# Internship report

**Members**

Vo Van Nghia

**Date**

29 Sep, 2022

# Table of contents

# 1 Introduction

## 1.1 Institut de Recherche en Informatique de Toulouse (IRIT)

### 1.1.1 About the institut

The Institut de Recherche en Informatique de Toulouse (IRIT), created in 1990, is a Joint Research Unit (UMR 5505) of the Centre National de la Recherche Scientifique (CNRS), the Institut National Polytechnique de Toulouse (INP), the Université Paul Sabatier Toulouse3 (UT3), the Université Toulouse1 Capitole (UT1) and the Université de Toulouse Jean Jaurès (UT2J).

IRIT is one of the largest UMR at the national level, is one of the pillars of research in Occitanie with its 600 members, permanent and non-permanent, and about 100 external collaborators. Due to its multi-tutorial nature (CNRS, Toulouse Universities), its scientific impact and its interactions with other fields, the laboratory constitutes one of the structuring forces of the IT landscape and its applications in the digital world, both at regional and national level.

Through its cutting-edge work and dynamics, our unit has been able to define its identity and acquire undeniable visibility, while positioning itself at the heart of changes in local structures: University of Toulouse, as well as the various mechanisms resulting from future investments (LabEx CIMI, IRT Saint-Exupéry, SAT TTT, 3IA ANITI).

IRIT has focused its research on five major scientific issues and six strategic application areas.

- Health, Autonomy, Living, Well-being
- Smart City
- Aerospace and Transportation
- Social Media, Digital Social Ecosystems
- e-Education for learning and teaching
- Heritage and People Safety

As well as strategic action:

- Scientific Computing, Big Data and AI

### 1.1.2 Organization

The 24 research groups of the laboratory are dispatched in seven scientific departments:

- Dpt ASR : Architecture, Systems, Networks
- Dpt CISO : HPC, Simulation, Optimization
- Dpt FSL : Reliability of Systems and Software
- Dpt GD : Data Management
- Dpt ICI : Interaction, Collective Intelligence
- Dpt IA : Artificial Intelligence
- Dpt SI : Signals, Images

## 1.2 The internship

Markov decisions processes (MDPs) and their model free counterpart in reinforcement learning (RL) have known a large success in the last two decades. Although research in these two areas has been taking place for more than fifty years, the field gained momentum only recently following the advent of powerful hardware and algorithms with which suprahuman performance were obtained in games like Chess or Go.

However, these impressive successes often rely on quite exceptional hardware possibilities and cannot be applied in many ”usual” contexts, where, for instance, the volume of data available or the amount of computing power is more restricted. To define the next generation of more ”democratic” and widely applicable algorithms, such methods still need to deal with very demanding exploration issues as soon as the state/action spaces are not small. One way around this is to use underlying knowledge and structure present in many MDPs. This is especially true for problems related to scheduling and resources sharing in among others server farms, clouds, and cellular wireless networks. The internships will revolve around this theme of improving the efficiency of learning algorithms by leveraging the structure of the underlying problem and focus mainly on modelfree approach.

# 2 System settings

Two type of systems are studied in this internship: queuing system and load-balancing system.

## 2.1 Queuing system

### 2.1.1 Parameters

We have $n$ classes of queue $C_1, \ldots, C_n$. For each class of queue, its behavior is fully controlled by which environment it is in. An environment can be in one of the states $m_1, \ldots, m_m$. And the environment of class $C_i$ is a random variable, denoted by $M_i$, which is in one of the states $m_1, \ldots, m_m$. For each environment, $C_i$ has their own holding cost $c_{i,j}$ (the cost of one unfinished unit of work on the queue), arrival rate $\lambda_{i,j}$ (the rate of one more unit of work arriving to the queue) and departure rate $\mu_{i,j}$ (the rate of one unit of work on the queue has finished). Furthermore, the environment $M_i$ can change from $m_j$ and $m_k$ with the rate of $\xi_{i,j,k}$. In this internship, we focus on the case where $n := 2$. A table summarizing all parameters is shown below.

Table 1: Queuing system parameters

| | | $m_1$ | ... | $m_m$ |
|---|---|---|---|---|
| $C_1$ | Holding cost | $c_{1,1}$ | ... | $c_{1,m}$ |
| | Arrival rate | $\lambda_{1,1}$ | ... | $\lambda_{1,m}$ |
| | Departure rate | $\mu_{1,1}$ | ... | $\mu_{1,m}$ |
| $\vdots$ | | | | |
| $C_n$ | Holding cost | $c_{n,1}$ | ... | $c_{n,m}$ |
| | Arrival rate | $\lambda_{n,1}$ | ... | $\lambda_{n,m}$ |
| | Departure rate | $\mu_{n,1}$ | ... | $\mu_{n,m}$ |

For each $C_i$, we have a matrix transition as below.

