Performance evaluation of green IT networks

Youssef Ait El Mahjoub

université PARIS-SACLAY

Performance evaluation of green IT networks

Évaluation des performances pour des réseaux IT économes en énergie

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 580, Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Réseaux, information et communications
Unités de recherche: (1) Université Paris-Saclay, UVSQ, Données et Algorithmes pour une ville intelligente et durable, 78035, Versailles, France.
(2) Institut Polytechnique de Paris, Télécom SudParis, Services Répartis, Architectures, Modélisation, Validation, Administration des Réseaux, 91000, Evry, France.
Référent: Université de Versailles-Saint-Quentin-en-Yvelines

Thèse présentée et soutenue à Paris-Saclay, le 18 mars 2021, par

Youssef AIT EL MAHJOUB

Composition du jury:

Lynda Mokdad **Professeur des universités, Université Paris-Est**	**Présidente**
Olivier Brun **Directeur de recherche, LAAS-CNRS**	**Rapporteur & examinateur**
Gérardo Rubino **Directeur de recherche, INRIA/IRISA**	**Rapporteur & examinateur**
Thomas Begin **Maître de conférences HDR, Université Lyon1**	**Examinateur**
Anne-Cécile Orgerie **Chargée de recherche et HDR, IRISA**	**Examinatrice**
Véronique Vèque **Professeur des universités, Université Paris-Sud**	**Examinatrice**

Direction de la thèse

Jean-Michel Fourneau **Professeur, DAVID - UVSQ**	**Directeur**
Hind Castel-Taleb **Professeur, SAMOVAR - Télécom SudParis**	**Co-directrice**
Emmanuel Hyon **Maître de Conférences, Université Paris Nanterre**	**Invité**

Thèse de doctorat

NNT: 2021UPASG011

Acknowledgments

It will be very difficult for me to thank everyone because it is thanks to the help of many people that i was able to complete this thesis.

First of all, i would like to greatly thank my thesis director and co-director, Jean-Michel Fourneau and Hind Castel-Taleb, for their help. I am delighted to have worked with them because in addition to their scientific support, they were always there to support and advise me during the development of this thesis.

I also thank the Labex DigiCosme which believed in my research project.

Olivier Brun and Gérardo Rubino honored me by being the rapporteurs of my thesis, they took the time to listen to me and to discuss with me. Their remarks allowed me to consider my work from another angle. For all that i thank them.

I would like to thank the members of the jury for having accepted to participate in my thesis committee and for their scientific participation as well as the time they devoted to my research.

I thank the members of the DAVID laboratory for their kindness and our discussions around a coffee in the hallways, especially the "ALMOST" team in which i developed my research during my thesis and my internships. A special thought to Stefi Nouleho, Mael guiraud, Chen Wei, Thierry Mautor, Franck Quessette, Sandrine Vial, Yann Strozeski, Pierre Coucheney, Dominique Barth ...

I thank from the bottom of my heart for their support and their wisdom my parents, my uncle Brahim, my grandfather Mohamed. I thank my brothers Ayoub, Reda, and Anouar, my cousins. My family in Morocco and in France.

Related international conference papers

- Youssef Ait El Mahjoub, Jean-Michel Fourneau and Hind-Castel Taleb, "Energy Packet Networks with general service time distribution". A conference paper. In 28th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), pp. 1-8. Nice, France, November 2020, DOI: 10.1109/MASCOTS50786.2020.9285965.

- Youssef Ait El Mahjoub, Jean-Michel Fourneau and Hind Castel-Taleb. "Analysis of Energy Consumption in Cloud Center with Tasks Migrations". A conference paper. In: CN2019, International Conference on Computer Networks, pp 301−315. Kamień Śląski, Poland, July 2019. DOI: 10.1007/978 − 3 − 030 − 21952 − 9_23.

- Youssef Ait El Mahjoub, Jean-Michel Fourneau and Hind Castel-Taleb. "A numerical approach of the analysis of optical container filling". A conference paper. In: VALUETOOLS 2019, the 12th EAI International Conference on Performance Evaluation Methodologies and Tools, pp. 159-162. Palma, Spain, March 2019. DOI: *doi.org*/10.1145/3306309.3306333.

- Youssef Ait El Mahjoub, Hind Castel-Taleb and Jean-Michel Fourneau. "Performance and energy efficiency analysis in NGREEN optical network". A conference paper. In: WIMOB 2018, 14th International Conference on Wireless and Mobile Computing, Networking and Communications, pp. 1−9. Limassol, Cyprus, October 2018. DOI: 10.1109/W*i*MOB.2018.8589144.

- Jean-Michel Fourneau, Youssef Ait El Mahjoub, Franck Quessette and Dimitris Vekris. "XBorne 2016: A Brief Introduction". A Conference paper. In: ISCIS 2016, International Symposium on Computer and Information Sciences, vol. 659, pp. 134−141. Kraków, Poland, October 2016. DOI: 10.1007/978 − 3 − 319 − 47217 − 1_15.

Related journal paper

- Jean-Michel Fourneau and Youssef Ait El Mahjoub. "Processor sharing G-queues with inert customers and catastrophes: A model for server aging and rejuvenation". A journal paper. In: Probability in the Engineering and Informational Sciences, vol. 31, no. 04, pp. 420−435, Cambridge, April 2017. DOI: 10.1017/S0269964817000092.

Contents

List of Figures

List of Tables

List of Algorithms

List of abbreviations

ACPI Advanced Configuration and Power Interface

AMD Advanced Micro Devices

BBU Base-Band Unit

BFS Breadth-First Search

BGS Block Gauss Seidel

CRAN Cloud Radio Access Network

CTMC Continuous-Time Markov Chain

DP Data Packet

DPM Dynamic Power Management

DTMC Discrete-Time Markov Chain

DVFS Dynamic Voltage and Frequency Scaling

EP Energy Packet

EPN Energy Packet Network

ES Energy Storage

FCFS First Come First Serve

GBE Global Balance Equation

GPU Graphics Processing Unit

GTH Grassmann, Taksar and Heymann

IAAS Infrastructure as a Service

ICT Information and communications technology

IoT Internet of Things

IP Internet Protocol

IT Information Technology

KMS Koury, McAllister and Stewart

LB Load Balancing

LORA Long Range

LPWAN Low Power Wide Area Network

MDP Markov decision process

MTU Maximum Transmission Unit

NCD Near Completely Decomposable

OBS Optical Burst Switching

OCS Optical Circuit Switching

OPM Optimized Power Management

OPS Optical Packet Switching

OSS Optical Slot Switching

OTDM Optical Time Division Multiplexing

PDU Protocol Data Unit

PEPA Performance Evaluation Process Algebra

PIT Propagation of Instantaneous Transitions

PS Processor Sharing

RCAT Reversed Compound Agent Theorem

RRH Remote Radio Head

SBBP Switched Bernouilli Batch Process

SDU Service Data Unit

SOR Successive Over-Relaxation

SPM Static Power Management

TCP Transmission Control Protocol

VM Virtual Machine

VMM Virtual Machine Monitor

WDM Wavelength Division Multiplexing

WSN Wireless Sensor Network

Part I

Introduction

A. The context and problematics

The IT sector has a very high contribution on the increase of the overall energy consumption. Hence a CO_2 emissions increase. Many methods to reduce consumption in other industries or services results in more IT and telecommunications (the "Green by IT" approach [CIG17]) and therefore in an increase of consumption in IT domains. Green by IT concept aims to reduce the economic, ecological and social footprint of the company's activities (a product or a service) through digital technologies. Several themes can be explored: smart grids, mobility and smart transportation, environmental and urban monitoring, dematerialization, remote work and video-conferencing, intelligent buildings and eco-design software.

In the processing and the networking domain, energy optimization is mainly based on an adaptation of the architecture and the resources employed according to the traffic flows to be transported (or processed) and the promised quality of service QoS. We therefore seek to adapt resources to demand, which results in a dynamic dimensioning adapted to the load. This is by nature different from the worst-case dimensioning commonly used. In terms of technology, this requires network equipment to have "sleep", "deep sleep", or "hibernate" modes (the terminology varies among suppliers), but all of these modes are associated with the same concept: putting the equipment in sleep mode to reduce its energy consumption. The decision for switching modes is not trivial, for instance putting down or in sleep mode a device for a very short period of time could be not efficient due to the restarting or awakening power for an eventual use. For a relevant performance/energy tradeoff, it is important to use energy consumption formulas obtained from the network resource utilization and devices.

The methods we used in this document are based on the queueing network theory, Markov chain analysis and stochastic comparison theory. We first determine the queueing system (or Markovian process) to analyze, and then we investigate the analytical solution for the steady-state distribution (if it exists). A semi-closed solution is suggested in Chapter 3 and product-form solutions are proposed in Chapter 6, 8, and 9. According to the nature of the system, we also perform a numerical resolution using: the GTH algorithm, one of the algorithms implemented in the XBorne tool in Chapter 2), is a very precise direct method that does not benefit from the chain structure, the Power (Block-Power), the Gauss-Seidel (Block-Gauss-Seidel) which are classical iterative algorithms. Block-resolution versions are efficient if the Markov chain exhibits a block structure. The resolution of multiple blocks and their coupling works efficiently and faster than the resolution of a large block. In that way, we have proposed a new resolution algorithm for the "Near Completely Decomposable (NCD)" Markov chains. This algorithm is derived from Robertazzi's algorithm which assumes a specific behaviour of the Markov chain (more details in Chapter 4). After obtaining the steady-state distribution probability of the system, we can derive various performance measures we call 'rewards'. The main advantage of the product form solution of a network of queues is that the rewards can be obtained separately for each queue, which facilitate the calculations. Also we can conduct an optimization of the rewards based on a cost function that combines several measures. We were particularly interested in efficient analytical solutions and

fast numerical algorithms in order to conduct an optimization of the rewards for large scale systems. Some of the rewards we have derived are: the mean number of jobs, the mean response time, the emptiness probability of the system, the lost rate of jobs (for example, if the queue is finite), the probability of servers switching state (for example, from "IDLE" to "AWAKE" state), the mean power (and IDLE power) consumption of the system, the energy per job consumption. The mean power and energy consumption are derived from an energetic equation that should consider all the states of the system. This equation varies from one model to another, and depends on the components and features of the system.

In the application level, we have addressed several issues:

- At processing level;
  - The Dynamic Voltage and Frequency Scaling (DVFS) in a processor's cores, The processor adapts its core's speed to the current workload. In order to optimize the resource utilization.
  - The migration of tasks between physical servers in a cloud center. The Over-loaded servers share a part of their workload with less-loaded servers in order to minimize overall energy consumption and performance.
- At networking level;
  - The dimensionning of an optical network by studying the resident time of packets which is the sum of the gathering time of packets in optical containers, the insertion time of containers in the optical network, and the transport time of packets in the network.
  - The assignment of energy packets in a sensor network (as LoRa network), where each sensor gets energy packets from photo-voltaic solar panels and the energy flow is stored in sensor's battery.

B. The organization of the document

The document is composed of 5 parts (see Fig. 1). In each part, with the exception of the introduction and perspectives, we present the state of the art of the incoming works. In next, we give a brief review of chapters.

In the first part, a general introduction is made which covers all the chapters. In the second part, we have brought together all the studies that have involved the numerical analysis of Markov chains, in particular, a numerical resolution for steady-state distribution. Whether using a classical resolution algorithm, or by proposing a new resolution algorithm:

- Chapter 1 (state of the art): We point out some power management techniques, and optical network technologies.
- Chapter 2 : We present the last version of XBorne a software tool for the probabilistic modeling with Markov chains. The numerical analysis of Markov

chains always deals with a trade-off between complexity and accuracy. Therefore we need tools to compare the approaches, the codes and some well-defined examples to use as a test-bed

- Chapter 3 : We proposed a comparative numerical study for a DVFS processor. In particular, for the AMD Opteron processor. Using a global cost function, we derive the performance and power consumption per task for different processor configurations. In that way, we show how to determine the best configuration for a given set of input parameters.

- Chapter 4 : Here, we collaborated on the NGREEN project that aims to design and validate a versatile network architecture with a scalable capacity, low cost and low energy consumption. In this work, we studied two parts of the network. The first one, is the mechanism used to fill the optical container with the electronic packets (i.e. Internet Protocol (IP) or Ethernet) and the second one is the insertion node where the flows of optical containers are queued before being emitted on the ring.

In part III, we present an analytical analysis for the energy consumption in a cloud/data center

- Chapter 5 (state of the art) : In this chapter we recall some load balancing of tasks strategies. Also, we have listed many power equations related to physical servers consumption.

- Chapter 6 : In this work, we study how to optimally minimize the power consumption using an exact analysis of the queueing network with customers migration. We use the multi-server Jackson network to represent the behavior of the cloud center.

In part IV, we conduct an analytical study on Energy Packet Networks (EPNs) and G-Networks (also called Gelenbe-Networks).

- Chapter 7 (state of the art) : We discuss the previous models of the Energy Packet Networks (EPNs) and G-networks and their resolution techniques.

- Chapter 8 : In this work, we give the proof of the product form of the steady-state distribution of an EPN model we propose. We focus on performance evaluation and energy losses rate. We also show how to optimize a sensor network with a solar panels harvesting capacity. This illustrates one of the main advantages of EPNs models. Based on its closed form solution, it is possible to conduct an optimization of the systems utilities (rewards).

- Chapter 9 : Here, we present a proof of the product form of the steady-state distribution of a G-network model. The proposed model is about the aging and rejuvenation of servers. The aging of servers is triggered by a internal or external catastrophe signal.

Finally in Part V, we examine the perspectives to further improve of our works.

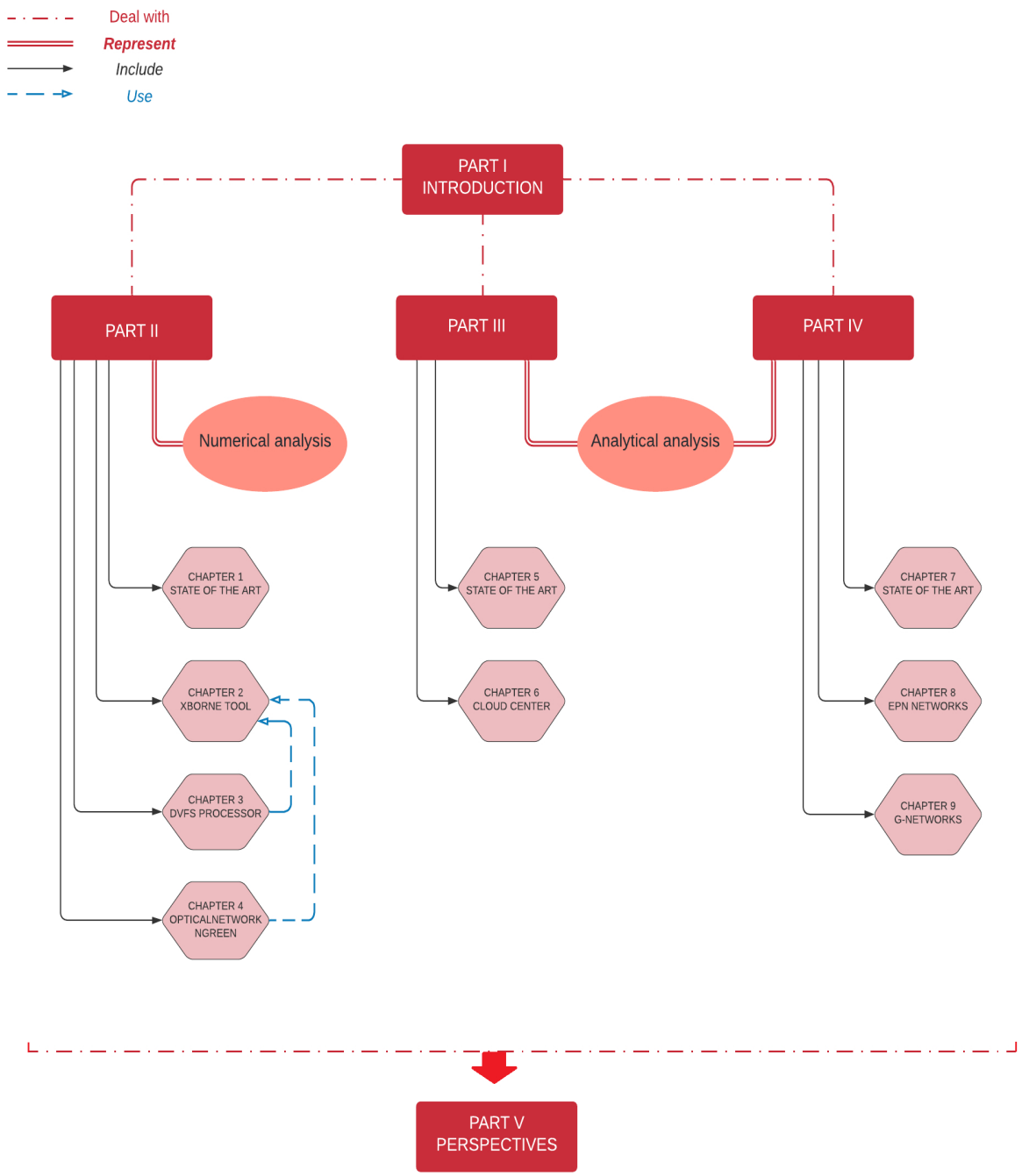


Figure 1: Chapters overview

Part II

Numerical analysis of energy consumption and performance in networking and processing systems

Chapter 1

State of the art of power management and optical networks

Contents

1.1 Power and energy consumption strategies

Table 1.1 presents multiple surveys on energy and power consumption in different areas of Information and communications technology (ICT): computing, storage, data management, network, infrastructure. These surveys are mainly derived from [DWF16], we have added new ones especially for data centres.

Technologies, methods or approaches which are used to minimize the power/energy consumption of ICT equipment are known as Power Management Techniques [Bel+11]. In the following we will present several methods, that are widely cited in the literature ([Bel+11; OAL14; Zak18]):

1.1.1 SPM, DPM and DVFS

The energy consumption of a device (servers, disks, routers ...) is the power consumption needed for the device to operate for a period of time. So in order to minimize the energy, either the device needs less power to operate, or simply switch it off when it is not in use. However, a switched off device is unavailable to perform any task and might take considerable time to become available. Therefore, hardware designers implement other capabilities to devices such as Dynamic Voltage and Frequency Scaling (DVFS) so that energy can be minimized if the device is not in use [OAL14]. Static Power Management (SPM) are techniques where system's behavior does not change. SPM makes the hardware suitable for Dynamic Power Management (DPM) if the hardware has a certain capability such as DVFS, then DPM techniques make it possible to use that hardware capability. DPM includes methods for run-time adaptation of the system behaviour according to resource demand. DPM relates to application level resource management techniques, which is considered more energy efficient than SPM both in single server and large systems [Bel+11].

1.1.2 Virtualization

Virtualization means to create a virtual version of a device or resource, such as a server, storage device, network or even an operating system where the framework divides the resource into one or more execution environments. In terms of cloud computing and data-centers, virtualisation is considered as the most promising approach to save energy, which increases resource utilization. For different types of workload scheduled on a virtualized and physical (non-virtualized) servers, the study in [LA12] suggests that a virtualized server (running two VMs) can save up to 51.7% more energy as compared to two physical servers (non-virtualized) treating the same type of workload.

1.1.3 Scheduling – VM allocation/placement

A virtualized host can accommodate several vms and it is possible that a number of hosts could run the VM with variations in energy use due to resource heterogeneity – it may take more, or less, energy to run the same VM's work on different hosts. A cluster scheduler is responsible to allocate hosts for VMs inside a cluster or

Year	Contributors	Area of focus
2005	Venkatachalam et al. [VF05]	Power consumption of microprocessor systems
2011	Beloglazov et al. [Bel+11]	Energy-efficient design of data centers and cloud computing systems
2011	Wang et al. [Wan+11]	Energy-saving techniques for data management
2012	Reda et al. [RN12]	Power modeling and characterization for processors
2012	Sekhar et al. [SJD12]	Servers consolidation with Virtual Machine (VM) live migration
2013	Ge et al. [GSW13]	Energy efficiency of data centers and content delivery networks
2013	Bostoen et al. [BMB13]	Power reduction techniques for data-center storage systems
2014	Orgerie et al. [OAL14]	Energy efficiency of computing and network resources
2014	Mittal [SMi14]	Energy efficiency in embedded computing systems
2014	Mittal et al. [SV14]	GPU energy efficiency
2014	Hammadi et al. [HM14a], Bilal et al. [al14; BKZ13]	Data center networks and their energy efficiency
2014	Ebrahimi et al. [EJF14]	Data center cooling technology
2014	Rahman et al. [RLK14], Mittal [Mit14]	Data center power management
2014	Gu et al. [GHJ14]	VM power metering
2014	Kong et al. [KL14]	Renewable energy usage and/or carbon emission in data centers
2015	Mastelic et al. [Mas+15]	Energy efficiency in ICT technology
2016	Shuja et al. [al16]	Data center energy efficiency
2018	Zakarya [Zak18]	Different approaches to make data-centers greener

Table 1.1: Comparison of related surveys .

data-center [Ver+15]. Therefore, to reduce the energy consumption and guarantee the desired level of performance, it is essential to allocate energy and performance efficient resources. There are many of scheduling and VM placement (optimal,

heuristics and approximate) algorithms available in the literature [Bel+11; Ver+15; TNR11; VAN08; BKB07].

1.1.4 Consolidation

Virtualization allows gathering several virtual machines into a single physical server using a technique called VM consolidation. VM consolidation can provide significant benefits to cloud computing by facilitating better use of the available data center resources [Abd+14]. It can be performed either statically or dynamically. In static VM consolidation, the Virtual Machine Monitor (VMM) allocates the physical resources to the VMs based on peak load demand. This leads to resource wastage because the workloads are not always at peak. In case of dynamic VM consolidation, the VMM adapts the VM capacities according to the current workload demands (resizing). This helps in utilizing the data centers resources efficiently. In [Rei+12] the authors suggest that in Google's cluster, hosts are not highly utilized, and some power might be saved through consolidation. Studies [Rei+15; Cou14] also suggest that approximately **30%** of the running servers in US datacenters are idle and the others are under-utilized, making it possible to save energy and money by using VM consolidation to reduce the number of hosts in use.

