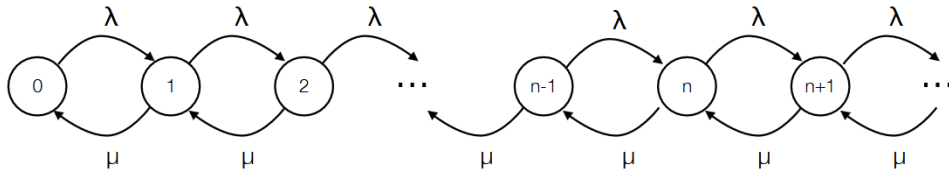


Figure 7.1: M/M/1 queueing model

### 7.3.1 M/M/1 queue

In a M/M/1 queue, the arrival process is a Poisson( $\lambda$ ) and the service time follows an Exponential( $\mu$ ). Therefore, the transition rate matrix is

$$Q = \begin{pmatrix} -\lambda & \lambda & 0 & 0 & \dots \\ \mu & -(\lambda + \mu) & \lambda & 0 & \dots \\ 0 & \mu & -(\lambda + \mu) & \lambda & \dots \\ 0 & 0 & \mu & -(\lambda + \mu) & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \quad (7.13)$$



We make the sensible assumption that  $\lambda < \mu$  (otherwise the queue would grow indefinitely). The steady-state probabilities are calculated as usual and give

$$p_0 = \frac{1}{\sum_{i=0}^{\infty} \left(\frac{\lambda}{\mu}\right)^i} = 1 - \frac{\lambda}{\mu} \quad p_i = \left(\frac{\lambda}{\mu}\right)^i \left(1 - \frac{\lambda}{\mu}\right) \quad \forall i > 0 \quad (7.14)$$

And the average number of jobs in the system is

$$L = \sum_{i=0}^{\infty} i p_i = \left(1 - \frac{\lambda}{\mu}\right) \sum_{i=0}^{\infty} n \left(\frac{\lambda}{\mu}\right)^n = \frac{\lambda/\mu}{1 - \lambda/\mu} = \frac{\lambda}{\mu - \lambda} \quad (7.15)$$

We can also calculate the following parameters:

- Average time spent in the system:  $W = L/\lambda = 1/(\mu - \lambda)$ ;
- Average time spent in the queue:  $W_q = W - 1/\mu = \frac{\lambda}{\mu(\mu - \lambda)}$ ;
- Average number of clients waiting in the queue:  $L_q = \lambda W_q = \frac{\lambda^2}{\mu(\mu - \lambda)}$ ;

Let us now calculate the probability to wait less than  $t$  minutes in the queue. By definition,

$$P[W < t | L = n] = \int_0^t \mu e^{-\mu\tau} \frac{(\mu\tau)^n}{n!} d\tau \quad (7.16)$$

And so

$$P[W < t] = \sum_{n=0}^{\infty} P[W < t | L = n] P[L = n] = 1 - e^{-(\mu - \lambda)t} \quad (7.17)$$

### 7.3.2 M/M/s queue

The M/M/s queue is a generalization of the M/M/1 queue to  $s$  servers instead of a single one.

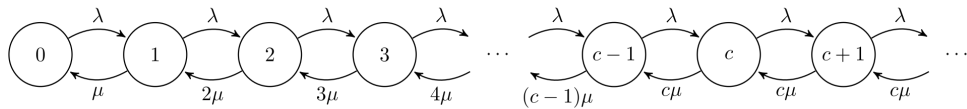


Figure 7.2: M/M/s queueing model

→ Note: the service time is not identical between every node: when there are less than  $s$  people in the system, not all servers are working.

→ Note: once again, we need an assumption for the convergence. Here, it is  $\lambda/s\mu < 1$ .