Table 2: Matrix transition of the environment of $C_i$

| | $m_1$ | ... | $m_m$ |
|---|---|---|---|
| $m_1$ | $\xi_{i,1,1}$ | ... | $\xi_{i,1,m}$ |
| $m_m$ | $\xi_{i,m,1}$ | ... | $\xi_{i,m,m}$ |

The state of the system is represented by two vectors:

- $S = (X_1, \ldots, X_n)$ where $X_i$ is a random variable represents the current number of works of class $C_i$ and is observable.
- $E = (M_1, \ldots, M_n)$ and this vector is not observable.

### 2.1.2 Cost

The cost of the system is a function of $S$ and $E$. We propose two functions of cost. The first one is a simple linear function.

$$f_1(S, E) = \sum_{1}^{n} c_{i,M_i} X_i$$

And the second one is a convex function which is specialized for the case $n = 2$, where $\epsilon$ is a fixed positive constant.

$$f_2(S, E) = c_{1,M_1} X_1 + c_{2,M_2}(\epsilon X_2^2 + X_2)$$

### 2.1.3 Agent

The agent will decide which queues should be activated. His mission is to minimize the cost of the whole system. Only works on activating queues can be processed and finished. In more generic problems, the agent is allowed to activate / deactivate multiple queues at the same time based on some conditions. However, in this internship, the agent can only activate one queue at a time.

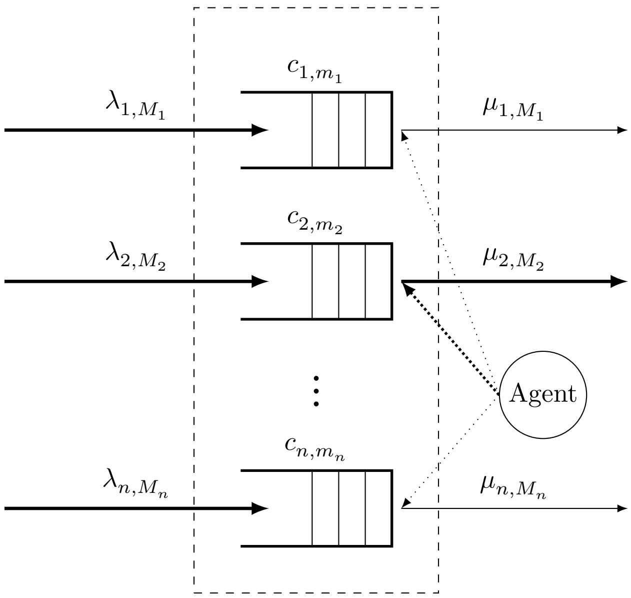


Figure 1: Visualization of the queueing system when the agent activates queue 2, the bold lines represent the flow of works inside the system.

### 2.1.4 Evolution

In the continuos-time scale, each type of event (work arrival, work departure and environment changing) happens independently. That system is quite hard to program, therefore, we used a technique called "uniformization" to move from the continuos-time scale to discrete-time scale. In this time scale, at a given time, only one event can take place, regardless of their type.

In particular, given the system state is $(S, E)$, the agent decides to activate queue $a$. One of these $n + 1 + n(m - 1) + 1$ events can happen.

- The work of class $C_i$ increases by 1. Because there are $n$ classes, we have $n$ events of this kind with the rate $\lambda_{1,M_1}, \ldots, \lambda_{n,M_n}$ respectively (if the number of work of class $C_i$ reaches an upper limit, we consider $\lambda_{i,M_i} = 0$).

- The work of class $C_a$ decreases by 1. Because we can only activate only one class, there is only one event of this type and its rate is $\mu_{a,M_a}$.
- The environment of class $C_i$ changes to a different environment other than $m_i$ and the rate changing to environment $j$ is $\xi_{i,M_i,j}$. Because there are $m-1$ possible changes for each class and there are $n$ classes, the number of this kind of event is $n(m-1)$.
- And a special dummy event where nothing changes.

A discrete probability distribution is used to express that. In order to satisfy the condition of a probability distribution, all the rates above are divided by a normalization constant $C$ to make sure that their sum are not greater than 1. If that sum is smaller than 1, the special dummy event is used to fill the gap so that the final sum will be equal to 1.

The normalization constant has the form as follow, which is deduced from the above evolution of the system.

$$C = \sum_{i=1}^{n} \max_{j} \lambda_{i,j} + \max_{i,j} \mu_{i,j} + \sum_{i=1}^{n} \max_{j} \sum_{k=1,k\neq j}^{m} \xi_{i,j,k} + \epsilon$$

If $\epsilon > 0$, the probability of the dummy event will always be greater than 0. In this internship, as we do not want the system evolves too slowly, we choose $\epsilon := 0$.

After obtaining all the information above, we use that discrete probability distribution to obtain the next transition $T$ of the system and denote $S'$ and $E'$ the next state of the system.