1.1.5 Thresholds strategies

Stochastic methods and queuing theory together provide a valuable approach to answer important questions about data centre systems, in particular performance and power/energy consumption. In [Mit11] Mitrani proposed a dynamic operating policy where a subset of the available servers is designated as 'reserve'. The state of the reserves is controlled by two thresholds: they are powered up when the number of jobs in the system becomes sufficiently high, and are powered down when that number becomes sufficiently low. For analytic study, Mitrani uses generating functions to solve system's balance equations. Then he expressed losses (R) and server's energy consumption (S) in a single function $C = c_1 R + c_2 S$. Using heuristics this function is minimized and optimal values of thresholds are derived. In chapter 2 we studied two variations of this model using XBorne tool (a software tool for the probabilistic modeling with Markov chains). Authors In [MD15] used only one threshold. It could be seen as an particular case of Mitrani's model when the two thresholds are equal. But the queue has only one server and many server's state are considered (LOW, SETUP, BUSY, IDLE, OFF). Other well-known threshold strategy is the hysteresis queueing system [LG99; Sho+15; Tou+]. This k servers model uses $k-1$ thresholds to power on $(F_1, F_2, \ldots, F_{k-1})$ or power off $(R_1, R_2, \ldots, R_{k-1})$ the servers. For instance, when a customer arrives to an empty system, it is served by a single server. Whenever the number of customers exceeds a forward threshold F_i, a server is added to the system. Whenever the number of customer falls below a reverse threshold R_i, a server is removed from the system. Using stochastic complement many variation of this model (as (1) homogeneous servers with Poisson arrivals, (2) homogeneous servers with bulk (Poisson) arrivals and (3) heterogeneous servers with Poisson arrivals) is studied in [LG99]. Also in [Tou+] authors propose

effective optimization methods (heuristics and Markov chain aggregation) for the search for threshold values that minimize an overall cost function that considers performances and resource use. These thresholds strategies could be very efficient if the powering on/off of servers is instantaneous (which is not often verified in server's data center).

Many communication networks and computer systems use load balancing to improve performance and resource utilization. The ability to efficiently divide service requests among system resources can have a significant effect on performance and energy consumption. Static [SM91; Sou+12] load balancing mainly based on the information about the average of the system work load. It does not take the actual current system status into account. In [AD85] static load balancing algorithms aim at finding optimal customer routing to optimize the throughput and other performance indices under certain constraints. Dynamic or adaptive load balancing policies are the most efficient ones. The system reacts dynamically to the network state and traffic is directed to routes with less load or extra load, depending on a cost function. Dynamic load balancing algorithms are usually classified into two families: receiver-initiated and sender-initiated [DEJ86]. In the first case, an overloaded node decides to send some of its job to another node that receives them and tries to process or reallocate them elsewhere. In the latter case, it is the receiver that decides when it can poll another node to import some of its jobs. In [DEJ86] also in [RDJ90] for heterogeneous systems, authors compare sender- and receiver-initiated strategies and conclude that, under heavy load, receiver-initiated algorithms give lower expected response time than sender-initiated one. Also In [ASJ17], authors address the problem of dynamic load balancing for networks with open topology where an arbitrary number of nodes implement a receiver-initiated dynamic load balancing algorithm. The algorithm's point is to compute the polling rates among the network stations that ensure both the network to be in product-form and that the sets of specified stations have their load balanced.

Regarding the literature on product-form analysis (see e.g. [SJH13; SA13; GM15; Gar+16]) for some works in the field, Load Balancing (LB) networks present important particularities. Compared to the literature on signals in G−networks and similar models ([XMM99; Gel93a; Gel93b; Gel93d; FV95; AB12]), LB networks have the property that the network population is preserved by node interactions. In Chapter 6, we present a study based on the load balancing between physical servers, in order to reduce energy consumption of the system.

1.2 Optical networks

In order to respond to the continued growth in data traffic, new technologies and network structures must be implemented. For example, with the evolution of communication materials, electrical cables have been replaced by optical fibres, which has made it possible to increase the bandwidth of communication channels from Kbit/s to Tbit/, thanks to new multiplexing technologies such as Wavelength Division Multiplexing (WDM). In optical networks The two leading multiplexing technologies are: Optical Time Division Multiplexing (OTDM) and WDM .

1.2.1 OTDM and WDM Multiplexing technologies

In OTDM, lower bit rate optical streams are assigned to different time slots on the multiplexed channel [TEK88]. In other words, it's practical to combine a set of low-bit-rate streams, each with a fixed and pre-defined bit rate, into a single high-speed bit stream that can be transmitted over a single channel. In contrast to WDM, OTDM only uses one wavelength. WDM technology is very similar to can allow multiple non-overlapping wavelength channels to transmit in the same optical fibre link. Each of these channels can operate at a different data rate. Essentially, the bandwidth capacity of the optical fibre is multiplied by the number of wavelengths multiplexed onto it. Each wavelength being an independent channel can transmit data at a different rate [Don12]. This parallelism mechanism increases the bandwidth, consequently reduces the optical fibre cable cost and use of equipment.

1.2.2 Optical Switching methods

An optical switch is a multi-port network bridge which connects multiple optic fibers to each other and controls data packets routing between inputs and outputs. Some optical switches convert light to electrical data before forwarding it and converting it into a light signal again. The main objectives of optical switching are (a) increasing bandwidth (b) reducing protocol issues and density of the network. For that purpose, in below we present briefly the three main optical switching techniques:

- Optical Circuit Switching (OCS), was the first optical switching technique used in optical networks. In OCS the network is configured to establish a circuit, from an entry to an exit node, by adjusting the optical cross connect circuits in the core routers in a manner that the data signal, in an optical form, can travel in an all-optical manner from the entry to the exit node. Ideally, the packets that enter the network should be transported from the ingress to the egress point in an all-optical form. The technology needed to process the headers of the packets using only optics is not yet available, and thus, the packets need to be converted to electrical form so that current electronic integrated circuits can interpret the header and make the convenient routing decisions [11b]. This approach suffers from all the disadvantages of circuit switching i.e. the circuits require time to set up and to destroy, and while the circuit is established, the resources will not be efficiently used to the unpredictable nature of network traffic.

- Optical Packet Switching (OPS), in OPS the payload is switched optically. OPS can be faster, and also cheaper to purchase and maintain than traditional switching [SZC09]. OPS hardware could lower power requirements, dissipate less heat and take less space compared to electronic equipment [31]. We can expect OPS to eventually replace traditional electronic switching, because optical network equipment is cheaper to maintain, more reliable and consume less energy [Ram06] then their electronic counterparts.

- Optical Burst Switching (OBS) is used in core networks, and viewed as a feasible compromise between the existing Optical Circuit Switching OCS and

the yet not viable Optical Packet Switching OPS [11a]. In OBS, the packets are aggregated in the entry node, for a very short period of time. This allows that packets that have the same constraints (the same destination, the same quality of service requirements...) are sent together as a burst of data. When the burst arrives at the exit node, it is disassembled and gathered packets are routed to their destination.

Chapter 2

XBorne the software tool

Contents

2.1 Introduction

We present the last version (2016) of XBorne a software tool for the probabilistic modeling with Markov chains. The tool which has been developed initially as a test-bed for the algorithmic stochastic comparisons of stochastic matrices and Markov chains, is now a general purpose framework which can be used for the Markovian modelling in education and research.

The numerical analysis of Markov chains always deals with a trade-off between complexity and accuracy. Therefore we need tools to compare the approaches, the codes and some well-defined examples to use as a test-bed. After many years of development of exact or bounding algorithms for stochastic matrices, we have gathered the most efficient into XBorne, our numerical analysis tool [Fou+03]. Typically using XBorne, one can easily build models with tens of millions of states. Note that solving any questions with this size of models is a challenging issue. XBorne was developed with the following key ideas:

1. Build one software tool dedicated to only one function and let the tools communicate with file sharing
2. If another tool already exists for free and is sufficiently efficient, use it and write the export tool (only create tools you cannot find easily).
3. Allow to recompile the code to include new models.
4. Separate the data and the description of the data.

As a consequence, we have chosen to avoid the creation of a new modelling language. The models are written in C and included as a set of 4 functions to be compiled by the model generator. This aspect of the tool will be emphasized in section 2 with the presentation of an example (a queue with hysteresis). The tool decomposition approach will also be illustrated in the study.

XBorne is now a part of the French project MARMOTE which aims to build a set of tools for the analysis of Markovian models. It is based on PSI3 to perform perfect simulation (i.e. Coupling from the past) of monotone systems and their generalizations [Bus+10], MarmoteCore to provide an object interface to Markov objects and associated methods, and XBorne that we will present in this study. The aim of XBorne (and the other tools developed in the MARMOTE project) is not to replace older modeling tools but to be included into a larger framework where we can share tools and models developed in well-specified frameworks which can be translated into one another. XBorne will be freely available upon request.

The technical part of this work is as follows: in section 2.2, we present how we can build a new model. We show in section 2.3 how it can be solved and we present some numerical results. In section 2.4, we consider the quasi-lumpability technique. We modify the Tarjan and Paige approach used for the detection of macro-states for aggregation or bi-simulation [VF10] to relax the assumption on the creation of macro states and accommodate a quasi-lumpable partition of the state space. In section 2.5 we show how to build, solve and extract some rewards in a Birth-death process . Section 2.6 is devoted to the study of performance and power consumption in a threshold queuing system using XBorne.

2.2 Building a model with XBorne

XBorne can be used to generate a sparse matrix representation of a Discrete Time Markov Chain (DTMC) from a high level description provided in C. Continuous-time models can be considered after uniformization (see the example in the following). Like many other tools, the formalism used by XBorne is based on the description of the states and the transitions. All the information concerning the states and the transitions are provided by the modeler using 2 files (1 for the constants and one for the code, respectively denoted as "const.h" and "fun.c"). States belong to a hyper-rectangle the dimension of which is given by the constant NEt. The bounds of the hyper-rectangle must be given by function "InitEtendue()". The states belong to the hyper-rectangle and they are found by a Breadth-First Search (BFS) visit from an initial state given by the modeler through function "EtatInitial()".

The transitions are given in a similar manner. The constant "NbEvtsPossibles" is the number of events which provoke a transition. The idea is that an event is a mapping applied to a state (not necessarily a one to one mapping). Each event has a probability given by function "Probabilite()" and its value may depend on the state description. The mapping realized by an event is described by function "Equation()". To conclude, it is sufficient to describe 4 functions in C and some definitions and recompile the model generator to obtain a new code which builds the transition probability matrix.

```
#define NEt   2              #define NbEvtsPossibles   4
#define AlwaysOn   10        #define BufferSize  20
#define OnAndOff    5        #define UPandDOWN  0
#define WARMING     1        #define ALL_UP  2
#define UP         10        #define DOWN  5
```

We now present an example for the various definitions and functions which are written in the files "const.h" and "fun.c" to describe the model developed by Mitrani in [Mit11] to study the tradeoff between power consumption and quality of service in a data-center. It is a model of a M/M/(a+b) queue with hysteresis and impatience. We have slightly changed the assumptions as follows: the queue is finite with size "BufferSize". The arrivals still follow a Poisson process with rate "Lambda". The services are exponential with rate "Mu". Initially only "AlwaysOn" servers are available. Once the number of customers in the queue is larger than "UP", another set (with size OnAndOff) of servers is switched on. The switching time has an exponential duration with rate "Nu". If the number of customers becomes smaller than "DOWN", this set of servers is switched off. This action is immediate. As NEt=2, a state is a two dimension vector. The first dimension is the number of customers and the second dimension encodes the state of the servers. The initial state is an empty queue with the extra block of servers which is not activated.

```
void InitEtendue()
{
    Min[0] = 0;  Max[0] = BufferSize; Min[1] = UPandDOWN;  Max[1] = ALL_UP;
}
void EtatInitial(E)
```

```
int *E;
{
    E[0] = 0; E[1] = UPandDOWN;
}
double Probabilite(int indexevt, int *E) {
   double p1, Delta;
    int nbServer, inserv;
    nbServer = AlwaysOn;
    if (E[1]==ALL_UP) {nbServer += OnAndOff;}
    inserv = min(E[0], nbServer);
    Delta = Lambda + Nu + Mu*(AlwaysOn + OnAndOff);
    switch (indexevt) {
         case ARRIVAL: p1 = Lambda/Delta; break;
         case SERVICE: p1 = (inserv)*Mu/Delta; break;
         case SWITCHINGON: p1 = Nu/Delta; break;
         case LOOP: p1 = Mu*(AlwaysOn + OnAndOff - inserv)/Delta; break;
    }
    return(p1);
}
```

The model is in continuous time. Thus we build an uniformized version of the model adding a new event to generate the loops in the transition graph which are created during the uniformization. After this process we have 4 events: ARRIVAL, SERVICE, SWITCHINGON, LOOP. In all the functions, E and F are states. The generation tool creates 3 files: one contains the transition matrix in sparse row format, the second gives information on the number of states and transitions and the third one stores the encoding of the states. Indeed the states are found during the BFS visit of the graph and they are ordered by this visit algorithm. Thus, we have to store in a file the mapping between the state number given by the algorithm and the state description needed by the modeler and some algorithms.

```
void Equation(int *E, int indexevt, int *F, int *R)
{
    F[0] = E[0]; F[1] = E[1];
    switch (indexevt) {
      case ARRIVAL: if (E[0]<BufferSize) {F[0]++;}
              if ((E[0]>=UP) && (E[1]==UPandDOWN)) {F[1]=WARMING;}
              break;
      case SERVICE: if (E[0]>0) {F[0]--;}
              if ((F[0]==DOWN) && (E[1]>UpandDOWN)) {F[1]=UPandDOWN;}
              break;
      case SWITCHINGON: if (E[1]==WARMING) {F[1]=ALL_UP;}
              break;
      case LOOP: break;
    }
}
```

Once the steady-state distribution is obtained with some numerical algorithms, the marginal distributions and some rewards are computed using the description

of the states obtained by the generation method and comes codes provided (and compiled) by the modeler to specify the rewards (see in the left part of Fig. 2.1 the marginal distribution for the queue size).

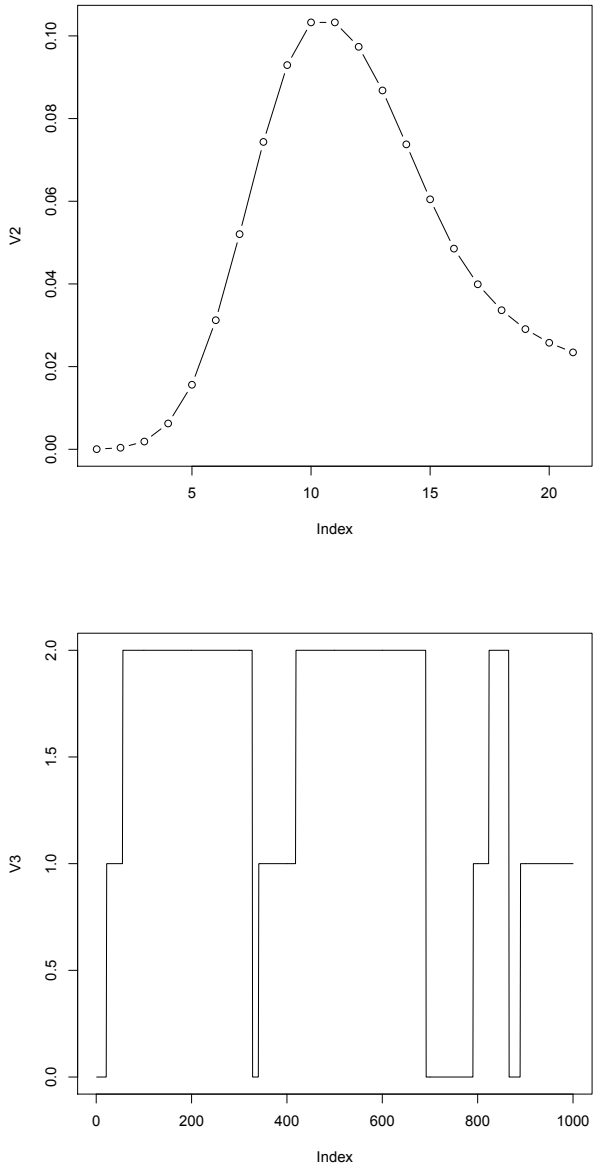


Figure 2.1: Mitrani's model. Steady-state for the queue size (the first figure in above). Sample path of the state of the servers (the second figure in above).

2.3 Numerical Resolution

In XBorne, we have developed some well-known numerical algorithms to compute the steady-state distribution (Grassmann, Taksar and Heymann (GTH) for small

matrices), Over-Relaxation (SOR) and Gauss Seidel for large sparse matrices but we have chosen to export the matrices into MatrixMarket format to use state of the art solvers which are now available on the web. But we also provide new algorithms for the stochastic bounds or the element-wise bound of the matrices, the stochastic bound or the entry-wise bounds of the steady-state distribution. These bounds are based on the algorithmic stochastic comparison of Discrete Time Markov Chain (see [FP02] for a survey) where stochastic comparison relations are mitigated with structural constraints on the bounding chains. More precisely, the following methods are available:

- Lumpability: to enforce the bounding matrix to be ordinary lumpable. Thus, we can aggregate the chain [FLQ04].
- Pattern based: to enforce the bounding matrix to follow a pattern which provides an ad-hoc numerical algorithm (think at a upper Hessenberg matrix for instance) [BF05].
- Censored Markov chain: only the useful part of the chain is censored and we provide bounds based on this partial representation of the chain [DPY06; BDF12].

Other techniques for entry-wise bounds of the steady state distribution have also been derived and implemented [BF11]. They allow in some particular cases to deal with infinite state space (otherwise not considered in XBorne).

More recently, we have developed a new low rank decomposition for a stochastic matrix [BFB14]. This decomposition is adapted to stochastic matrices because it provides an approximation which is still a stochastic matrix while singular value decomposition gives a low rank matrix which is not stochastic anymore. Our low rank decomposition allows to compute the steady-state distribution and the transient distribution with a lower complexity which takes into account the matrix rank. For instance, for a matrix of rank k and size N, the computation of the steady-state distribution requires $O(Nk^2)$ operations. We also have derived algorithms to provide stochastic bounds with a given rank for any stochastic matrix (see [BFB14]).

Note that the integration with other tools we mention previously is not limited to numerical algorithms provided by statistical package like R. We also use their graphic capabilities and the layout algorithms. We illustrate these two aspects in Fig. 2.2. In the left part we have drawn the layout of the Markov chain associated with Mitrani's model for a small buffer size (i.e. 20). We have developed a tool which reads the Markov chains description and write it as a labelled directed graph in "tgf" format. With this graph description, we use the graph editors available on the web to obtain a layout of the chain and to visualize the states and their transitions. On the right part of the figure, we have depicted a heat diagram for the mean power consumption associated to Mitrani's model for all the values of the thresholds U and D.

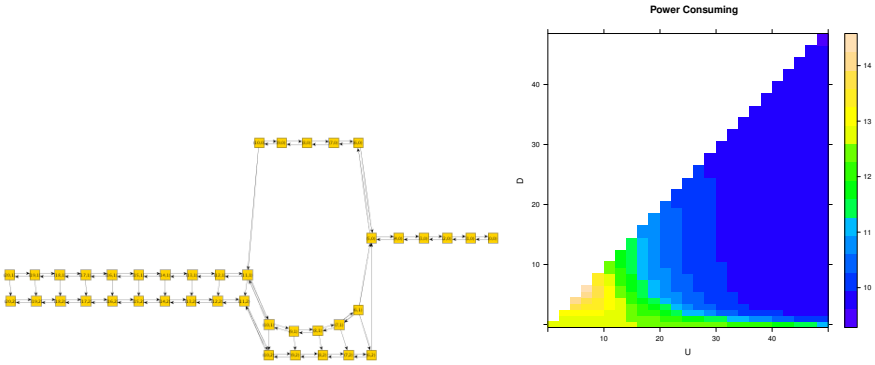


Figure 2.2: Mitrani's model. Directed graph of the chain (left). Mean power consumption (right).

2.4 Quasi-Lumpability

Quasi-Lumpability testing has been recently added into XBorne to analyze very large matrices. The numerical algorithms which have been developed are also used to analyze stochastic matrices which are not completely specified. It is well-known now that Tarjan's algorithm can be used to obtain the coarsest partition of the state space of a Markov chain which is ordinary lumpable and which is consistent with an initial partition provided by the modeler. Lumpable matrix can be aggregated to obtain a smaller matrix, easier to analyze. Logarithmic reduction in size are often reported in the literature. We define quasi-lumpability of partition $A_1, A_2, \ldots, A_k$ with threshold ϵ of stochastic matrix M as follows: for all macro-states A_i and A_j we have

$$\max_{l1,l2 \in A_i} | \sum_{k \in A_j} M(l1,k) - \sum_{k \in A_j} M(l2,k)| = E(i,j) \leq \epsilon. \tag{2.1}$$

When $\epsilon = 0$ we obtain the definition of ordinary lumpability. We have modified Tarjan's algorithm to obtain a partition which is quasi-lumpable given an initial partition and a maximum threshold ϵ. The output of the algorithm is the coarsest partition consistent with the initial partition and the real threshold needed in the algorithm (which can be smaller than ϵ). Note that the algorithm always returns a partition. However the partition may be useless as it may have a large number of nodes. The next step is to lump matrix M according to the partition found by the modified Tarjan's algorithm. If the real threshold needed is equal to 0, the matrix is lumpable and the aggregated matrix is stochastic. It is solved with classical methods.

If the threshold needed is positive, we obtain two aggregated matrices Up and Lo: one where the transition probability between macro states A_i and A_j is equal to $\max_{l \in A_i} \sum_{k \in A_j} M(l,k)$ and one where it is equal to $\min_{l \in A_i} \sum_{k \in A_j} M(l,k)$. Up is super-stochastic while Lo is sub-stochastic. These two bounding matrices also appear when the Markov chains are not completely specified and transitions are associated with intervals of probability. We have implemented Courtois and Semal algorithm [CS85] to obtain entry-wise bounds on the steady-state distribution of all matrices between Up and Lo. We are still conducting new research to improve this algorithm.