The steady state probabilities are

$$p_0 = \left( \left[ \sum_{n=0}^{s-1} \frac{\lambda^n}{n! \mu^n} \right] + \frac{\lambda^s}{s! \mu^s} \frac{1}{1 - \frac{\lambda}{s\mu}} \right)^{-1} \quad (7.18)$$

and

$$p_n = \frac{\lambda^n}{n! \mu^n} p_0 \quad \forall n \leq s \quad p_n = \frac{\lambda^s}{s! \mu^s} \left( \frac{\lambda}{s\mu} \right)^{n-s} p_0 \quad \forall n > s \quad (7.19)$$

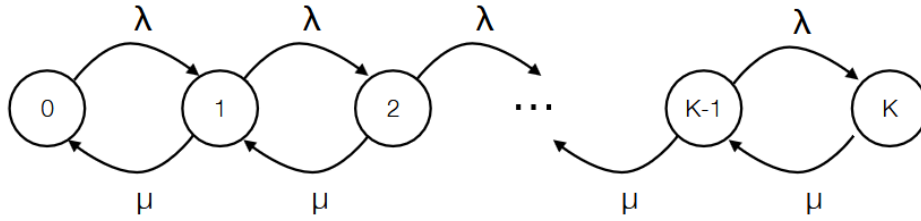
And once again, we have some parameters:

- Average number of clients waiting in the queue:  $L_q = p_0 \frac{\lambda^s}{s! \mu^s} \frac{\lambda/s\mu}{(1-\lambda/s\mu)^2}$ ;
- Average time spent in queue:  $W_q = L_q / \lambda$ ;
- Average time spent in system:  $W = W_q + 1/\mu$ ;
- Average number of clients in the system:  $L = \lambda W = L_q + \frac{\lambda}{\mu}$ ;

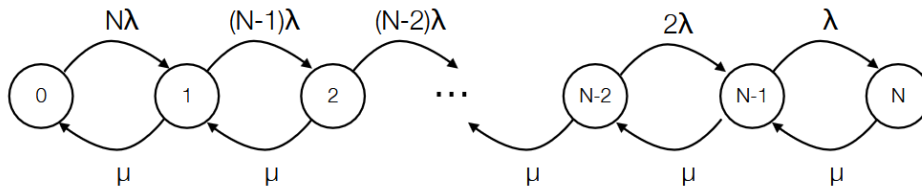
### 7.3.3 Other queues

There exists other types of queues used in Markov processes.

- M/M/1/K queue: In this queue, there is maximum number  $K$  of clients in the system and only one server.



- M/M/1//N: In this queue, there is a finite population of  $N$  people and only one server.



→ Note: Both queues can be combined to  $s$  servers.

## 7.4 Reversibility

We already saw that the backward transition probabilities of the embedded Markov chain are given by  $\pi_i p_{ij} = \pi_j p_{ji}^*$ . As the holding times follow an exponential distribution, the backward process does too. This means that

$$q_{ij}^* = v_i p_{ij}^* = \frac{v_i \pi_j q_{ji}}{\pi_i v_j} \implies \pi_i q_{ij} = \pi_j q_{ji}^* \quad (7.20)$$

And thus the process is reversible iff  $q_{ij}^* = q_{ij}$ .

We conclude that for a Markov process with stationary probabilities such that  $\sum_i p_i v_i < \infty$ , the embedded chain is reversible if the Markov process is reversible.

**Theorem 7.3.** For an irreducible Markov process, let the values  $\{p_i > 0\}$  be such that

$$\sum_i p_i = 1 \quad \sum_i p_i v_i < \infty \quad p_i q_{ij} = p_j q_{ji} \quad \forall i, j \quad (7.21)$$

Then the values  $\{p_i\}$  are the stationary probabilities and the process is reversible with positive recurrent embedded Markov chain.

Furthermore, if  $\{q_{ij}^* = 0\}$  such that

$$\sum_j q_{ij} = \sum_j q_{ji}^* \quad p_i q_{ij} = p_j q_{ji}^* \quad \forall i, j \quad (7.22)$$

Then,  $\{q_{ij}^*\}$  are the transition rates of the backward process.