2.5 Birth-death process

Birth-death processes have been used in many applications including ecology, population genetics, epidemiology, and queuing theory. Here we are interested in birth-death processes that generate the same state transition rate after a given state (such as a M/M/C queue). For example, to study the average response time of a data processing server, birth transitions will represent the arrival of tasks and death transitions represent the processing of tasks. For this type of system, the resolution of the steady-state is half numerical (from state 0 to the given state) and half analytical (for all states after the given state). The steady-state algorithm we use (C code) requires three description files and generates two output files. The input files are

- "Model.Rank" this file must contain two integers R_1 and R_2 separated by a blank space or a line break. R_1 represents the state at which all arrivals rate transitions occur at the same rate, R_1 must be a positive integer ($R_1 \geq 0$). Also, R_2 is the triggering rate of the service transitions, R_2 must be a non-zero positive integer ($R_2 > 0$).

- "Model.Lambda" file must contain $R_1 + 1$ numbers that represents arrival rate at each state in $\{0, \ldots, R_1\}$.

- "Model.Mu" must include service rate at each state in $\{1, \ldots, R_2\}$.

The generated files are "Model.pi" that contains the steady-state distribution, and "Model.rewards" that will contain the mean number of tasks in the server and the mean response time.
For example, the three files representing a M/M/4 queue:

"MM4.R*ank*"	"MM4.L*ambda*"	"MM4.M*u*"
0	5	2
4		4
		6
		8

The sequence of service rates in file "MM4.Mu" reports that in state 1 the service rate is 2 tasks per unit of time, in state 2 the service rate is 4, in state 3 the service rate is 6, and for all state $x \geq 4$ the service rate is 8. "MM4.Lambda" claims that arrival rate of tasks is the same (5 tasks per unit of time) for all the states. The results obtained from this will automatically be stored in "MM4.pi" and "MM4.rewards". Also note that "Model.pi" and "Model.rewards" will not be generated if the birth-death system considered is not stable (i.e. the last transition in "MM4.Lambda" file is strictly lower than the last transition in "MM4.Mu" file). In the next chapter, we study some variations of the M/M/C queue in order to investigate the performance, and the power and energy consumption of the so-called "Opteron processor". The Opteron processor uses six configurations for it's cores speed. We have performed the C code of this section to compare the different configurations (more details in Chapter 3).

2.6 A numerical example in XBorne

To study performance and power consumption, we looked at Mitrani's model [Mit11] (as briefly presented in the last section). Mitrani offers a multi-server threshold model with an analytical form that combines performance and mean power consumption in a single function. This model is based on thresholds, i. e. if the number of customers exceeds the threshold "UP" then reserve servers will be switched on and conversely after a "DOWN" threshold the reserve servers will be switched off. Note that the servers switching-on time is not instantaneous but follows an exponential distribution. We were interested in studying the power consumption and then the performance of the system (separately) according to different server's activating time distributions. More specifically, does the variation of activation time distribution (while keeping the same average rate) influence the performance/power consumption of the system.

2.6.1 Presentation of Mitrani's model

Mitrani's model (Fig. 2.3) is presented as follows: Arrivals are "Poissonian", switching on servers and service times are "Exponential", the queue is infinite, the servers are homogeneous, a server can be either "OFF" or on "WARMING" or "ON" (resp. "UPandDOWN", "WARMING" and "ALL_UP" in the "C" code in section 2.2) , the queue contains "N" servers of which "n" are always running and N-n "reserve servers" which turn on and off according to the threshold policy. The reserve servers are set to "UPandDOWN" if the number of customers is below a threshold "DOWN" and vice versa, they are set to "WARMING" if the number of customers is above a threshold "UP ". The servers stop instantly and can be shut down even if they are in the WARMING state, a customer cannot wait indefinitely, at the end of a deadline the customer leaves the queue without being served (abandonment of a customer), the deadline of a customer follows the exponential distribution. The Continuous-Time Markov Chain (CTMC) that describes this model is (X, Y) where X is the number of customers with X $\in [0, +\infty]$ and Y is the state of the reserve servers with Y $\in$ {0 (UPandDOWN), 1 (WARMING), 2 (ALL_UP) }.

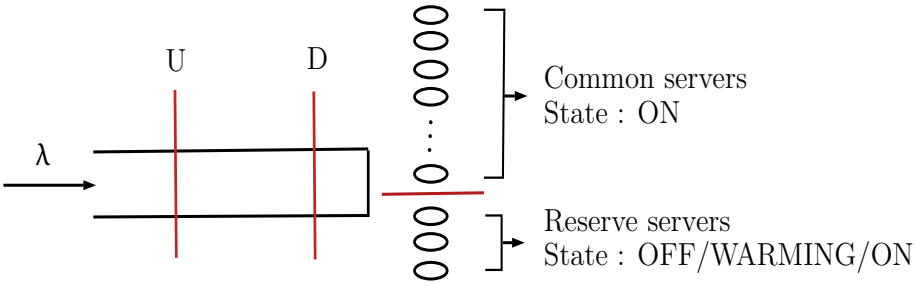


Figure 2.3: Mitrani's queuing system

2.6.2 Variation of server's activating time distribution

In this model, we consider that the queue is finite in order to make Mitrani's model more realistic. Thus we are no longer talking about customer abandonment but rather customer losses that are caused by the arrival of a customer when the queue is full. The other changes concern the distribution of servers activating time. We studied three activation times distributions with **the same average rate** and different **coefficient of variation**. The three distributions are: the "Exponential" with a coefficient of variation $r = 1$, the " $Erlang(k)$ " with $r = \frac{1}{k} < 0$ when $k > 1$, and the " Hyper-Exponential " with a coefficient $r > 0$. For this new model, we keep the same Markov chain (X,Y) but the description of X and Y has changed. X the number of customers is finite (X $\in$ [0, BufferSize]) and Y depends on the number of transitions before activating the reserve block. In "Erlang(k)" case we have $Y \in [0, k]$.

2.6.3 Variation of customers inter-arrival distribution

In this model, we are investigating the variation in customers inter-arrival time distribution. We studied the Poisson distribution and the Poisson distribution under a 2-states modulating chain (i.e. Switched Bernouilli Batch Process (SBBP)), while keeping the same average. the modulating chain has 2 states, i.e. if the chain is in state 0 (resp. state 1) then we expect an arrival of customers of Poisson type with a low intensity rate (resp. or high intensity rate). this model requires additional information which is the phase or state of the modulating chain. The Markov chain describing the system is (X,Y,Z) where X is the number of customers in the queue (X $\in$ [0, BufferSize]), Y is the state of the reserve servers which depends on the servers activation distribution and Z is the state of the modulating chain (Z $\in$ {0, 1 }).

2.6.4 Numerical results

In all the experiments below, we consider 10 servers always UP and 5 reserve servers and a queue capacity of 50 customers. For each server, the service rate is one customer per time unit and the activating rate is 0.2. The arrivals rate is 9 customers per time unit. An active switched on server consumes one Watt, an idle server consumes 0.6 Watts, the switching on of a server consumes 1.5 Watts, and the switching off does not consumes power. Note that, we are studying different distributions for the arrivals of customers and the activating time of servers, but the mean rates remains the same.

Below (Figures 2.4, 2.5 and 2.6) we present some numerical results on the mean power consumed and the loss rate of customers for each type of activation distribution. We keep the same arrival distribution, Poisson, for all three figures. The x-axis represents the UP threshold and y-axis the Down threshold. It should be remembered that the UP and DOWN thresholds play a crucial role in performance. It is these two parameters that regulate the activation/deactivation of the reserve servers whatever the variation of the Mitrani model studied. It is by turning OFF or ON the servers that the system consumes less or more. The power

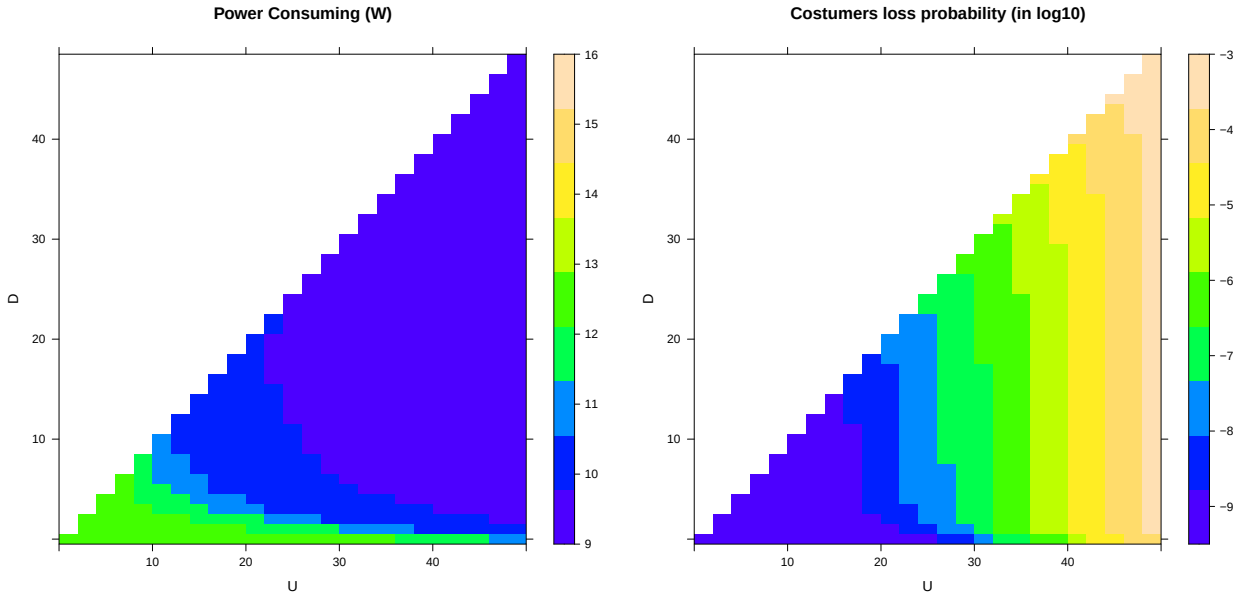
consumption equation we considered takes into account the power consumed by an active powered server P_{ON}, idle server $P_{IDL} = P_{ON} * 0.6$ and the power required for the activation $P_{WARMING} = P_{ON} + \frac{P_{ON}}{2}$ and $P_{OFF} = 0$. The point is to minimize the losses as well as the power consumption of the system, it is clear that there is a compromise and that we should find the right equilibrium between these two performance indicators. The "UP" and "DOWN" thresholds must comply with the $BufferSize > UP \geq DOWN > 0$ condition. The graphs below illustrate the power consumption/performance trade-off. It can be seen that when the power consumed is low (the purple area in Fig. 2.4-a) the average probability of customers loss increases (the green to yellow and light brown areas in Fig. 2.4-b). So the equilibrium area is the blue area in both figures in Fig. 2.4. Therefore, for an ecological and efficient use of the system, in the case of Erlang distribution, the value of the UP and DOWN thresholds should be chosen at the intersection of the intervals delimited by the two blue zones so around $DOWN = 10$ and $UP = 20$. The blank field in figures is the case where $DOWN > UP$, which is not allowed since the activation threshold must always be greater than (or equal to) the deactivation threshold.

Concerning the variation of the ignition distribution (Erlang(3) Fig. 2.4, Exponential Fig. 2.5, and HyperExponential(2) Fig. 2.6) it has been observed that Erlang's distribution provides better results in both power and performance in contrast to HyperExponential. This is particularly important in cases of thresholds that trigger server behavior change. For example, if one takes a high enough DOWN threshold, the servers will be practically all the time off and therefore the activation process that one wants to study is not present enough. Therefore, the more the activation time distribution is varying (high coefficient of variation), the less efficient the system is, and so the more steady the time distribution is, the better the system behaves.

In order to compare the power consumption for each activation time distribution, one should observe figures 2.4-a, 2.5-a and 2.6-a. Also comparing system losses means comparing 2.4-b, 2.5-b and 2.6-b. The losses are expressed in log_{10} since we are comparing very small probabilities, and the power consumption in units of Watts.

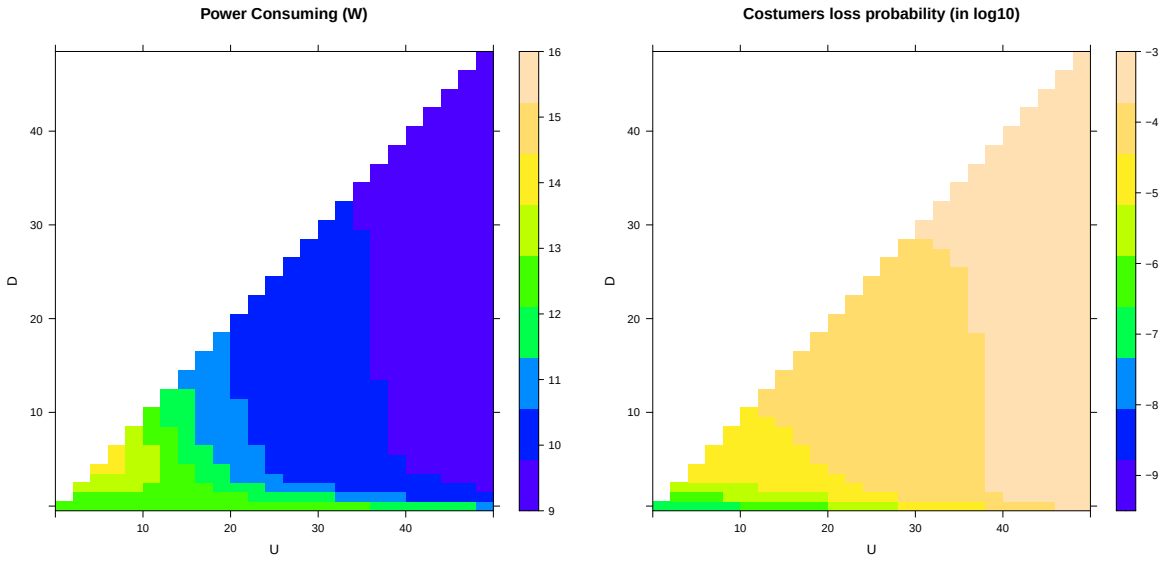
In Fig. 2.7 we represents the results for an SBBP arriving time distribution with an Erlang(3) distribution for the ignition of servers. We observe the same behavior for the trade-off between power consumption and losses rate. Also, comparing this figure with Fig. 2.4 which considers Poissonian arrivals and also Erlang activating servers. We conclude that the Poissonian configuration for the arrivals behaves better than the SBBP one for both rewards.

We deduce from these numerical results that the system performs better with less power consumption when considering low coefficient of variation for the distribution of inter-arrivals and servers activation time.



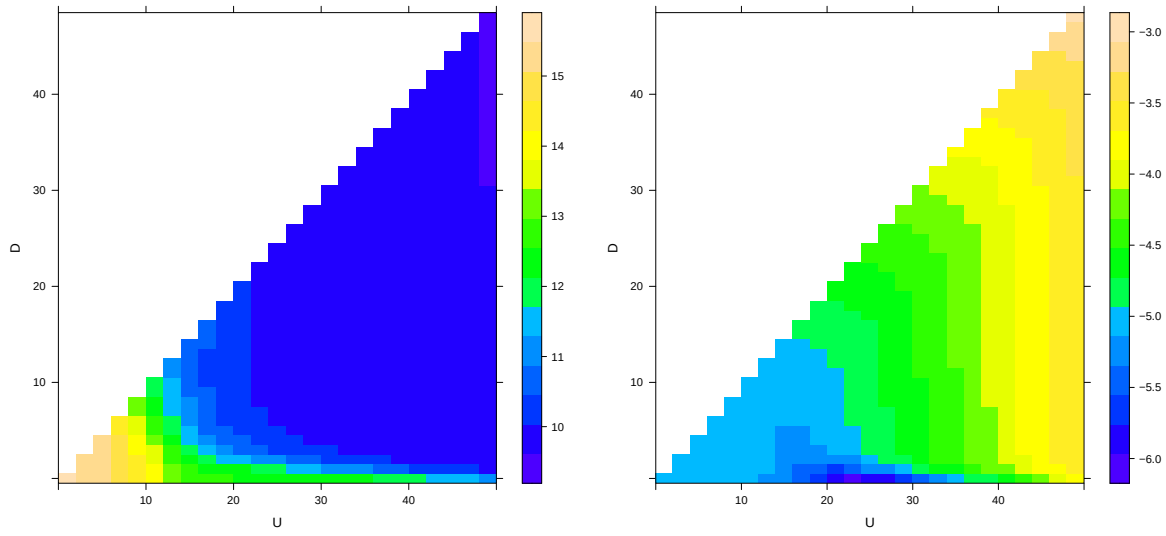
(a) (b)

Figure 2.4: Poissonian arrivals and Erlang servers activation: Mean power consumption (a) vs Loss probability (in log_{10}) (b)



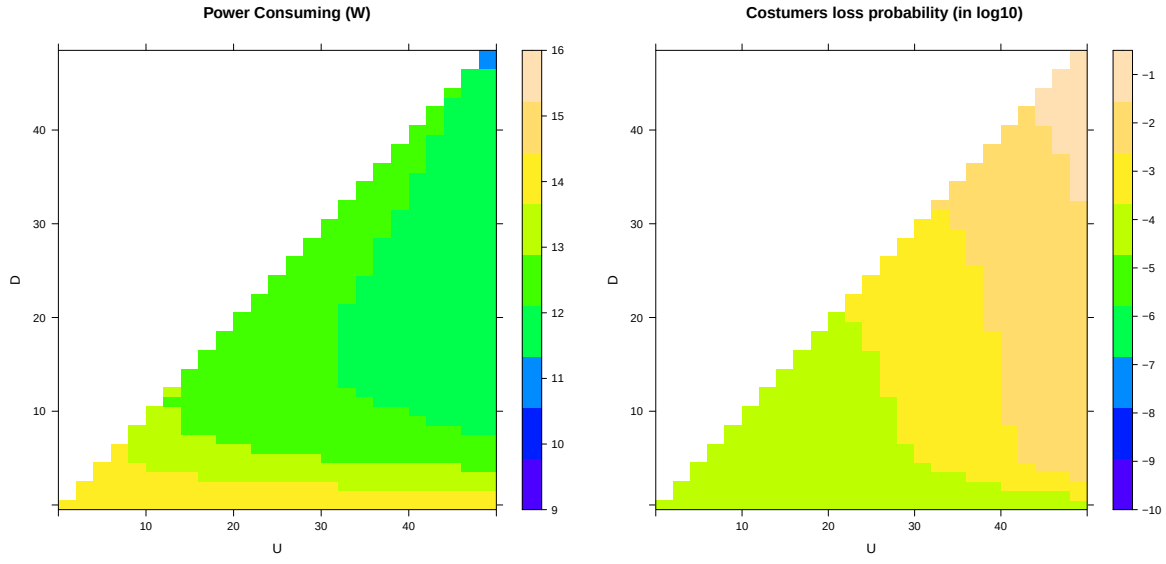
(a) (b)

Figure 2.5: Poissonian arrivals and Exponential servers activation: Mean power consumption (a) vs Loss probability (in log_{10}) (b)



(a) (b)

Figure 2.6: Poissonian arrivals and Hyper-Exponential servers activation: Mean power consumption (a) vs Loss probability (in log_{10}) (b)



(a) (b)

Figure 2.7: SBBP arrivals and Erlang servers activation: Mean power consumption (a) vs Loss probability (in log_{10}) (b)

Chapter 3

Dynamic Voltage and Frequency Scaling (DVFS) processor

Contents

3.1 Introduction

The AMD Opteron processor [Inc05] has a well-deserved reputation for high performance and low power consumption. Its industry-leading dynamic power management solution, AMD PowerNow technology, first became available in June 2000. In fact, AMD was the first company to introduce dynamic voltage and frequency scaling (DVFS) support capability in x86-based processors. This technology, which has been continually refined, delivers performance on demand and greatly reduces power consumption when full CPU performance is not needed. In its latest, most advanced form, AMD PowerNow technology with OPM (Optimized Power Management) allows the processor to run at multiple frequencies and voltages without changing the memory/front-side bus speed, under industry-standard ACPI (Advanced Configuration and Power Interface) program control. The multiple frequencies and voltages of processors are called "Pstates" (see Fig. 3.1). ACPI was originally developed for power management on notebook computers, but in its latest iteration, it has become an operating system-independent power management scheme with inherent multiprocessor support that can tailor each processor's power consumption level to its workload when the platform supports multiple Pstates. The highest Pstate runs the processor at full clock speed and full voltage. But during off-peak conditions, the clock can drop all the way back to a 1GHz "idle," saving as much as 75 percent of the full-speed power (See the original paper [Inc05] for more details).

States	P_1	P_2	P_3
Frequency	1 GHz	1.8 GHz	2 GHz
Power	≈ 32W	≈ 55W	≈ 65W

States	P_4	P_5	P_6
Frequency	2.2 GHz	2.4 GHz	2.6 GHz
Power	≈ 76W	≈ 90W	≈ 95W

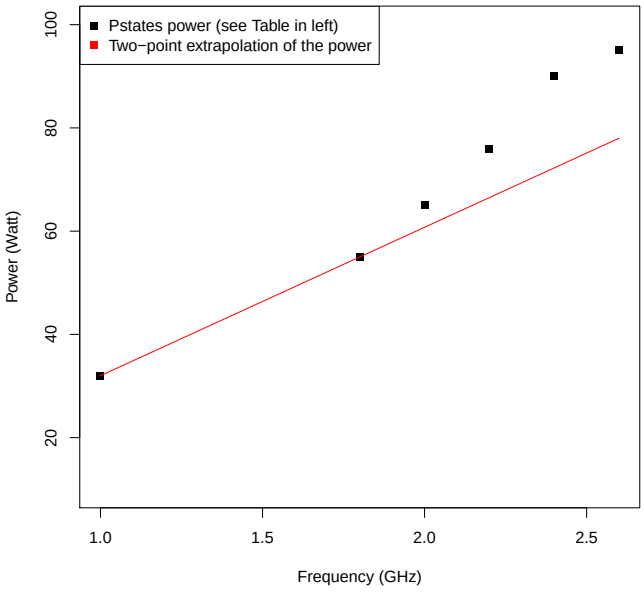


Figure 3.1: Pstates Support in AMD Opteron Processor [Inc05].

Fig. 3.1 represents the frequency (in GHz) that corresponds to the number of processor's instruction per time unit 't'. This table is obtained from AMD paper [Inc05]. We notice that the power consumption increases with the frequency of an Opteron processor in a non-linear approach. The Pstates power is a super-linearly function of the frequency.

Here, we suppose that Opteron processor's Pstates only depends on the tasks it performs. so the higher the workload, the higher the processor's speed. We also assume that:

- Task scheduling discipline is FCFS (First Come, First Served).
- External arrivals of tasks follow independent Poisson process with rate λ.
- Service rates are distributed according to exponential distributions,
- Tasks arriving into a processor may require 'a' instructions (with $a > 0$). So the service rate (i.e. the number of tasks executed per time unit by a processor's core), is $\frac{f(\mathrm{P}state_i)}{a}$ where 'f' is the frequency.
- The service rate will depends on the model considered. In Model 1 the service rate is fixed to the frequency of the Pstate considered, while in Model 2 and Model 3 the service rate changes and depends on the current workload in processors.

The aim of this work is to evaluate the response time and energy per job consumed in three distinct processor configurations. In the first configuration, which we call Model 1, a single Pstate is used. In Model 2, we consider two Opteron Pstates. We have proposed a closed form for steady-state distribution and we derived a cost function that uses both performance and energy consumption. We also show, through an algorithm we suggest, the best combination of Pstates and thresholds that minimizes the cost function. In the last section we investigate the case of an Opteron processor that uses all Pstates. Note that all steady-state distributions in this chapter are obtained using the birth-death resolution in Section 2.5.

3.2 Model 1: A multi-core processor with one Pstate

In this section, we consider that an Opteron processor has only one Pstate 'i'. Thus, a classical M/M/C queue can represent this model. The servers speed rate $\mu_i > 0$ is fixed and only depends on the chosen Pstate 'i'. We denote such a system S_i. We will study the performance and power consumption in each Pstate separately. Steady-state distribution and rewards as mean response time, mean number of tasks in the system are well known in the literature. Here, we show how to calculate, analytically, the mean power consumption and energy per job/task in the system.

We first recall some definitions about the 'st' comparison between probability vectors and CTMCs processes. In the following, we will use it to compare the mean number of jobs and the mean response time between different systems.

Definition 3.2.1 ***(Comparison of probability vectors [Sto83])***
Let $\boldsymbol{p}$ and $\boldsymbol{q}$ be two probability vectors of size M (each entry is a positive real such that the sum of entries is equal to 1), then

$$\boldsymbol{p} \leq_{st} \boldsymbol{q} \quad iff \quad \sum_{j=k}^{\mathrm{M}} \boldsymbol{p}[j] \leq \sum_{j=k}^{\mathrm{M}} \boldsymbol{q}[j] \quad , \quad \forall k \in \{1,2,\ldots,\mathrm{M}\}. \tag{3.1}$$

Definition 3.2.2 ***(Comparison of CTMCs [Sto83])***
Let $\{X_1^t,\ t \geq 0\}$ *and* $\{X_2^t,\ t \geq 0\}$ *be the two CTMCs, then*

$$\{X_1^t,\ t \geq 0\} \leq_{st} \{X_2^t,\ t \geq 0\},$$

iff

$$X_1^0 \leq_{st} X_2^0 \quad \Rightarrow \quad \forall \quad t > 0\ ,\ X_1^t \leq_{st} X_2^t. \tag{3.2}$$

Lemma 3.2.1 ***(Comparison of steady-state distributions [Sto83])***
Let $\{X_1^t,\ t \geq 0\}$ *and* $\{X_2^t,\ t \geq 0\}$ *be two CTMCs. Hence, if the steady-state distribution* Π_1 *(resp.* Π_2*) exists, then*

$$\{X_1^t,\ t \geq 0\} \leq_{st} \{X_2^t,\ t \geq 0\} \ \Rightarrow\ \Pi_1 \leq_{st} \Pi_2.$$

Theorem 3.2.1 ***(Comparison of birth-death processes [Sto83])***
Stoyan theorem states that: considering two homogeneous birth-death processes $\{X_1^t,\ t \geq 0\}$ *and* $\{X_2^t,\ t \geq 0\}$ *with Poisson arrivals rate* $\lambda_x^{(S_1)}$ *and* $\lambda_x^{(S_2)}$*, and Exponential service rates* $\mu_x^{(S_1)}$ *and* $\mu_x^{(S_2)}$*. Note that arrivals rate and service rate depends on the state x. If*

$$\forall\ state\ x \geq 0 \qquad \mu_x^{(S_1)} \geq \mu_x^{(S_2)} \quad and \quad \lambda_x^{(S_1)} \leq \lambda_x^{(S_2)},$$

then

$$\{X_1^t,\ t \geq 0\} \leq_{st} \{X_2^t,\ t \geq 0\}.$$

3.2.1 Mean number of jobs and response time

Let (x) be the number of jobs in the system. Under the classical assumptions we mention in Section 3.1, $\{X_i^t, t \geq 0\}$ is the CTMC that describes the system S_i. If the system is stable, then the steady-state distribution exists. Let Π_i be this distribution, and $\mathbb{E}[X_i]$ (resp. $\overline{T_i}$) be the mean number of jobs (resp. the mean response time). So

$$\mathbb{E}[X_i] = \sum_{x=0}^{\infty} x\Pi_i(x) \quad and \quad \overline{T_i} = \frac{\mathbb{E}[X_i]}{\lambda} \quad (Little's\ \ law). \tag{3.3}$$

Corollary 3.2.1 *We consider* $N = 6$ *stable systems (corresponding to the sixth Pstates in Fig. 3.1). Each system denoted by* S_i *represent a multi-core Opteron processor with the Pstate 'i'. Let* $\mu_1, \mu_2, \ldots, \mu_6$ *be processor's speed (i.e. service rate of cores in each system). Let* $\forall\ t \geq 0$ $\{X_1^t\}, \{X_2^t\}, \ldots, \{X_6^t\}$ *be the Markov chain, and* $\Pi_1, \Pi_2, \ldots, \Pi_6$ *the steady-state distribution for each stable multi-core Opteron processor, then*

$$\mu_1 \leq \mu_2, \ldots, \leq \mu_6 \quad \Rightarrow \quad \Pi_6 \leq_{st} \Pi_5, \ldots, \leq_{st} \Pi_1. \tag{3.4}$$

Proof. *The proof is directly derived from Theorem 3.2.1 and Lemma 3.2.1.*

In our systems we have the same arrivals rate λ *in every system* $S_1, S_2, \ldots, S_6$ *and it does not depend on the state of the system. Then for all the states 'x',* $\lambda_x^{(S_1)} = \lambda_x^{(S_2)} = , \ldots, = \lambda_x^{(S_6)} = \lambda$*. Also from the assumption, we have for all state x,* $\mu_x^{(S_1)} \leq \mu_x^{(S_2)} \leq, \ldots, \leq \mu_x^{(S_6)}$ *as a consequence of* $\mu_x^{(S_i)} = \mu_i$*, then all conditions are satisfied to apply the theorem cited above, therefore we get* $\forall\ t \geq 0$, $\{X_6^t\} \leq_{st} \{X_5^t\} \leq_{st}, \ldots, \leq_{st} \{X_1^t\}$*. Finally, since the systems are stable, then from Lemma 3.2.1 we conclude that* $\Pi_6 \leq_{st} \Pi_5, \ldots, \leq_{st} \Pi_1$.

Property 3.2.1 *Let* $\mathbb{E}[X_1],\mathbb{E}[X_2],\ldots,\mathbb{E}[X_6]$ *(resp.* $\overline{T_1},\overline{T_2},\ldots,\overline{T_6}$*) be the mean number of jobs (resp. the mean response time) for stable multi-core Opteron processors* $S_1,S_2,\ldots,S_6$ *that uses one Pstate, then*

$$\Pi_6 \leq_{st} \Pi_5,\ldots,\leq_{st} \Pi_1 \quad \Rightarrow \begin{cases} \mathbb{E}[X_1] \geq \mathbb{E}[X_2],\ldots,\geq \mathbb{E}[X_6] \\ \\ \overline{T_1} \geq \overline{T_2},\ldots,\geq \overline{T_6}. \end{cases} \tag{3.5}$$

Proof. *The systems are supposed to be stable, therefore the steady-state distribution exists. Let* $f(x) = x \ : \mathbb{N} \rightarrow \mathbb{N}$ *be an increasing function, then*

$$\Pi_6 \leq_{st} \Pi_5,\ldots,\leq_{st} \Pi_1 \ \Rightarrow \ \sum_{x\geq 0} x\Pi_1(x) \geq \sum_{x\geq 0} x\Pi_2(x) \geq,\ldots,\geq \sum_{x\geq 0} x\Pi_6(x),$$

and using Equation (3.3) , we get

$$\Pi_6 \leq_{st} \Pi_5,\ldots,\leq_{st} \Pi_1 \ \Rightarrow \mathbb{E}[X_1] \geq \mathbb{E}[X_2],\ldots,\geq \mathbb{E}[X_6].$$

Mean response time is obtained using Little's law for the jobs in the system (see Equation (3.3)). Since the systems receive the same traffic rate λ *then we only have to divide the mean number of jobs* $\mathbb{E}[X_i]$ *by* λ *to get the mean response time. Therefore*

$$\Pi_6 \leq_{st} \Pi_5,\ldots,\leq_{st} \Pi_1 \ \Rightarrow \overline{T_1} \geq \overline{T_2},\ldots,\geq \overline{T_6}.$$

Corollary 3.2.2 *The result of this proposition is the conjunction of Corollary 3.2.1 and Corollary 3.2.1. Under stability condition of systems, we state that a higher speed of the Opteron processors (so higher Pstates) implies a less mean number of jobs and response time.*
If

$$\mu_1 \leq \mu_2,\ldots,\leq \mu_6,$$

then

$$\begin{cases} \mathbb{E}[X_1] \geq \mathbb{E}[X_2],\ldots,\geq \mathbb{E}[X_6] \\ \\ \overline{T_1} \geq \overline{T_2},\ldots,\geq \overline{T_6}. \end{cases} \tag{3.6}$$

No proof is needed since Equation (3.6) is the result of the conjunction of Equation (3.4) and Equation (3.5).

3.2.2 Power and energy consumption

Let $p_1,...,p_6$ be the power consumption corresponding to the systems $S_1,...,S_6$ (see Fig. 3.1). A server (i.e. core) in the idle state provides a power gain of 75%. Let p_i $(resp.\ p_{i,Id})$ for $i \in \{1,\ldots,6\}$ be the Opteron Pstate power (resp. Pstate idle power) used by a core in a system S_i, then

$$p_{i,Id} = \alpha p_i, \tag{3.7}$$

precisely $\alpha = 0.25$ in AMD features.

Lemma 3.2.2 *The mean power consumption of a stable mono Pstate system* PW_i *is the sum the mean power of the servers in activity* $\mathrm{PW}_i^{(a)}$ *and the idle power* $\mathrm{PW}_i^{(\mathrm{I}d)}$*:*

$$\mathrm{PW}_i = \mathrm{PW}_i^{(a)} + \mathrm{PW}_i^{(\mathrm{I}d)}. \tag{3.8}$$

Which can be expressed as :

$$\mathrm{PW}_i = p_{i,\mathrm{I}d}\mathrm{C} + \frac{(p_i - p_{i,\mathrm{I}d})\lambda}{\mu_i}. \tag{3.9}$$

Proof. *Let* (x) *be the number of jobs in system* S_i*, and* $\Pi_i(x)$ *be the steady-state distribution of the stable Markov chain* $\{\mathrm{X}_i^t,\ t \geq 0\}$*, then*

$$\begin{cases} \mathrm{PW}_i^{(a)} = \sum_{x=0}^{\infty} \Pi_i(x) \Big[p_i * min\{x, \mathrm{C}\} \Big], \\ \mathrm{PW}_i^{(\mathrm{I}d)} = \sum_{x=0}^{\infty} \Pi_i(x) \Big[(\mathrm{C} - min\{x, \mathrm{C}\}) * p_{i,\mathrm{I}d} \Big]. \end{cases} \tag{3.10}$$

$\sum_{x=0}^{\infty} \Pi_i(x) \Big[min(x, \mathrm{C}) \Big]$ *represents the mean number of servers in activity of the system* S_i*. The mean number of jobs in service corresponds to the mean number of servers in activity since a job is served by one server at each time. We will use Little's law for servers in activity, which states that the mean number of jobs in service is the mean service time* $\frac{1}{\mu_i}$ *(since servers are homogeneous) times the mean arrivals rate* λ*. Then we have* $\sum_{x=0}^{\infty} \Pi_i(x) \Big[min(x, \mathrm{C}) \Big] = \frac{\lambda}{\mu_i}$*. After simplifications, Equation (3.10) becomes:*

$$\begin{cases} \mathrm{PW}_i^{(a)} = \frac{\lambda p_i}{\mu_i} \\ \mathrm{PW}_i^{(\mathrm{I}d)} = \left(\mathrm{C} - \frac{\lambda}{\mu_i} \right) p_{i,\mathrm{I}d} \end{cases} \tag{3.11}$$

By summing $\mathrm{PW}_i^{(a)}$ *and* $\mathrm{PW}_i^{(\mathrm{I}d)}$ *we get the expression of the mean power consumption denoted in Equation (3.9). The proof is complete.*
Notice that the model we study here is a stable a M/M/C queue, and by definition the stability condition of the system is $\lambda < \mathrm{C}\mu_i$ *so* $(\mathrm{C} - \frac{\lambda}{\mu_i}) > 0$ *then* $\mathrm{PW}_i^{(\mathrm{I}d)} > 0$*, therefore* $\mathrm{PW}_i > 0$*.*

Lemma 3.2.3 *Let* $\mathrm{E}_i^{(Job)}$ *be the energy consumption per job in a stable system* S_i*. then*

$$\mathrm{E}_i^{(Job)} = \frac{\mathrm{PW}_i}{\lambda}. \tag{3.12}$$

Proof. *The energy consumption of a device is the power consumed on a period of time. Therefore we expressed the energy consumption of a stable Opteron system as the mean power consumption* PW_i *times the mean resident time of a task* $\overline{\mathrm{T}_i}$*. Let* $\mathrm{E}ng_i$ *be this energy*

$$\mathrm{E}ng_i = \mathrm{PW}_i * \overline{\mathrm{T}_i}.$$

$\mathrm{E}ng_i$ *is the energy consumption of the system when considering all jobs. Hence, in order to obtain the energy consumption per one job,*

$$\mathrm{E}_i^{(Job)} = \frac{\mathrm{E}ng_i}{\mathbb{E}[\mathrm{X}_i]}.$$

Using Little's law for the mean number of jobs in a steady-state system ($\mathbb{E}[\mathrm{X}_i] = \lambda\overline{\mathrm{T}_i}$), then by substitution we get Equation (3.12), the proof is complete.

Lemma 3.2.4 *Let $\mathrm{E}_i^{(Job)}$ (resp. $\mathrm{E}_j^{(Job)}$) be the energy consumption per job of a stable system S_i (resp. S_j). Also let PW_i (resp. PW_j) be the corresponding mean power consumption. In below, we present a sufficient and necessary condition for the comparison of the mean power and the energy per job consumption:*

$$\frac{p_i}{p_j} \leq \frac{\mu_i g(\mu_j)}{\mu_j g(\mu_i)} \quad \Longleftrightarrow \quad \begin{cases} \mathrm{PW}_i \leq \mathrm{PW}_j \\ \mathrm{E}_i^{(Job)} \leq \mathrm{E}_j^{(Job)} \end{cases} \tag{3.13}$$

where

$$g(\mu_i) = \alpha\mu_i \mathrm{C} + (1-\alpha)\lambda. \tag{3.14}$$

Proof. *From equation (3.9) we have*

$$\mathrm{PW}_i = p_{i,\mathrm{Id}}\mathrm{C} + \frac{(p_i - p_{i,\mathrm{Id}})\lambda}{\mu_i},$$

and using $p_{i,\mathrm{Id}} = \alpha p_i$ we get

$$\mathrm{PW}_i = \left(\frac{\alpha\mu_i \mathrm{C} + (1-\alpha)\lambda}{\mu_i}\right) p_i.$$

then

$$\mathrm{E}_i^{(Job)} \leq \mathrm{E}_j^{(Job)} \quad \Leftrightarrow \quad \mathrm{PW}_i \leq \mathrm{PW}_j \quad \Leftrightarrow \quad \frac{p_i}{p_j} \leq \frac{(\alpha\mu_j \mathrm{C} + (1-\alpha)\lambda)\mu_i}{(\alpha\mu_i \mathrm{C} + (1-\alpha)\lambda)\mu_j} \tag{3.15}$$

and using Equation (3.14) the proof is complete.

Lemma 3.2.5 *In this lemma we present a sufficient condition for the comparison of the mean power and the energy per job consumption in two stable systems S_i and S_j (with $i \leq j$):*

$$\mu_i \leq \mu_j \quad \textit{and} \quad \frac{p_i}{p_j} \leq \frac{\mu_i}{\mu_j} \quad \Longrightarrow \quad \begin{cases} \mathrm{PW}_i \leq \mathrm{PW}_j \\ \mathrm{E}_i^{(Job)} \leq \mathrm{E}_j^{(Job)}. \end{cases} \tag{3.16}$$

Proof. *Equation (3.13) in Lemma 3.2.4 presents a sufficient and necessary condition. Therefore:*

$$\frac{p_i}{p_j} \leq \frac{\alpha\mu_i\mu_j \mathrm{C} + (1-\alpha)\lambda\mu_i}{\alpha\mu_i\mu_j \mathrm{C} + (1-\alpha)\lambda\mu_j} \quad \Rightarrow \quad \begin{cases} \mathrm{PW}_i \leq \mathrm{PW}_j \\ \mathrm{E}_i^{(Job)} \leq \mathrm{E}_j^{(Job)}. \end{cases} \tag{3.17}$$

Hence, We only have to verify that

$$\frac{\mu_i}{\mu_j} \leq \frac{\alpha\mu_i\mu_j C + (1-\alpha)\lambda\mu_i}{\alpha\mu_i\mu_j C + (1-\alpha)\lambda\mu_j}. \tag{3.18}$$

Equation (3.18) *can be expressed as*

$$\alpha\mu_i^2\mu_j C + (1-\alpha)\lambda\mu_i\mu_j \leq \alpha\mu_i\mu_j^2 C + (1-\alpha)\lambda\mu_i\mu_j.$$

After simplification of terms we obtain

$$\mu_i \leq \mu_j,$$

which is the assumption stated in the presentation of this lemma. Hence, from Equation (3.17) *and* (3.18) *we obtain Equation* (3.16)*, and the proof is complete.*

3.2.3 Numerical comparison of the six Opteron Pstates

We consider here six systems $S_1,\ldots,S_6$, each system has $C = 20$ cores. The speed of the servers (i.e. cores) depends on the Pstate performed. Also, here we suppose that each task requires one core's instruction $a = 1$, therefore $\mu_i = f_i$.

3.2.3.1 Performance, power and Energy per job

In Fig. 3.2 we observe that, (a) by increasing the task's arrivals load, the system fills up more and the waiting time in the queue increases. (b) In stable cases of each system, Pstate6 (contrary to Pstate1) presents the best results in terms of performance i.e. mean number of tasks and waiting time in the network, but requires the highest power consumption (see Fig. 3.3). In the left figure in Fig. 3.3 we observe that Idle power decreases by increasing the load in the system, but active power will increase (since the servers will be activated as depicted in the right figure). These numerical results are clearly matching with our analytical results (Corollary 3.2.2, Lemma 3.2.4 and Lemma 3.2.5). The left figure of Fig. 3.4 represents the energy per job consumption when increasing the traffic. This consumption clearly decreases with an increase in the load (see Lemma 3.2.3).

3.2.3.2 The appropriate Pstate in performance and energy trade-off

In the right figure, we presented the mean waiting time as a function of the mean power consumption. This figure give us various information: as the minimal and the maximal power consumption requirement if using any Pstate and the corresponding delay. Also for a given delay (resp. power consumption value) we can derive the Pstate that consumes the least power consumption (resp. delay) as the goal is to minimize both of performance and mean power (resp. energy per job) consumption.

3.2.3.3 Condition verification for power and energy comparison

In Table 3.1, we show a verification of the condition cited in Lemma 3.2.5. This condition is based on the Pstates configurations of an Opteron processor (see Fig.

3.1). Note that in Lemma 3.2.5, we only need the core's speed and core's power consumption to make a comparison of the mean power consumption and the energy per job consumption between two single-Pstate Opteron processors. This comparison is sufficient and not necessary, which means that when $\mu_i \leq \mu_j$ and $\frac{p_i}{p_j} \leq \frac{\mu_i}{\mu_j}$ is verified, then Pstate 'i' consumes less mean power and energy per job consumption than the system with Pstate 'j'. Otherwise, we don't dispose of much information to compare the two systems. In that case, we can use Lemma 3.2.4, which, additionally, includes traffic arrivals rate and other parameters.