**Theorem 7.4.** Birth and death processes (see section 6.2) are reversible if there exists  $\{p_i\} > 0$  such that

$$\sum_i p_i = 1 \quad \sum_i p_i v_i < \infty \quad p_i \lambda_i = p_{i+1} \mu_{i+1} \quad \forall i \quad (7.23)$$

**Theorem 7.5 (Burke Theorem).** • The departure process of clients from a M/M/1 queue is a Poisson process.

- The state of the system at time  $t$  is independent from departures before  $t$ .
- In case of FCFS (first come first served) service, the holding time of a customer leaving the system at time  $t$  is independent from departures before  $t$ .

## 7.5 More complex queues

### 7.5.1 Serial queues

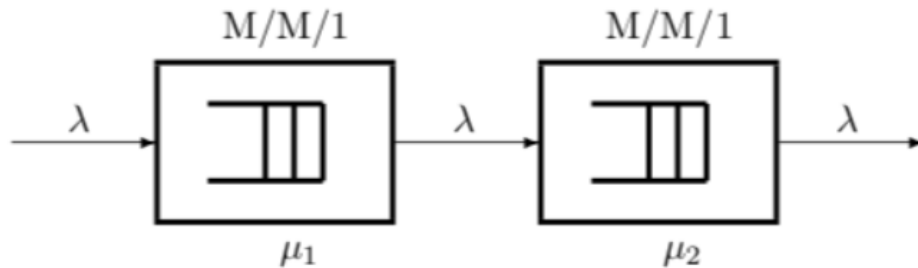


Figure 7.3: Serial queues

If the arrivals to the first queue form a Poisson Process and the service times are exponentially distributed for each server, the two systems behave as M/M/1 queues, and their states at any time  $t$  are independent, as are the holding times of each customer.

### 7.5.2 Feedback queues

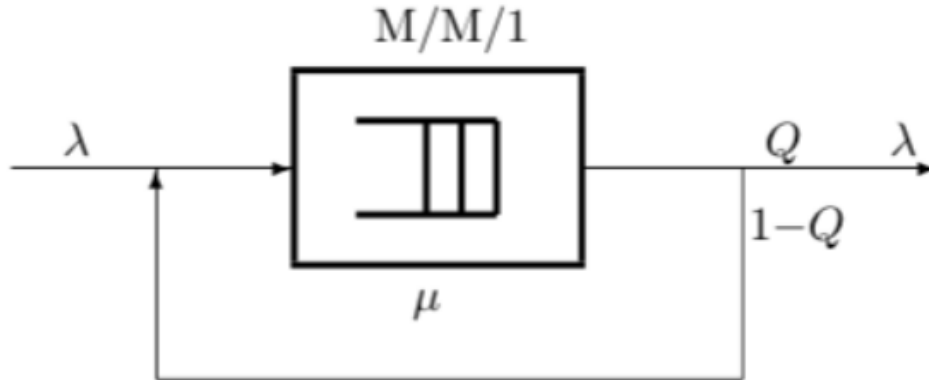


Figure 7.4: Feedback queues

The external arrivals form a Poisson process, but the overall arrivals to the queue do not. However, the global system does behave like a  $M/M/1$  queue as the return of a customer does not create a state change.