For the verified cases in Table 3.1, Lemma 3.2.5 ensures that $PW_i \leq PW_j$ and $E_i^{(job)} \leq E_j^{(job)}$. Otherwise, we use Lemma 3.2.4. This Lemma considers λ the arrivals rate (which is supposed to be the same in both systems). The non-verified cases for Lemma 3.2.5 are the couple of Pstates $i = 1, j = 2$ and $i = 5, j = 6$. For $i = 1, j = 2$, Lemma 3.2.4 ensures $PW_1 \leq PW_2$ (and $E_1^{(job)} \leq E_2^{(job)}$) for all values of λ that makes both systems stable. But for the case of $i = 5$ and $j = 6$ the behavior of the mean power and energy per job changes with the value of λ. Hence, when applying Lemma 3.2.4 with $\lambda \leq 34$ we obtain $PW_5 \leq PW_6$ (and $E_5^{(Job)} \leq E_6^{(Job)}$). Also, when $\lambda > 34$ then $PW_6 < PW_5$. That explains the crossover behavior between the red and the green curve in the right figure of Fig. 3.3.

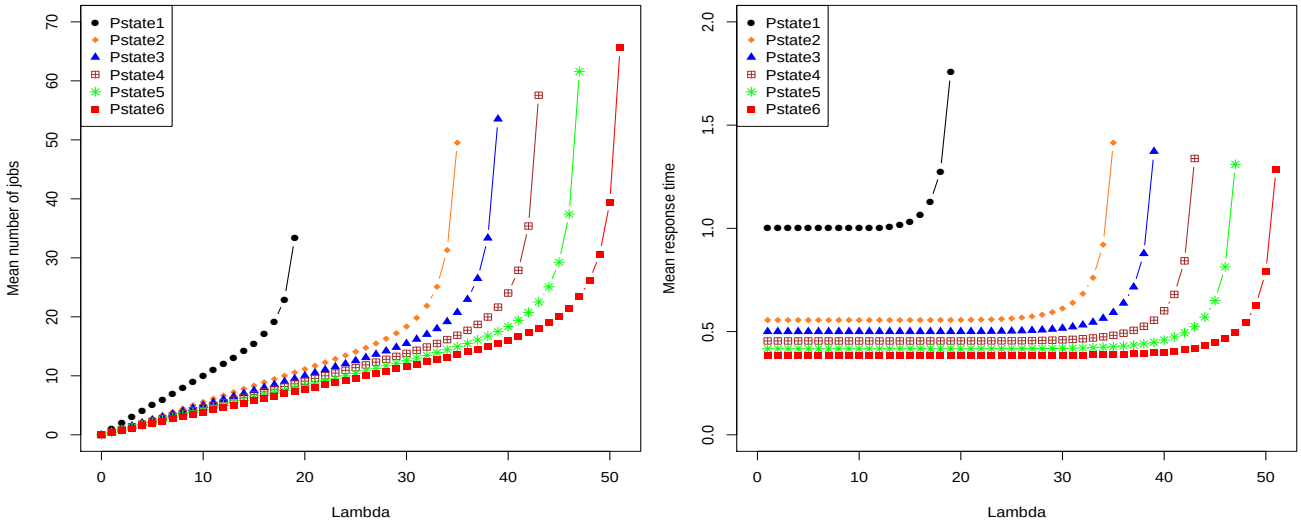


Figure 3.2: Mean number of jobs (left) and mean response time (right).
Parameters: $C = 20$ servers, server's rate (i.e. core's speed) in each Pstate is depicted in Fig. 3.1.

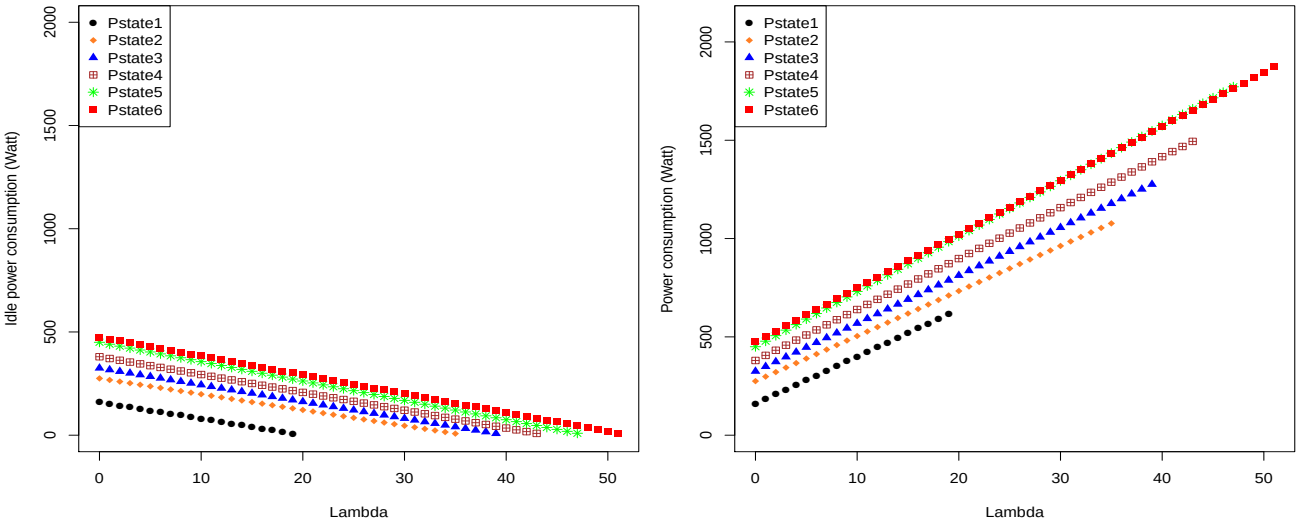


Figure 3.3: Idle power consumption (left) and total power consumption (right). Parameters are the same as in Fig. 3.2.

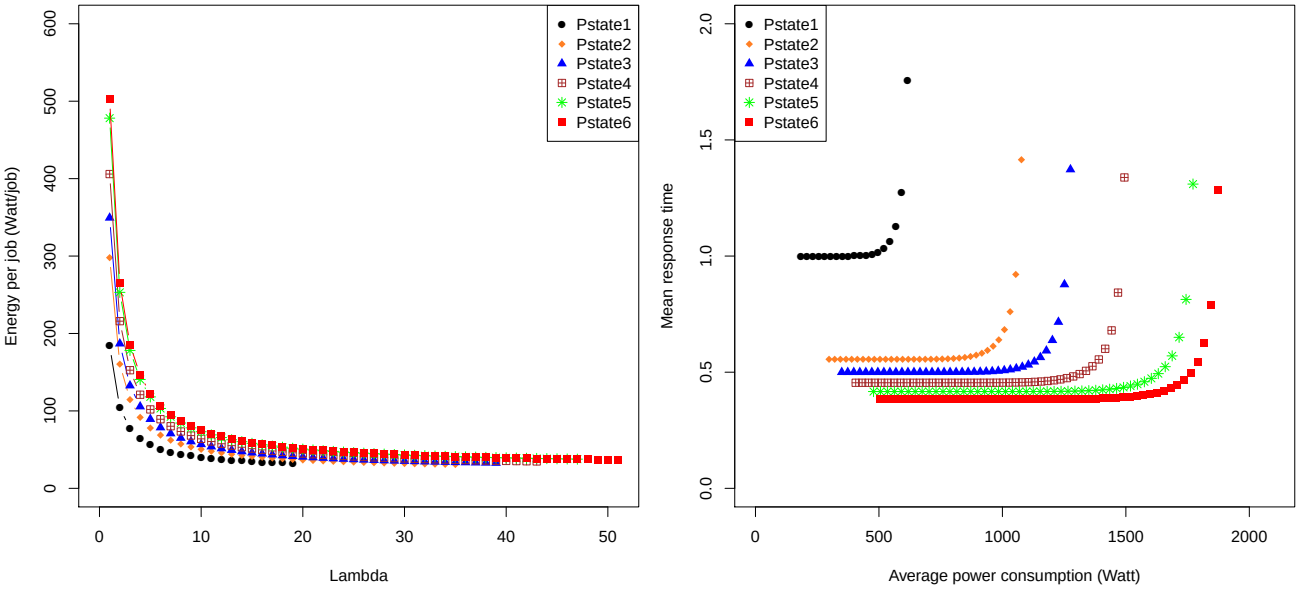


Figure 3.4: Energy per job consumption (left) and response time as a function of mean power consumption. Parameters are the same as in Fig. 3.2.

Pstate 'i'	**Pstate 'j'**	$\frac{p_i}{p_j} \leq \frac{\mu_i}{\mu_j}$
1	1	Verified
1	2	Not verified
1	3	Verified
1	4	Verified
1	5	Verified
1	6	Verified
2	2	Verified
2	3	Verified
2	4	Verified
2	5	Verified
2	6	Verified
3	3	Verified
3	4	Verified
3	5	Verified
3	6	Verified
4	4	Verified
4	5	Verified
4	6	Verified
5	5	Verified
5	6	Not verified
6	6	Verified

Table 3.1: The examination of the condition of Lemma 3.2.5 for all pairs of Pstates (i,j) where $\mu_i \leq \mu_j$. Note that μ_i and p_i are obtained from the AMD table in Fig. 3.1.

3.3 Model 2: A multi-core processor with N = 2 Pstates

In this model we consider a birth-death process with two Pstates 'i' and 'j' (N = 2). We suppose that Pstate 'j' consumes more power than Pstate 'i' i.e. Pstate 'j' executes instructions faster than Pstate 'i' (see Fig. 3.1). Servers speed rate depends on the number of tasks in the system. It means that if the number of tasks is below a certain threshold 'th', then servers (cores) speed corresponds to μ_i otherwise, it is supposed that the queue is well loaded therefore servers speed rate are fixed to μ_j. We denote such a system $S(i,j,th)$ and we suppose the $i \leq j$. Stability condition of this model, that verifies the existence of a steady-state distribution, is $\lambda < C\mu_j$.

3.3.1 Closed form for the steady-state distribution

The processor contains C cores. Therefore $min\{C,x\}$ task are in service and $(x - min\{C,x\})^+$ are queued. Under the classical assumptions we mention in Section 3.1, $\{X^t_{i,j,th},\ t \geq 0\}$ is a Markov chain, and the transitions are as follows :

$$\begin{aligned} (x) &\rightarrow (x+1) \text{ with rate } \lambda, \\ (x) &\rightarrow (\max\{0, x-1\}), \text{ with rate } min\{x, C\}\cdot\mu(x). \end{aligned}$$

and service rate $\mu(x)$ is expressed as :

$$\mu(x) = \begin{cases} \mu_i = \dfrac{f_i}{a} & if\ (x \leq th), \\ \mu_j = \dfrac{f_j}{a} & if\ (x > th). \end{cases} \tag{3.19}$$

f_i is the frequency (GHz) given in Fig. 3.1.

Theorem 3.3.1 *The transition diagram of this Markov chain is a birth-death process. Then, under stability condition (i.e. $\lambda < C\mu_j$), we can derive $\Pi_{i,j,th}(x)$ the steady-state probability of the two Pstates Opteron processor $S(i,j,th)$.*
Let

$$\Psi(x) = \prod_{k=1}^{x} min\{k, C\}\mu(k),$$

and

$$R = max\{C, th\}, \tag{3.20}$$

then

$$\Pi_{i,j,th}(x) = \begin{cases} \frac{\lambda^x}{\Psi(x)}\Pi_{i,j,th}(0) & \forall \quad 0 < x \leq R, \\ \lambda^x \left[(C\mu_j)^{x-R}\Psi(R)\right]^{-1}\Pi_{i,j,th}(0) & \forall \quad x > R, \end{cases} \tag{3.21}$$

where

$$\Pi_{i,j,th}(0) = \left[1 + \frac{\lambda^{R+1}}{(C\mu_j - \lambda)\Psi(R)} + \sum_{x=1}^{R} \frac{\lambda^x}{\Psi(x)}\right]^{-1}. \tag{3.22}$$

Proof. *The Markov chain we consider is a birth-death process. We also set 'R' (Equation (3.20)) the threshold at which the system maintains it's service rate (which is* $C\mu_j$*). This simplifies the calculation of the steady state distribution. Hence, we have divided the calculation of* $\Pi_{i,j,th}(x)$ *into two parts. States from* 1 *to 'R' and states from 'R' to* $+\infty$*. From classical equations in a birth-death process, we deduce that:*

$$\Pi_{i,j,th}(x) = \begin{cases} \dfrac{\lambda}{min\{x, C\}\mu(x)}\Pi_{i,j,th}(x-1) & \forall \quad 0 < x \leq R, \\ \\ \left(\dfrac{\lambda}{C\mu_j}\right)^{x-R}\Pi_{i,j,th}(R) & \forall \quad x > R. \end{cases} \tag{3.23}$$

After simple substitutions in Equation (3.23) (i.e. to write $\Pi_{i,j,th}(x)$ *as a function of* $\Pi_{i,j,th}(0)$ *for all* $x \in [1, +\infty[$*), we get Equation (3.21). Finally, as the system is supposed to be stable, then Equation (3.22) is obtained after the normalization of probabilities (* $\sum_{x=0}^{+\infty}\Pi_{i,j,th}(x) = 1$*).*

3.3.2 Mean number of jobs and response time

Here we seek to compare the mean number of jobs and mean response time between two stable systems $S(i, j, th1)$ with $i \leq j$ and $S(k, l, th2)$ with $k \leq l$. Note that, to calculate the mean number of jobs $\mathbb{E}[X_{i,j,th}]$ and mean response time $\overline{T}_{i,j,th}$, we use the same approach as in Equation (3.3).

Corollary 3.3.1 *We consider the two stable systems described above, then for any thresholds th1 and th2, we have:*

$$j \leq k \quad \Rightarrow \quad \begin{cases} \mathbb{E}[X_{k,l,th2}] \leq \mathbb{E}[X_{i,j,th1}] \\ \\ \overline{T}_{k,l,th2} \leq \overline{T}_{i,j,th1}. \end{cases} \tag{3.24}$$

Proof. *Let* $\{X^t_{i,j,th1},\ t \geq 0\}$ *(resp.* $\{X^t_{k,l,th2},\ t \geq 0\}$*) the Markov chain of the stable system* $S(i, j, th1)$ *(resp.* $S(k, l, th2)$*). Also, let* $\Pi_{i,j,th1}$ *and* $\Pi_{k,l,th2}$ *be the steady-state distributions of both systems. Let* $\lambda_x^{S(i,j,th1)}$, $\lambda_x^{S(k,l,th2)}$ *(resp.* $\mu_x^{S(i,j,th1)}$, $\mu_x^{S(k,l,th2)}$*) be the arrivals rate (resp. service rate) generated at the state 'x' in the system* $S(i, j, th1)$ *and* $S(k, l, th2)$*.*

- *We first derive the 'st' comparison between the two systems. The arrivals rate* λ *is supposed the same. Then for all the states 'x',* $\lambda_x^{S(i,j,th1)} = \lambda_x^{S(k,l,th2)} = \lambda$*. Also, we have* $j \leq k$ *from the assumption, and* $i \leq j$ *and* $k \leq l$ *from the definition of systems, then* $i \leq j \leq k \leq l$*. It means that, for any values of 'th1' and 'th2', services rates in* $S(k, l, th2)$ *are always greater than the ones in* $S(i, j, th1)$*, so* $\mu_x^{S(k,l,th2)} \geq \mu_x^{S(i,j,th1)}$*. From Theorem 3.2.1 we deduce that*

$$\{X^t_{k,l,th2},\ t \geq 0\} \leq_{st} \{X^t_{i,j,th1},\ t \geq 0\}.$$

- *Both systems are supposed stable, then using Lemma 3.2.1, we have*

$$\{X^t_{k,l,th2},\ t \geq 0\} \leq_{st} \{X^t_{i,j,th1},\ t \geq 0\} \Rightarrow \Pi_{k,l,th2} \leq_{st} \Pi_{i,j,th1}.$$

- *Finally, the same approach of property 3.2.1 is used to get*

$$\Pi_{k,l,th2} \leq_{st} \Pi_{i,j,th1} \quad \Rightarrow \quad \begin{cases} \mathbb{E}[X_{k,l,th2}] \leq \mathbb{E}[X_{i,j,th1}] \\ \\ \overline{T}_{k,l,th2} \leq \overline{T}_{i,j,th1}. \end{cases}$$

By combining the three last equations we obtain Equation (3.24), that concludes the proof.

Corollary 3.3.2 *In this corollary, we use another assumption, that concerns two stable systems using the same Pstates and different thresholds* $S(i,j,th1)$ *and* $S(i,j,th2)$. *We have:*

$$th1 \leq th2 \quad \Rightarrow \quad \begin{cases} \mathbb{E}[X_{i,j,th1}] \leq \mathbb{E}[X_{i,j,th2}] \\ \\ \overline{T}_{i,j,th1} \leq \overline{T}_{i,j,th2}. \end{cases} \tag{3.25}$$

Proof. *The approach of the proof is similar to the one in Corollary 3.3.1.*

- *Arrivals rate* λ *is identical in both systems, then for all the states 'x',* $\lambda_x^{S(i,j,th1)} = \lambda_x^{S(k,l,th2)} = \lambda$. *Also, we have from the assumption of the corollary that* $th1 \leq th2$. *It means that Opteron system* $S(i,j,th1)$ *will activate the higher Pstate 'j' earlier than system* $S(i,j,th2)$. *And as shown in Fig. 3.1, server's (core's) speed is an increasing function of the Pstates. Hence, for all states 'x',* $\mu_x^{S(i,j,th1)} \geq \mu_x^{S(i,j,th2)}$. *From Theorem 3.2.1 we deduce that*

$$\{X^t_{i,j,th1},\ t \geq 0\} \leq_{st} \{X^t_{i,j,th2},\ t \geq 0\}.$$

- *Using Lemma 3.2.1, and Property 3.2.1 for two Pstates systems. We obtain Equation (3.25), and the proof is complete.*

3.3.3 Power and energy consumption

Let p_i (resp. p_j) be the power consumption corresponding to Pstate i (resp. Pstate j) (see Fig. 3.1), with $i \leq j$. The power consumed by the processor depends on the speed of its cores, which is a function of the number of tasks assigned to it. Let (x) be the number of tasks in the processor, then the power consumed by each core, when hosting x tasks, is :

$$\begin{cases} p_i & if\ (x \leq th), \\ p_j & if\ (x > th). \end{cases} \tag{3.26}$$

Lemma 3.3.1 *Let (x), with* $x \in [0, +\infty[$, *be the number of jobs in the system.* $\{X^t_{i,j,th},\ t \geq 0\}$ *is an CTMC that under stability condition (mentioned in the description section of this model) admits a steady-state distribution* $\Pi_{i,j,th}(x)$. *Let* $PW_{i,j,th}$ *be the mean power consumption of the system* $S(i,j,th)$, *then*

$$\begin{aligned} PW_{i,j,th} &= p_i(1-\alpha)\Big[p_i \textstyle\sum_{x=0}^{th} min(x,C)\Pi_{i,j,th}(x) + p_j \sum_{x=th+1}^{+\infty} min(x,C)\Pi_{i,j,th}(x)\Big] \\ &+\alpha C\Big[p_j + (p_i - p_j)\textstyle\sum_{x=0}^{th}\Pi_{i,j,th}(x)\Big]. \end{aligned} \tag{3.27}$$

Proof. *Let* $\mathrm{PW}^{(a)}_{i,j,th}$ *(resp.* $\mathrm{PW}^{(\mathrm{Id})}_{i,j,th}$*) be the mean power consumption of the servers in activity state (resp. Idle state). Hence, as in Lemma 3.2.2, we have*

$$\mathrm{PW}_{i,j,th} = \mathrm{PW}^{(a)}_{i,j,th} + \mathrm{PW}^{(\mathrm{Id})}_{i,j,th} \ , \tag{3.28}$$

where

$$\begin{cases} \mathrm{PW}^{(a)}_{i,j,th} = p_i \sum_{x=0}^{th} min(x,\mathrm{C})\Pi_{i,j,th}(x) + p_j \sum_{x=th+1}^{+\infty} min(x,\mathrm{C})\Pi_{i,j,th}(x), \\ \mathrm{PW}^{(\mathrm{Id})}_{i,j,th} = p_{i,\mathrm{Id}} \sum_{x=0}^{th} (\mathrm{C} - min(x,\mathrm{C}))\Pi_{i,j,th}(x) + p_{j,\mathrm{Id}} \sum_{x=th+1}^{+\infty} (\mathrm{C} - min(x,\mathrm{C}))\Pi_{i,j,th}(x). \end{cases} \tag{3.29}$$

After simple simplifications using Equation (3.7) in Equation (3.29). Also by the substitution of $\mathrm{PW}^{(a)}_{i,j,th}$ *and* $\mathrm{PW}^{(\mathrm{Id})}_{i,j,th}$*. Then Equation (3.28) is expressed as Equation (3.27) and the proof is complete.*

Notice that Lemma 3.2.3 for energy consumption per job still hold for this model. Let $\mathrm{E}^{(\mathrm{Job})}_{i,j,th}$ be this energy, therefore

$$\mathrm{E}^{(\mathrm{Job})}_{i,j,th} = \frac{\mathrm{PW}_{i,j,th}}{\lambda}. \tag{3.30}$$

.

3.3.4 Optimization of energy consumption and response time

- From Corollary 3.3.2, we conclude that when comparing two stable systems $\mathrm{S}(i,j,th1)$ and $\mathrm{S}(i,j,th2)$ with $th1 \leq th2$ then $\overline{\mathrm{T}}_{i,j,th1} \leq \overline{\mathrm{T}}_{i,j,th2}$. Therefore, the mean response time (and mean number of jobs) is an increasing function of the threshold. Hence, to minimize the mean response time in a two-Pstates system, $th = 1$ is the optimal threshold to use.

- The 'st' comparison does not hold for the mean power consumption (resp. energy per job) function. The function is not monotone. In Fig. 3.5 we show the evolution of the mean power consumption with the increase of thresholds. In that example we consider $\lambda = 30$, $\mathrm{C} = 20$ servers, $i = 5$, $j = 6$, and $\mu_i = 2.4$, $\mu_j = 2.6$ (see Fig. 3.1). Note that, specially in that example the mean power function behaves as a convex function. This behavior is not always true.

- So in order to optimize the mean power consumption (resp. energy consumption), we shall consider an exhaustive algorithm looking for the best threshold.