### 7.5.3 Network queues

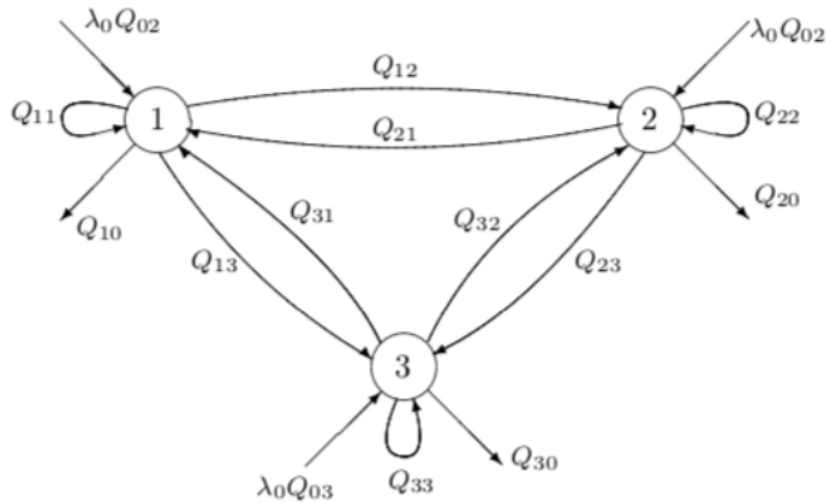


Figure 7.5: Network queues

We make the assumption that the external arrivals form a Poisson process, and that there are  $K$  stations with with a single queue and  $N_i$  servers. A satisfied customer is sent to another station or leaves the system with a probability that does not depend on what happened previously.

- The Markov process representing the system is reversible;
- The steady-state probabilities of having  $n$  customers at a station is identical to the distribution of an  $M/M/s$  queue;

- For all time  $t$ , the number of customers at each station is independent;
- The outgoing flows leaving the system are Poisson processes, but not the flows within the system.

## 7.6 Above and beyond

When the arrival and processing times are not Markovian, there exists no general analytical solution, but we have a good approximation through the Variability-Utilization-Time principle. For a single-server queue, this is equivalent to approximating the average number of customers in the queue by:

$$L_q \approx \left( \frac{c_a^2 + c_s^2}{2} \right) \frac{\rho/\mu}{1-\rho} \quad (7.24)$$

where  $c_a$  are respectively the coefficients of variation for the interarrival times  $\sigma_a\lambda$  and  $\sigma_s\mu$ , and  $\rho$  is the utilization rate of the server. **what does that mean?**

We can also approximate the coefficient of variation for the departure flow:

$$c_d^2 = c_s^2\rho^2 + c_a^2(1-\rho^2) \quad (7.25)$$

In the case of a  $s$ -server queue, the formulae become

$$L_q \approx \left( \frac{c_a^2 + c_s^2}{2} \right) \left( \frac{\rho\sqrt{2(s+1)-1}}{1-\rho} \right) \frac{1}{s\mu} \quad (7.26)$$

$$c_d^2 = 1 + \frac{c_s^2 - 1}{\sqrt{s}}\rho^2 + (c_a^2 - 1)(1 - \rho^2)$$

# Fluid approximation for non-stationary queueing systems

## 8.1 Transient behaviour

To reach its steady state, a queueing system needs to forget its initial situation. The relaxation time  $T_0$  is defined as the time it takes for the system to forget.

$$T_0 = \frac{\lambda/s\mu + C^2(s)}{(1 - \lambda/s\mu)^2} \frac{1}{s\mu} \quad (8.1)$$

If no process changes during a period of time  $T_0$ , then the system is said to be in steady state.

### 8.1.1 Non-stationary Poisson process

As seen in chapter 2, the counting process  $N(t)$  is a non-stationary Poisson process if it has independent increments, and if

$$\begin{aligned} P[N(t+dt) - N(t) = 0] &= 1 - \lambda(t)dt \\ P[N(t+dt) - N(t) = 1] &= \lambda(t)dt \\ P[N(t+dt) - N(t) \geq 2] &= 0 \end{aligned} \quad (8.2)$$

with  $\lambda(t)$  the arrival rate at time  $t$ , and thus the expected number of arrivals until time  $t$  is

$$\Lambda(t) = \int_0^t \lambda(\tau) d\tau \quad (8.3)$$

From that,

$$P[n \text{ arrivals between times } a \text{ and } b] = \frac{(\Lambda(b) - \Lambda(a))^n}{n!} e^{-\Lambda(b) - \Lambda(a)} \quad (8.4)$$

→ Note: the memoryless property is not satisfied anymore.

The distribution of the ordered arrivals is

$$P[\text{first arrival before time } t | N(T) = n] = 1 - \left( \frac{\Lambda(T) - \Lambda(t)}{\Lambda(T)} \right)^n \quad (8.5)$$

- We use a quasi-steady state approximation when the load stays below 1 and fluid approximation when it goes higher than 1.

## 8.2 Steady-state approximation