In the following, we suggest merging the mean response time $\overline{\mathrm{T}}_{i,j,th}$ and the energy per job consumption $\mathrm{E}^{(job)}_{i,j,th}$ in a single function to minimize. Let $c1$ (resp. $c2$) be the cost of the mean response time (resp. energy per job consumption). Let Ω be the total cost to minimize. Ω is normalized through costs $c1$ and $c2$.

$$\Omega = \overline{\mathrm{T}}_{i,j,th} * c1 + \mathrm{E}^{(job)}_{i,j,th} * c2 \tag{3.31}$$

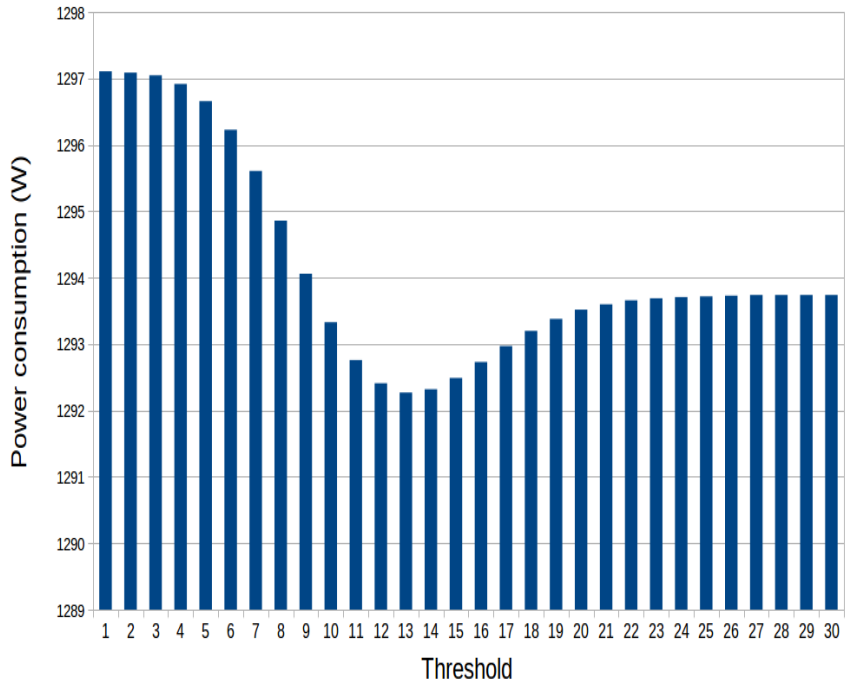


Figure 3.5: The mean power consumption as a function of the threshold.

To analyze Equation (3.31), we propose the algorithm below. This algorithm, under stability constraint, generates the best threshold (in the range [1,...,THMAX]) to use for each couple of Pstates (i,j). Given the input parameters: number of servers C, services rate (inspired from Fig. 3.1), an upper bound THMAX for the thresholds, arrival rate of tasks λ, and the rewards costs $c1$ and $c2$.

Algorithm 1: Purchasing the best threshold for each two-pstates Opteron system

Input : Number of servers C, arrivals rate λ, rewards cost $c1$, $c2$, and a value of THMAX.

Output: Threshold that minimizes cost function Ω for each couple of Pstates (i, j).

for (i, j) ***with*** $1 \leq i \leq j \leq 6$ **do**
  if $\lambda < C\mu_j$ **then** `// The system is stable`
    initiate 'th^*';
    for $th \leftarrow 1$ **to** THMAX **do**
      1) Calculate the steady-state distribution (Equation (3.21) and (3.22)) for the system $S(i, j, th)$;
      2) Derive the mean response time $\overline{T}_{i,j,th}$ (Equation (3.3)) and energy per job consumption $E^{(job)}_{i,j,th}$ (Equation (3.30));
      3) Calculate the cost function Equation (3.31) ;
      4) Update 'th^*' if the current cost function is lower than the cost function for the previous iteration.
    end
    Print "The best threshold for the couple of Pstates (i,j) is 'th^*' ;
  else
    Print "The system is not stable for the couple of Pstates (i,j)";
  end
end

3.3.5 Numerical results

In Fig. 3.6 and Fig. 3.7 we investigate the influence of cost values $c1$ and $c2$ on pstates (i,j) and their optimal thresholds. We considered the following parameters: arrivals rate $\lambda = 20$, number of servers $C = 20$, and the maximal threshold to purchase is $THMAX = 100$. Services rate are inspired from Fig. 3.1 with $a = 1$ (see Equation (3.19)). We present here two experiments:

- In the first experiment (Fig, 3.6), we consider higher costs $c1$ for the response time and lower costs $c2$ for the energy per job. In particular, $c1 = 200$ and $c2 = 1$ in the left heat-map of Fig. 3.6, we also took $c1 = 500$ and $c2 = 1$ in the right heat-map.

- In the second experiment (Fig. 3.7) we focus on the energy per job consumption, by adopting higher costs $c2$ for the energy and lower costs $c1$ for the response time. In the left heat-map of Fig. 3.7 we take $c1 = 1$ and $c2 = 10$, and in the right one $c1 = 1$ and $c2 = 50$.

Note that, the following results are derived from Algorithm 1. In both figures below, white areas are due to the relation $i \leq j$ which is an assumption of a two-Pstates (i,j) system. Colored square areas are the values of the objective function (Equation (3.31)). Blue integer in colored areas is the value of the optimal threshold we obtained (in the range $[1,\ldots,THMAX]$) that minimizes the objective function for the corresponding couple of Pstates (i,j). Case of $i = j$ the threshold has no meaning since the system remains in the same Pstate, that explains the '-' entry in the figures.

3.3.5.1 Performance optimization

By promoting the mean response time costs. We observe in Fig. 3.6, that to optimize cost function:

- We should consider higher Pstates for the Pstate j, in particular $j = 6$. The system opts for high Pstates to reduce the response time component in the cost function. The best couple of Pstates to use is in the green column ($i \in \{1,\ldots,6\}$, $j = 6$).

- The blue integer in each colored square area,is the optimal threshold, it's a very small value. It shows that the Opteron processor switches quickly its cores to the higher Pstate 'j' , i.e. after having 1,2,3,4, or 5 jobs in the system depicted in the left heat-map, or after having 1 job in the system as shown in the right heat-map.

- The impact of Pstate 'i' is not relevant (in contrast to Pstate 'j'). Small thresholds here are usually exceeded. That explains the almost similar color in each column.

- When increasing $c1$ cost, to the value of 500, and still taking $c2 = 1$. We observe that the best threshold for all configurations is $th = 1$. Energy per job cost is irrelevant with regard to the response time's cost. Therefore, the

system behaves as there is no optimization for the energy per job. In that case, we have proved (as a consequence of Corollary 3.3.2) that response time function increases with the threshold.

3.3.5.2 Energy per job optimization

For the case of the enhancement of energy per job cost's. We observe in Fig. 3.6, that:

- The energy per job consumption increases with the value of Pstate 'i'. Therefore, the best couples of Pstates minimizing the objective function are in the green range ($i = 1, j \in \{2, \ldots, 6\}$). The system opts for low Pstates, in order to reduce the energy consumption.
- In contrast to the response time results, the optimal thresholds (blue integers in the colored squares) for the energy per job, achieves the highest levels (up to $th = \text{THMAX} = 100$). Notice that the aim is to minimize the energy per job, then it is consistent that the system switches 'lately' to the higher Pstate 'j' which consumes more. The system remains mostly all the time in Pstate 'i'. That explains the unnoticed modifications of cost function when changing the Pstate 'j'.

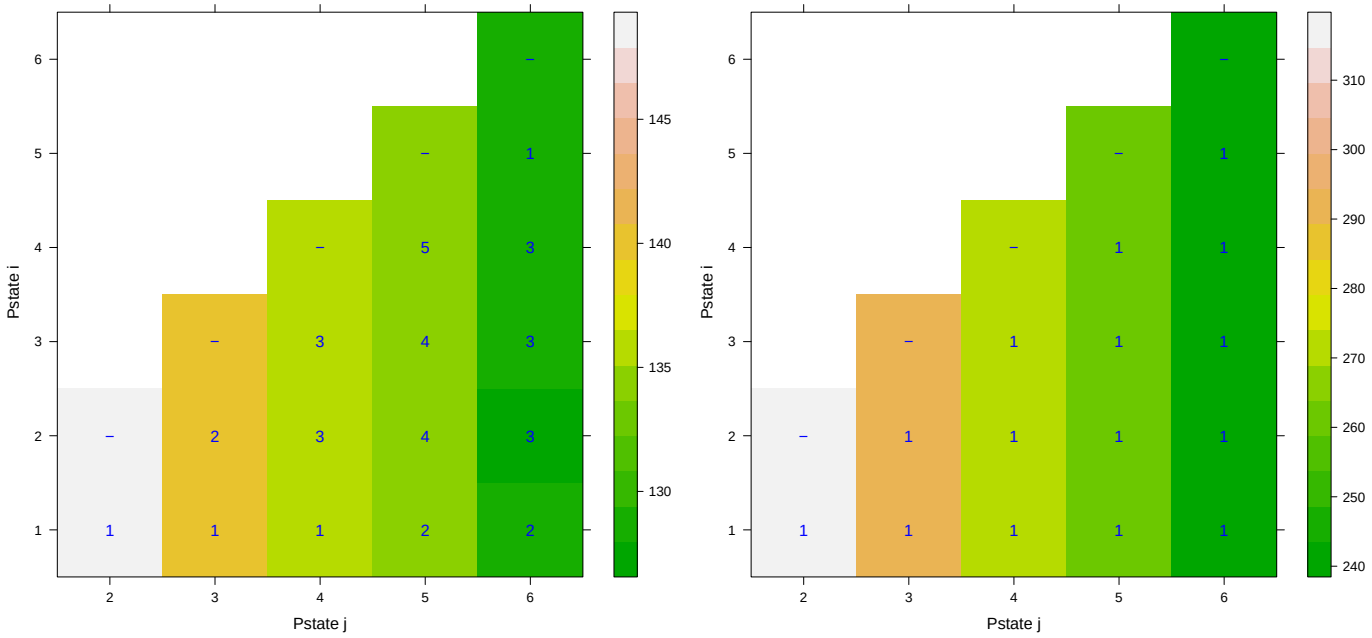


Figure 3.6: Objective function: $c1 = 200$, $c2 = 1$ (left) and $c1 = 500$, $c2 = 1$ (Right).
For instance, in the left heatmap, the optimal threshold for the entry $i = 2, j = 3$ is $th = 2$ with a total cost of $\Omega = 140.6$. In the right heatmap, for the same entry, the optimal threshold is $th = 1$ with a total cost of $\Omega = 290.4$.

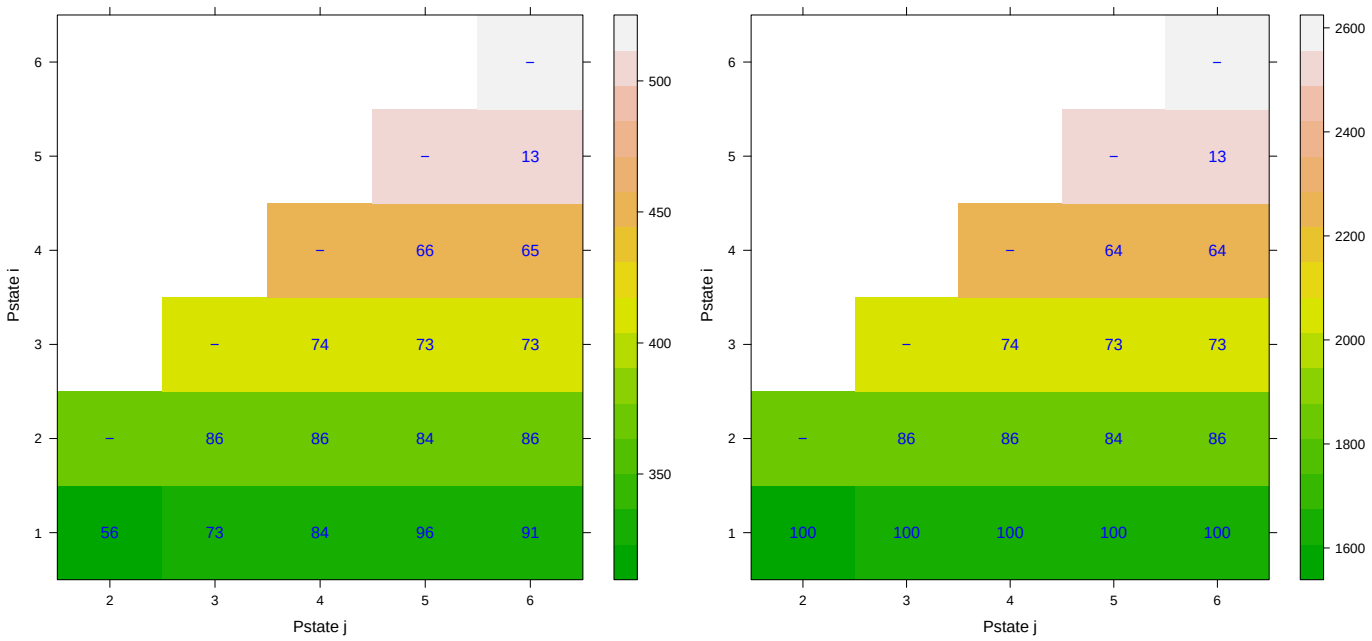


Figure 3.7: Objective function: $c1 = 1$, $c2 = 10$ (left) and $c1 = 1$, $c2 = 50$ (Right).
For instance, in left heatmap for the entry $i = 1, j = 4$. The optimal threshold is $th = 84$ with a total cost function of $\Omega = 324.2$, while in the right heatmap $th = 100$ and $\Omega = 1610.1$

3.4 Model 3: A multi-core processor with all Pstates

In this section, we also represent the Opteron processor by a birth-death process. We consider all ($N = 6$) Pstates of the Opteron processor. Servers rates (i.e. cores speed) depends on the number of tasks in the queue. Let $\Gamma = [th1, th2, \ldots, th_5]$ be the vector of the triggering thresholds (see Equation (3.33)). The existence of the steady-state distribution in this model is subject to the condition $\lambda < C\mu_6$. We suppose that all the systems we study are stable. We denote S_Γ such a system.

We first recall the 'el' comparison between vectors of thresholds.

Definition 3.4.1 *Let* $\Gamma 1$ *and* $\Gamma 2$ *be integer vectors of size* M, *then*

$$\Gamma 1 \leq_{el} \Gamma 2 \;\Rightarrow\; \Gamma 1(j) \leq \Gamma 2(j) \quad \forall j \in \{1, \ldots, M\}. \tag{3.32}$$

3.4.1 Closed form for the steady-state distribution

The processors contains C cores. Therefore $min(C, x)$ tasks are in service, and $(x - min\{C, x\})^+$ are queued. Under the classical assumptions we mention in Section 3.1, $\{X_\Gamma^t,\ t \geq 0\}$ is a Markov chain, and the transitions are as follows :

$$\begin{aligned} (x) &\rightarrow (x+1) \text{ with rate } \lambda, \\ (x) &\rightarrow (\max\{0, x-1\}), \text{ with rate } min\{x, C\} \cdot \mu(x). \end{aligned}$$

and service rate $\mu(x)$ is expressed as :

$$\mu(x) = \begin{cases} \mu_1 = \dfrac{f_1}{a} & if \ (x \leq th1), \\ \mu_2 = \dfrac{f_2}{a} & if \ (th1 < x \leq th2), \\ \mu_3 = \dfrac{f_3}{a} & if \ (th2 < x \leq th_3), \\ . & \\ . & \\ \mu_6 = \dfrac{f_6}{a} & if (x > th_5). \end{cases} \tag{3.33}$$

f_i is the frequency (GHz) given in Fig. 3.1.

Theorem 3.4.1 *The Markov chain is a birth-death process. Then, under stability condition (i.e.* $\lambda < C\mu_6$*) we can derive* $\Pi_\Gamma(x)$ *the steady-state probability.*
Let

$$\Psi(x) = \prod_{k=1}^{x} min\{k, C\}\mu(k),$$

and

$$R = max\{C, th_5\}, \tag{3.34}$$

then

$$\Pi_\Gamma(x) = \begin{cases} \frac{\lambda^x}{\Psi(x)}\Pi_\Gamma(0) & \forall \quad 0 < x \leq R, \\ \lambda^x \left[\Psi(R)(C\mu_6)^{x-R}\right]^{-1}\Pi_\Gamma(0) & \forall \quad x > R. \end{cases} \tag{3.35}$$

where

$$\Pi_\Gamma(0) = \left[1 + \frac{\lambda^{R+1}}{(C\mu_6 - \lambda)\Psi(R)} + \sum_{x=1}^{R} \frac{\lambda^x}{\Psi(x)}\right]^{-1}. \tag{3.36}$$

Proof. *In Equation (3.37), we present the classical birth-death equations for the steady-state distribution of that model.*

$$\Pi_\Gamma(x) = \begin{cases} \dfrac{\lambda}{min\{x, C\}\mu(x)}\Pi_\Gamma(x-1) & \forall \quad 0 < x \leq th_5, \\ \left(\dfrac{\lambda}{C\mu_6}\right)^{x-th_5}\Pi_\Gamma(th_5) & \forall \quad x > th_5. \end{cases} \tag{3.37}$$

The rest of the proof is similar to the one used in Theorem 3.3.1 for $N = 2$ *Pstates.*

3.4.2 Mean number of jobs and response time

Note that all Pstates are used and the system adapts the speed of its servers according to the number of pending jobs in the queue.

Corollary 3.4.1 *This corollary is the extension of Corollary 3.3.2 to* $N = 6$ *Pstates. The same approach is used. Let* $\{X^t_{\Gamma 1},\ t \geq 0\}$ *(resp.* $\{X^t_{\Gamma 2},\ t \geq 0\}$*) be the Markov chain of the stable system* $S_{\Gamma 1}$*(resp.* $S_{\Gamma 2}$*) with the thresholds vector* $\Gamma 1$ *(resp.* $\Gamma 2$*). Let* $\mathbb{E}[X_{\Gamma 1}]$, $\mathbb{E}[X_{\Gamma 2}]$ *(resp.* $\overline{T}_{\Gamma 1}, \overline{T}_{\Gamma 2}$*) be the mean number of tasks (resp. mean response time) in system* $S_{\Gamma 1}$ *resp.* $S_{\Gamma 2}$*. Therefore*

$$\Gamma 1 \leq_{el} \Gamma 2 \quad \Rightarrow \quad \begin{cases} \mathbb{E}[X_{\Gamma 1}] \leq \mathbb{E}[X_{\Gamma 2}] \\ \\ \overline{T}_{\Gamma 1} \leq \overline{T}_{\Gamma 2} \end{cases} \tag{3.38}$$

Proof. *Let* $\mu_x^{(S_{\Gamma 1})} = min\{x, C\}\mu(x)$ *(the same for* $\mu_x^{(S_{\Gamma 2})}$*) be the services rate transition generated at state* x *in system* $S_{\Gamma 1}$ *(resp.* $S_{\Gamma 2}$*) then*

$$\Gamma 1 \leq_{el} \Gamma 2 \ \Rightarrow \ \ \mu_x^{(S_{\Gamma 1})} \geq \mu_x^{(S_{\Gamma 2})} \ \ \forall\ state\ x,$$

also forall x, we have $\lambda_x^{(S_1)} = \lambda_x^{(S_2)} = \lambda$ *. From Theorem 3.2.1 we have*

$$\{X^t_{\Gamma 1},\ t \geq 0\} \leq_{st} \{X^t_{\Gamma 2},\ t \geq 0\}.$$

Using Lemma 3.2.1 for two stable systems, we get $\Pi_6^{\Gamma 1} \leq_{st} \Pi_6^{\Gamma 2}$*, therefore we conclude from Property 3.2.1 that* $\mathbb{E}[X_{\Gamma 1}] \leq \mathbb{E}[X_{\Gamma 2}]$ ***and*** $\overline{T}_{\Gamma 1} \leq \overline{T}_{\Gamma 2}$*.*

3.4.3 Power and energy consumption

Let $p_1, ..., p_6$ be the power consumption corresponding to Pstates $1, \ldots, 6$ (see Table3.1). Cores speed depends on the number of tasks affected to the Opteron processor. Let (x) be the number of tasks in the processor, then the power consumed by each core when hosting x tasks in the system is :

$$\begin{cases} p_1 & if\ (x \leq th1), \\ p_2 & if\ (th1 < x \leq th2), \\ p_3 & if\ (th2 < x \leq th_3), \\ . & \\ . & \\ p_6 & if (th_5 < x). \end{cases} \tag{3.39}$$

As in previous models. Let p_i $(resp.\ p_{i,Id})$ for $i \in \{1, \ldots, 6\}$ be the Opteron Pstate power (resp. Pstate idle power) used by a core in Pstate 'i' so $p_{i,Id} = \alpha p_i$.

Lemma 3.4.1 *Let* PW_Γ *be the total power consumption of a stable Opteron system. For the simplicity of the power consumption equation, we fix* $th_0 = -1$ *and* $th_6 = \infty$ *(effective thresholds are* $th1,\ th2,\ \ldots\ th_5$*). Let* $PW_\Gamma^{(a)}$ *(resp.* $PW_\Gamma^{(Id)}$*) be the mean power consumption of the servers in activity states (resp. Idle states). Let* $x \in [0, +\infty[$ *be the number of jobs in the system.* $\{X^t_\Gamma,\ t \geq 0\}$ *is an CTMC and, under stability constraint, admits a steady-state distribution* $\Pi_\Gamma(x)$*. Then, similarly to Lemma 3.2.2 we have :*

$$PW_\Gamma = PW_\Gamma^{(a)} + PW_\Gamma^{(Id)}\ , \tag{3.40}$$

where

$$\begin{cases} \mathrm{PW}_{\Gamma}^{(a)} = \sum_{s=0}^{5} \left[\sum_{x=th_s+1}^{th_{s+1}} \Pi_{\Gamma}(x) \Big(min(x, \mathrm{C}) p_{s+1} \Big) \right], \\ \mathrm{PW}_{\Gamma}^{(\mathrm{I}d)} = \sum_{s=0}^{5} \left[\sum_{x=th_s+1}^{th_{s+1}} \Pi_{\Gamma}(x) \Big(\mathrm{C} - min(x, \mathrm{C}) \Big) p_{s+1,\mathrm{I}d} \right]. \end{cases} \tag{3.41}$$

then

$$\mathrm{PW}_{\Gamma} = \sum_{s=0}^{5} \left[\sum_{x=th_s+1}^{th_{s+1}} \Pi_{\Gamma}(x) \Big(\mathrm{C} p_{s+1,\mathrm{I}d} + \Big(p_{s+1} - p_{s+1,\mathrm{I}d} \Big) min(x, \mathrm{C}) \Big) \right]. \tag{3.42}$$

Notice that Lemma 3.2.3 for energy consumption per job still hold for this model. Let $\mathrm{E}_{\Gamma}^{(Job)}$ be this energy, then

$$\mathrm{E}_{\Gamma}^{(Job)} = \frac{\mathrm{PW}_{\Gamma}}{\lambda}. \tag{3.43}$$

.

In next section, we do not proceed to develop Algorithm 1 for the case of $\mathrm{N} = 6$ Pstates.The number of systems to analyze can be up to $4 * 10^6$ (with $\mathrm{THMAX} = 100$). Note that in each system, we derive the steady-state distribution and the rewards (mean response time and energy per job). The steady state's calculation is, however, very fast with a complexity of $\mathrm{O(R)}$ (see Equation (3.34)). Hence, for numerical results in all Pstates Opteron system, we will show how the rewards evolves with thresholds vectors.

3.4.4 Numerical results

In the following, we present results of an Opteron processor with $\mathrm{C} = 20$ cores. This Opteron processor uses all the Pstates $\mathrm{N} = 6$ (as stated in the description section of this model). We decided to compare the results for two systems. The first system uses $\Gamma 1 = \{3, 6, 12, 24, 48\}$ as a distribution of thresholds. In that system thresholds range is growing exponentially, while in the second system, we consider a regular thresholds range $\Gamma 2 = \{10, 20, 30, 40, 50\}$.

In Fig. 3.8 we observe that, for all values of λ where the system is stable ($\lambda < \mathrm{C}\mu_6$), mean number of jobs and response time is lower in the system with $\Gamma 1$ thresholds. This result is justified in Corollary 3.4.1. In Fig. 3.9 it is noted that the mean power consumption and energy per job consumption is lower in the system with $\Gamma 2$ thresholds. This result can be seen as a consequence of the results in Fig. 3.8. Here $\Gamma 1 \leq_{el} \Gamma 2$, then it is apparent that the system with $\Gamma 1$ thresholds, would consume more power (and energy per job), since high Pstates are quickly reached. However, we can't yet, offer an analytical proof for this result.

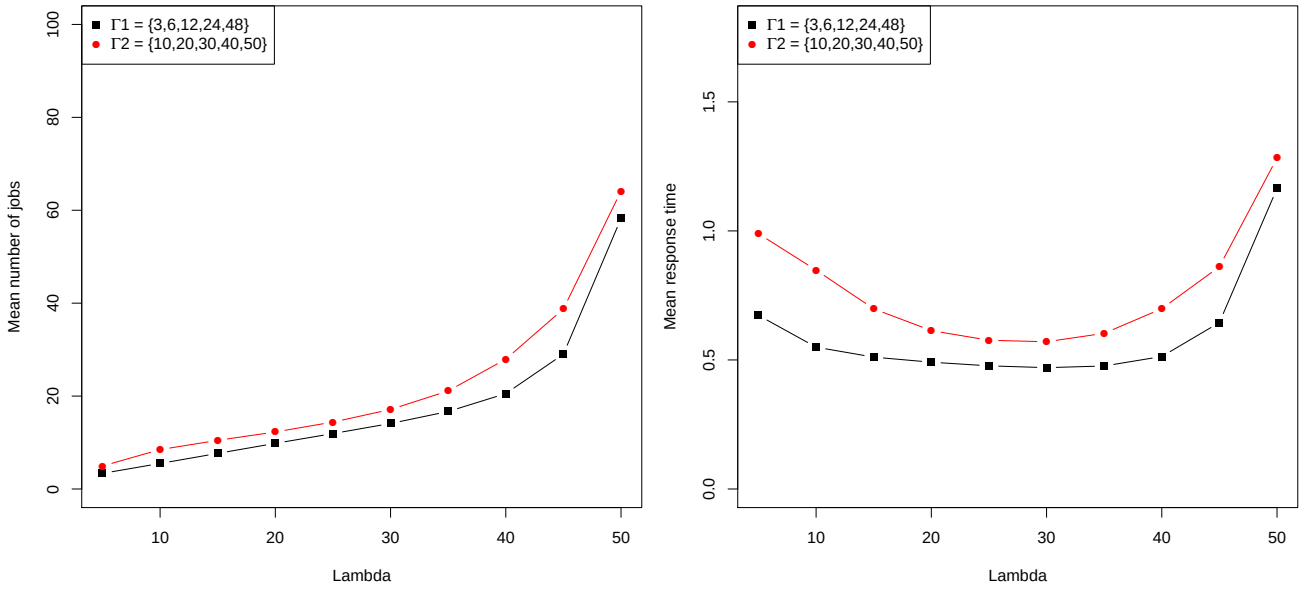


Figure 3.8: Mean number of jobs (left) and Mean response time (right).

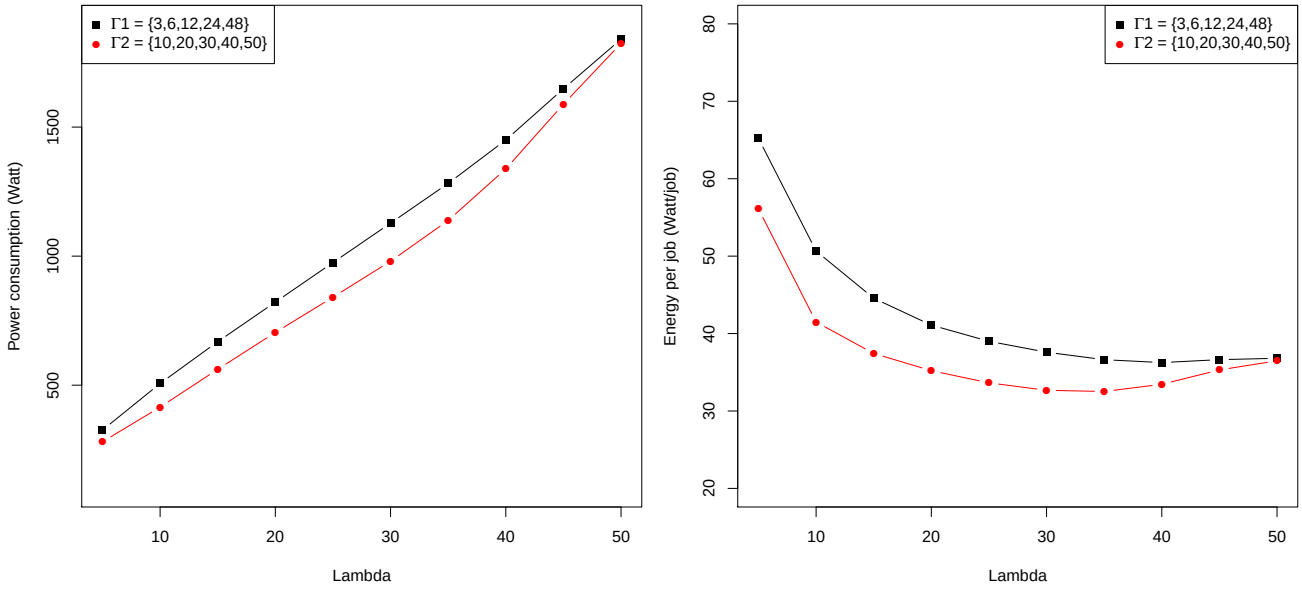


Figure 3.9: Mean power consumption (left) and Energy per job consumption (right)

Chapter 4

Performance and energy efficiency analysis in NGREEN optical network

Contents

4.1 Introduction

The fifth generation of mobile networks technology represents wireless communication systems related with the enormous growth of data traffic, due to the number of connected devices and the popularity of some applications as video streaming, augmented and virtual reality, cloud gaming, smart homes, connected cars, and remote control of machines [GK15]. These applications have strict constraints such as ultra-low latency, ultra-high bandwidth, to ensure the delivery of real-time services. These requirements can be met by efficiently integrating heterogeneous wireless and optical network segments and massive computing and storage services, delivered by means of cloud computing.

The Cloud Radio Access Network (CRAN) has been recently proposed [Ase+16]. Because of the stringent requirements (latency in the order of tens of milliseconds) of the several interfaces needed in C-RAN and the maturity and evolution of different optical network technologies, optical networks have been proposed to support interfaces between Remote Radio Head (RRH) and Base-Band Unit (BBU) (front-haul network), among BBUs and between BBUs and their peering point in the mobile core network (back-haul network).

The NGREEN network takes advantages of several improvements of optical technology [D C17; Chi17a; Chi17b; WC17]. First, for metro networks, it is now possible to synchronize entire networks up to nanoseconds. This enables architectures based on Optical Slot Switching (OSS) where packets circulate on specific time slots. Second, the use of WDM (Wavelength Division Multiplexing) packets allows some kind of optical parallelism. The introduction of parallelism in the optical processing has been considered as a powerful approach to reduce cost and energy consumption for optical systems.

The NGREEN project aims to design and validate a versatile network architecture with a scalable capacity, low cost and low energy consumption. For the performance point of view, we expect low latency and high utilization of the optical capacity. In OSS networks, the delay is due to the optical container construction and its insertion into the optical ring. Once the container has been emitted, it is never converted to electronic nor queued until its arrival to destination. The transportation time is only related to the distance (i.e. the length of the optical link) between source and destination.

In this work, we study two parts of the network. The first one, is the mechanism used to fill the optical container with the electronic packets (i.e. Internet Protocol (IP) or Ethernet) and the second one is the insertion node where the flows of optical containers are queued before being emitted on the ring. Here, we evaluate mathematically both mechanisms and the results from the filling process are used as an input parameter for the analysis of the insertion. Thus we do not need to make the typical assumption about Poisson arrivals of optical containers to the insertion node. The discrete distribution for the inter-arrivals of container is obtained through numerical analysis of the Markov chain and we found that it is quite different from a Poisson process.

The aggregation of several data units into optical container must have a deadline to avoid that some information wait too long during this first step. Adding a deadline

which is triggered when the container begins to fill will help to keep the delay shorter. However it makes the container filling incomplete because the Protocol Data Unit (PDU) moves to the insertion node when the deadline occurs. This movement can also be triggered if the filling is larger than a threshold. Thus we have two mechanisms to study the trade-off between latency and energy consumption. Indeed, the energy needed is the same irrespective of the number of Service Data Units (SDUs) carried by the PDU, the optical container has a fixed size and padding is added after the filling to complete the PDU. Thus a shorter deadline and a smaller threshold make the container less energy efficient and reduce the delay. For the insertion of the PDU into the ring we analyze both the opportunistic and reservation mode, in order to compare the delays. In [Gra+18], two mathematical models have been proposed to compare these two insertion modes, by considering Poisson arrivals for the PDU. In our work, we compute the distribution of inter-arrival time of the PDUs obtained from the filling process, in order to evaluate delays for the different insertion modes. We compute both insertion delays and end to end delays from SDU arrival to its depart from the optical ring.

This work is organized as follows. First, in section 4.2, we present the model of a container filling with stationary batch arrivals of service data units which arrive according to a stationary batch process. We derive a Discrete Time Markov Chain (DTMC in the following) to represent the remaining time and the number of service data units still present in the container. We prove that this chain satisfies a property already studied by Robertazzi in [Rob89]. All the directed cycles of the DTMC go through a single state. This property allows to compute the steady-state distribution very efficiently. We obtain the distribution of the PDU size when it is inserted on the ring and the distribution of the delay to fill a container and the delay between the release of two successive containers. Through these models we get the distribution of the delay between two successive arrivals of container at the insertion point on the optical ring which are used in Section 4.4 to perform extensive simulations of the ring. In section 4.3 we extend the model to a non stationary batch of arrivals. In section 4.5 We compare distributions of delays for both opportunistic and reservation insertion mode, and we study the trade off between energy efficiency and delays. At the end, we finish with some remarks.

4.2 Model for optical container filling

We first model the filling process (Fig. 4.1) to obtain the distribution of the number of bytes (aggregated in chunks) in an optical container (also denoted as PDU) and the distribution of time between two successive releases of PDU. We fill a container by aggregating various SDUs like Ethernet frames or Transmission Control Protocol (TCP) packets. We release a container either due to a deadline or a minimal occupancy.

4.2.1 Markov Chain model

We consider a discrete time model the time unit of which is the WDM slot duration. We assume that the arrivals follow a stationary batch process (we will change this

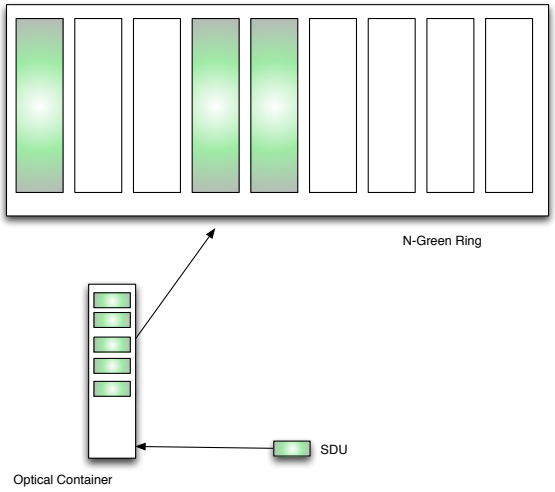


Figure 4.1: Container filling and insertion.

assumption in the next section), called A. We also assume independence between the successive batches of arrivals. The optical container size is 12500 bytes for the NGreen architecture [Chi17a; WC17] and the container size is associated with a slot equal to 10µs.

The arrivals may represent SDU or chunks with a constant number of bytes depending of the granularity needed by the model. For the example below, we assume that all the SDU (also equal to a chunk in this example) have the same size equal to 1500 bytes like an Ethernet MTU. Thus, the PDU can contain at most 8 chunks and we can depict the chain (see for instance Fig. 4.2). This assumption is only needed here for the sake of readability of the figure. Let A_n be the number of chunks which arrives at time n, $A_n =_{st} A$. The support of A is an upper bounded subspace of $\mathbb{N}$. We only assume that $0 \leq A_n \leq MaxArrival$. This value (i.e. MaxArrival) depends on the number of incoming lines connected to the station.

We design a discrete time Markov chain with two components: the occupancy of the container and the clock. The timer is initially equal to 0 and it jumps to 1 when the first SDU arrives in the buffer. It is then increased during each time unit until the buffer is released. Let H_n be the value of the timer and let X_n be the number of chunks in the buffer at time n. The system evolves based of the following events at time n:

- a batch of A_n chunks arrives and they are added into the buffer (i.e. A_n may be 0),
- the timer is increased if the number of Data Units in the container is positive,
- if the timer is equal to the deadline, or if the buffer is sufficiently filled, the buffer is released into an optical container to be inserted on the ring. Then, we begin the filling again.

Assume that chunks have a size equal to Y. Let Z be the size of the PDU. Let J be Z/Y be the number of chunks which can be included into the container. Let C be

the deadline and Thr be the utilization ratio of the buffer which triggers the release of the container.

$$\left[\begin{array}{ll} X_{n+1} = min(X_n + A_n, J) & \\ \text{IF } \ X_{n+1} > 0 & H_{n+1} = min(H_n + 1, C) \\ If \ (H_{n+1} = C) & Container\ Ready \\ If \ (X_{n+1} \geq Thr * J) & Container\ Ready \end{array}\right.$$

Once the "Container Ready" event occurs, we make the following transitions at the next step : $X_n = 0$ and $H_n = 0$.

Clearly, due to the independence and stationary assumption, (X_n, H_n) is a finite discrete time Markov chain the state space of which is included into $[0, J] \times [0, C]$. An example of such a chain for small parameters C and J and three possible batches of arrivals with size 0, 1 and 3 is displayed in Fig. 4.2. The Markov chain is built with the XBorne tool [Fou+16].

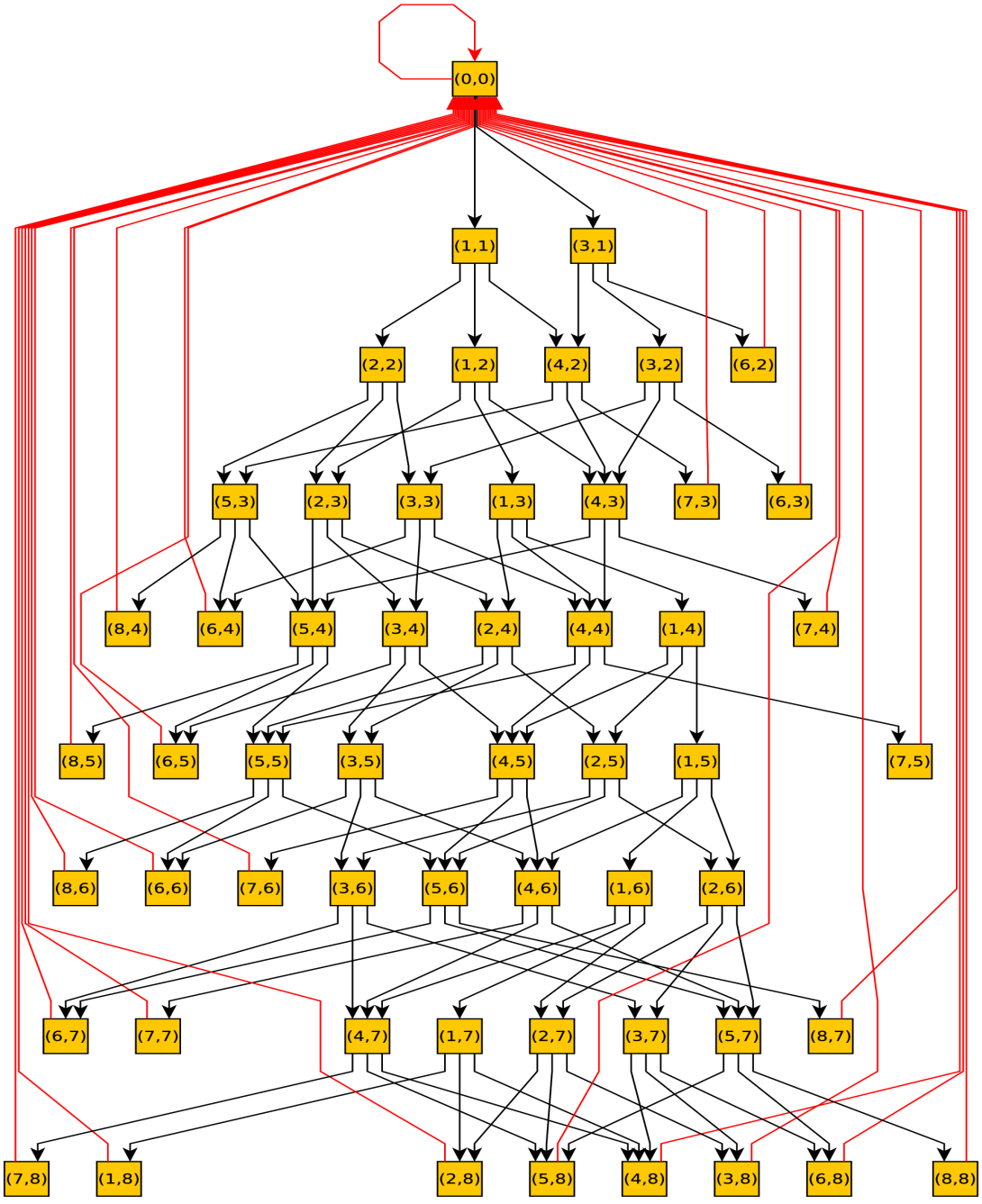


Figure 4.2: ToyModel: The Markov chain for J= 8, C= 8, and arrivals of 0, 1 or 3 SDU per slot.

4.2.2 Numerical Analysis

The numerical analysis of the chain takes into account its structural properties.

Property 4.2.1 *Assume that* $\mathrm{E}[\mathrm{A}_n] > 0$ *(positive arrival) and that* $\mathrm{J} * \mathrm{T}hr + \mathrm{M}ax\mathrm{A}rrival \leq \mathrm{J}$ *(no loss), the chain has the following properties:*

- *All the nodes in* $[0,\mathrm{J}] \times [0,\mathrm{C}]$ *are not reachable. Let* $\mathscr{S}$ *be the set of reachable states reachable from state* $(0,0)$.
- *The chain consists in many directed trees rooted at* $(0,0)$ *with back edges from the tree leaves returning to* $(0,0)$.
- *Therefore the matrix of the Markov chain (say* P*) can be decomposed into an upper diagonal matrix plus a matrix whose first column is positive and all the other entries are equal to* 0.

Proof. *To prove the first point, we just mention that state* $(0,2)$ *is not reachable from state* $(0,0)$ *because the clock jumps out of state* 0 *only with the first arrival. Let us now consider the graph of the Markov chain. Let* b *be a node of* $\mathscr{S}$*. Assume that* $b \neq (0,0)$*. By definition there is an oriented path from* 0 *to* b *in* $\mathscr{S}$*. During all the transitions out of* b*, the component* H_n *increases when* $\mathrm{X}_n > 0$ *and the component* X_n *does not decrease. Therefore, after some transitions, the components* H_n *is equal to* C *or the component* X_n *is larger than* $\mathrm{J} * \mathrm{T}hr$*. The next transition leads to state* $(0,0)$*. Therefore there is also a path from* b *to* $(0,0)$*. Thus, for every state* b *in* $\mathscr{S}$*, state* $(0,0)$ *is in at least one cycle going through* b*. Clearly, during all transitions (except the loop in state* $(0,0)$*), component* H_n *always increases. Therefore all the directed cycles of the chain are going through state* $(0,0)$*. The last point follows directly from the property on the cycles.*

Property 4.2.2 *If* $\mathrm{T}hr \leq \frac{\mathrm{J}+1-\mathrm{M}ax\mathrm{A}rrival}{\mathrm{J}}$*, no SDU are lost. The PDU is released before it is sufficiently filled.*

Indeed, this assumption implies that $\mathrm{J} * \mathrm{T}hr - 1 + \mathrm{M}ax\mathrm{A}rrival \leq \mathrm{J}$, and $\mathrm{J} * \mathrm{T}hr - 1$ is the largest buffer occupancy which does not trigger the emission of the container.

The first property allows that one can use a very efficient algorithm proposed by Robertazzi [Rob89] to compute the steady-state distribution (say $\pi()$).

Such an algorithm has a complexity which is linear in the number of non zero entries in matrix of the Markov chain while the Gaussian elimination for a full matrix has a complexity cubic in the number of states. Thus, solving the steady-state distribution only requires few seconds.

Once the steady-state distribution is obtained (typical results are represented in Fig. 4.3 as a heatmap to illustrate the bivariate nature of the chain), one can derive the distribution of the timer when the container is released. We have two conditions to release a container: the buffer occupancy X_n is larger than $\mathrm{J} * \mathrm{T}hr$ or the clock H_n is equal to C. Let $\mathscr{S}1$ be the set of states which satisfy one of these constraints. We first compute the probability of $\mathscr{S}1$ $Pr(\mathscr{S}1) = \sum_{(\mathrm{X},\mathrm{H}) \in \mathscr{S}1} \pi(\mathrm{X},\mathrm{H})$. Then, the distribution of the clock at a release instant are obtained after conditioning.

$$Pr_{\mathrm{H}}(t) = \frac{1}{Pr(\mathscr{S}1)} \sum_{(\mathrm{X},\mathrm{H}) \in \mathscr{S}1} Pr(\mathrm{X},\mathrm{H}) 1_{\mathrm{H}=t}$$

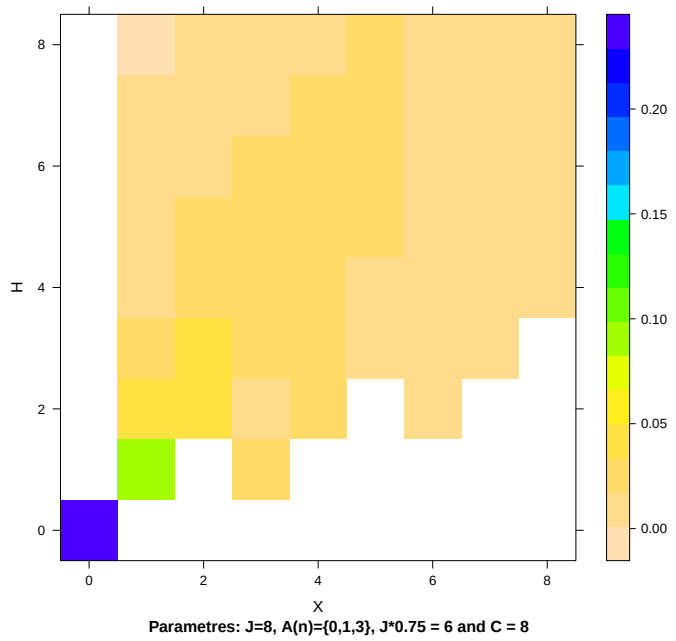


Figure 4.3: ToyModel: Distribution of the steady-state probability for (X,H) for the simple model. Non reachable states are depicted in white.

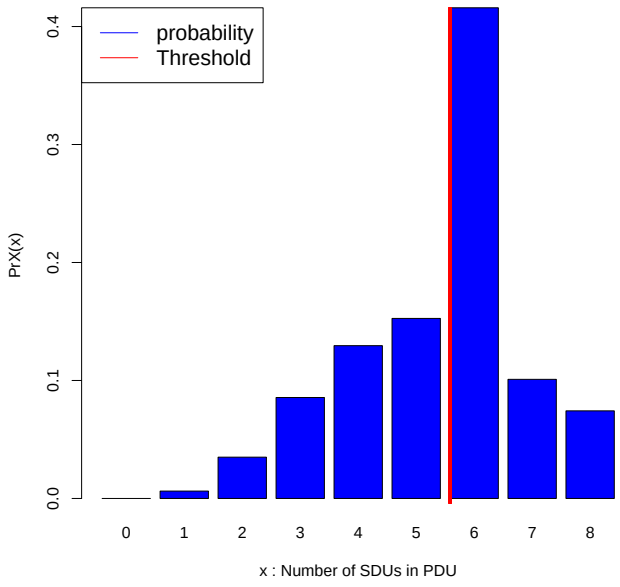


Figure 4.4: ToyModel: Distribution of the container size (in chunks of 1500 bytes) at release time.

Algorithm 2: Computation of the steady-state distribution: Robertazzi's algorithm

Input : Matrix P.
Output: Steady-state distribution $\pi(x, y)$.

Step 1) Initialize $\pi(0,0) = 1$.
Step 2) Compute all the values of $\pi(x, y)$ along the trees using the global balance Equation (in a tree, a node has only one predecessor, therefore solving the balance Equation is trivial).
Step 3) When the probability for all the leaves have been obtained, compute the sum of the probability S.
Step 4) Normalize: divide each probability by S to obtain a sum equal to 1.0.

Let $\mathscr{D}$ be this distribution of time needed for filling a PDU. Note that it has a bounded support: the upper bound is C while the lower bound is $\lfloor \mathrm{J} * \mathrm{T}hr/\mathrm{M}ax\mathrm{A}rrival \rfloor$.

One can also compute the distribution of the buffer size when the PDU is released.

$$\mathrm{P}r_{\mathrm{X}}(x) = \frac{1}{\mathrm{P}r(\mathscr{S}1)} \sum_{(\mathrm{X},\mathrm{H}) \in \mathscr{S}1} \mathrm{P}r(\mathrm{X},\mathrm{H}) 1_{\mathrm{X}=x}$$

See Fig. 4.4 for this distribution for the model depicted in Fig. 4.2.

Once we have obtained the distribution of the timer at a release instant, we have to add the time period when the buffer is empty. As the arrivals are independent and stationary, the duration of time between the release of the container and the arrival of the SDU has a geometric distribution with rate $(1 - \mathrm{P}r(\mathrm{A}_n = 0))$ and its support is N. Let $\mathscr{E}$ be this distribution of the empty period. The time between two successive releases of a container is the sum of the empty period and the duration to fill a container (i.e. $\mathscr{D}$) As the arrivals are independent, the distribution of the timer at release time and the distribution of the empty period for the buffer are also independent. Therefore the distribution of time between successive releases of a container is the convolution of $\mathscr{D}$ and $\mathscr{E}$. We compute a truncation of $\mathscr{F} = \mathscr{D} \otimes \mathscr{E}$. Note that we must truncate the distribution because the support of $\mathscr{E}$ is not upper bounded. This distribution $\mathscr{F}$ can be used an input parameter of the simulation of NGREEN optical ring.

In Fig. 4.4, Fig. 4.5 and Fig. 4.6, we have depicted typical results for these distributions. These results were obtained for arrivals of Ethernet Maximum Transmission Units (MTUs) with full size. Therefore, the container size is 8. We have consider a deadline equal to 8 and the threshold ratio for occupancy is 75% leading to an integer threshold equal to 6. We have assumed that the batch of arrivals may contain between 0, 1 or 3 SDU with respective probability: 0.5, 0.4, and 0.1. Thus the expected number of SDU per time slot is 0.7.

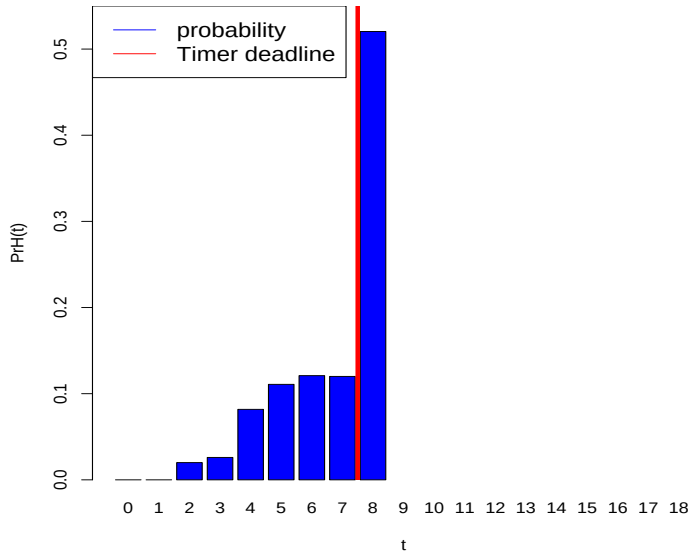


Figure 4.5: ToyModel: Distribution of the Timer at release time.

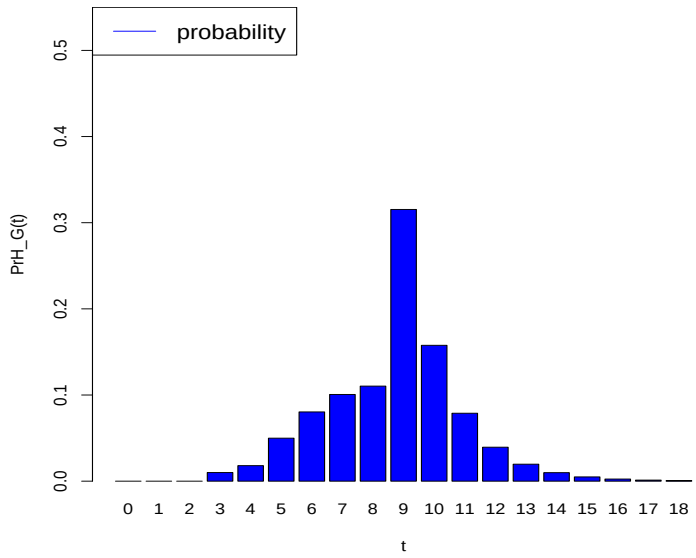


Figure 4.6: ToyModel: Distribution of inter-PDU release time.

4.2.3 A more realistic example with Ethernet and TCP SDUs

We model the arrivals of several types of SDU (short SDU of 50 bytes like TCP acknowledgments, and larger SDU of 1500 bytes like Ethernet frames). We also consider some others SDUs which will describe in the following. As the gcd of 50 and 1500 is 50 we describe the buffer occupancy by chunks of 50 bytes. TCP acks consist in one chunk while a Ethernet MTU is modeled by 30 chunks. The buffer occupancy varies from 0 to $J = 250$ chunks. To keep this representation of the buffer, we assume that the sizes of the other SDUs are also a multiple of 50 bytes.

4.2.3.1 Model 1

In that model we consider three possible batches : 1 for the TCP packets, 30 for the Ethernet frames, and 0 for no arrivals. Their respective probabilities are assumed to be 0.4, 0.2 and 0.4.

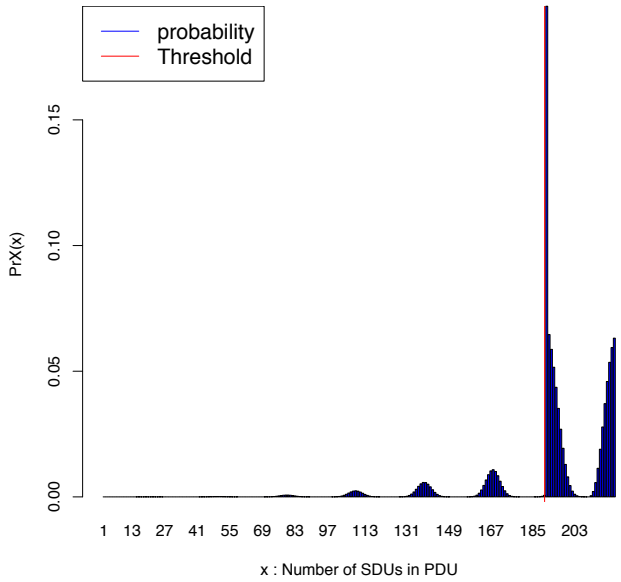


Figure 4.7: Model1: Distribution of the PDU size at release time (in chunks of 50 bytes)

Thus the average number of bytes arriving per slot is 320 (i.e. 6.4 chunks of 50 bytes). We first chose a deadline equal to 40 and a threshold ratio equal to 75% leading to an integer threshold equal to 190. We obtain a Markov chain with 5200 states and roughly 15000 transitions, the steady-state distribution of which is obtained after $0.01s$ of computation time. The PDU contains an average payload of 971 bytes (i.e. 194.37 chunks of 50 bytes obtained from the expectation of the distribution depicted in Fig. 4.7). The average time to fill a container is 29.7 slots (see the distribution in Fig. 4.8), and the average inter-arrival time 31.3 slots (see the distribution depicted in Fig. 4.9).

We now study the effect of the deadline and the threshold ratio. In Fig. 4.10, we present the mean time to fill a container for various threshold ratios $(0.60, 0.70, 0.80, 0.90)$.

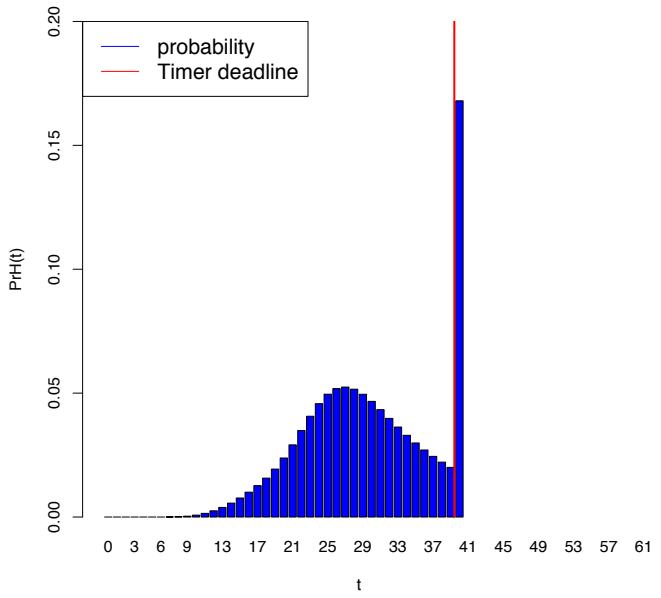


Figure 4.8: Model1: Distribution of the Timer at release time.

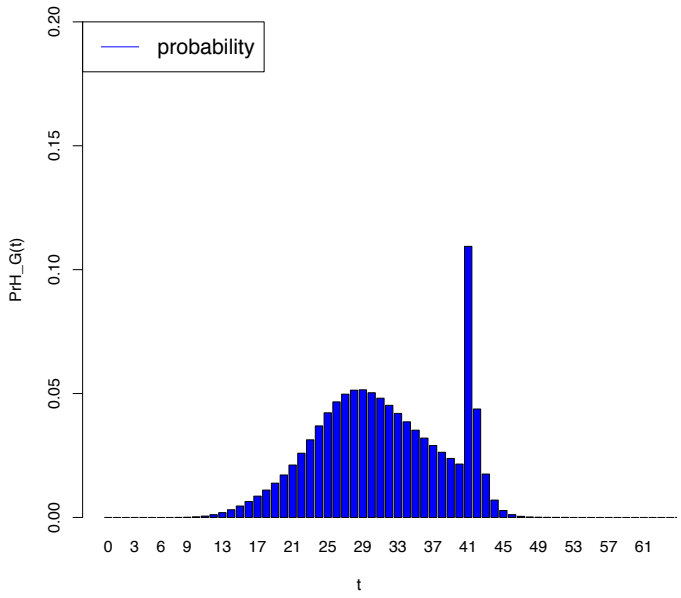


Figure 4.9: Model1: Distribution of inter-PDU release time.

The analysis shows that the deadline is useful when it is small. After a boundary value depicted by small dots in Fig. 4.10, the deadline has a weaker impact on the time needed to fill the containers. They are mostly released because they have the minimal size required. Obviously, the mean time to fill a container increases with the threshold ratio.

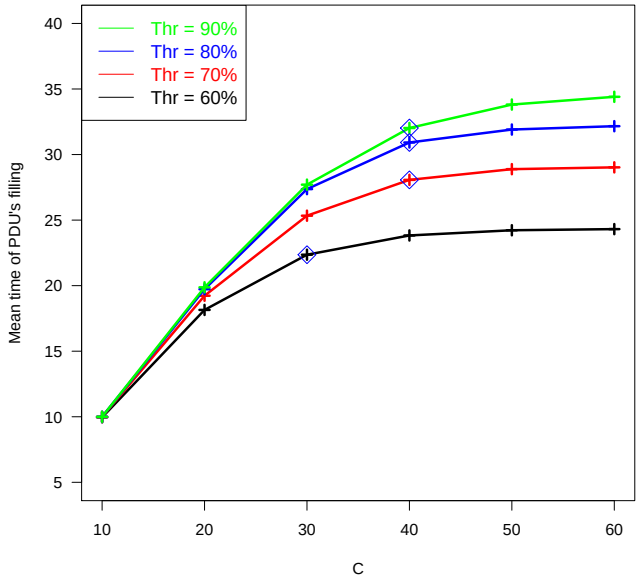


Figure 4.10: Model1: Distribution of the Timer at release time VS deadline, for different threshold ratios

We will use this model for the insertion of PDUs by the stations in the optical ring (see Section 4.4).

4.2.3.2 Model 2

In this model we consider five possible batches : 1 for the TCP packets, 30 for the Ethernet frames, 4 and 9 chunks for various SDUs and 0 for no arrivals. Their respective probabilities are 0.2, 0.15, 0.2, 0.1 and 0.35.

Thus, the average number of bytes arriving per slot is 320 (i.e. 6.4 chunks of 50 bytes). The threshold ratio and the deadline are the same as in the last model. Clearly this model will generate a bigger Markov chain since there is more batches of arrivals. We obtain a Markov chain with 7670 states and more than 33500 transitions, the steady-state distribution of which is obtained after $0.03s$ of computation time. Clearly our approach is extremely efficient.

The container contains an average payload of 980.5 bytes (i.e. 196.1 chunks of 50 bytes for the expectation of the distribution depicted in Fig. 4.11). The average time to fill a container is 30.1 slots (see the distribution in Fig. 4.12). and the average inter-arrival time of containers is 31.6 slots (the distribution is depicted in

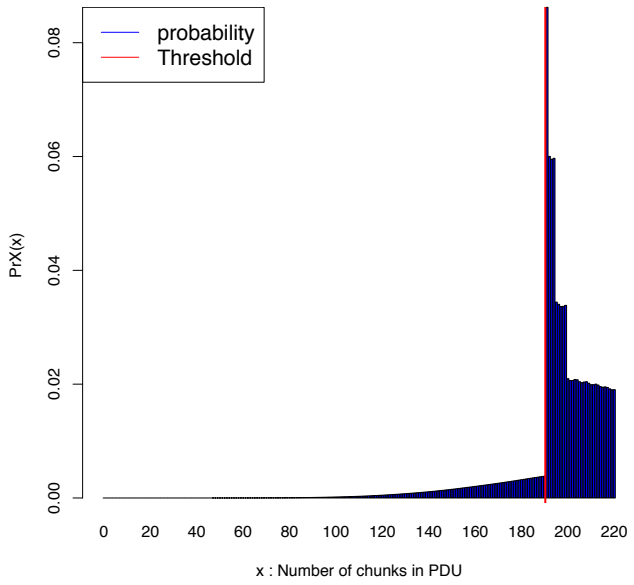


Figure 4.11: Model2: Distribution of the PDU size at release time (in chunks of 50 bytes).

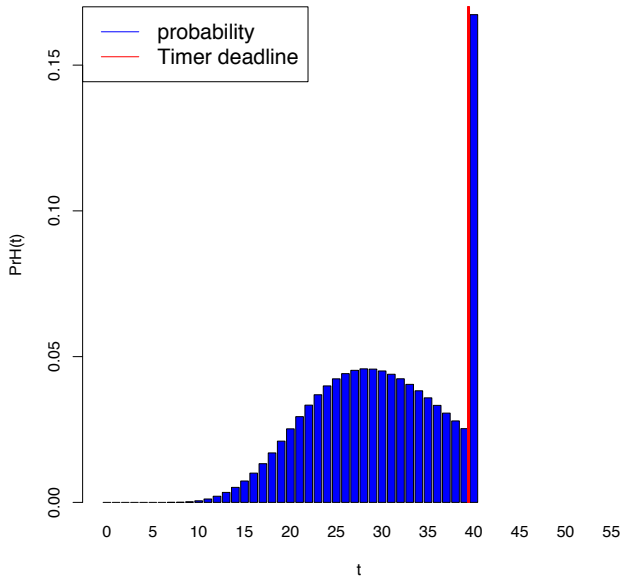


Figure 4.12: Model2: Distribution of the Timer at release time.

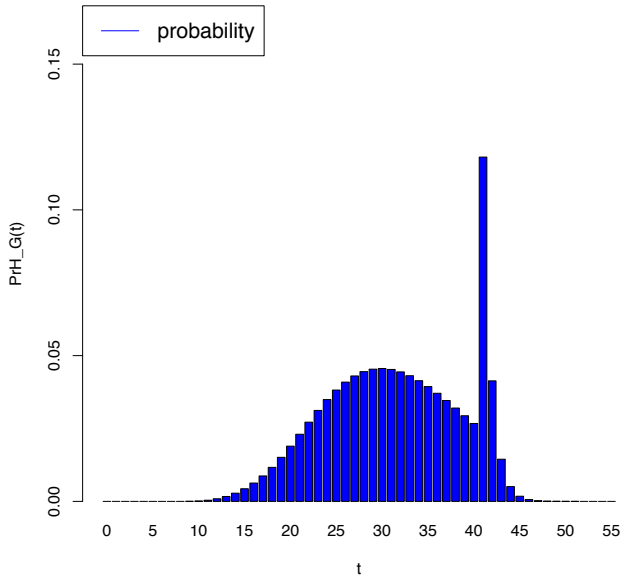


Figure 4.13: Model2: Distribution of inter-PDU release time.

Fig. 4.13). We will use this model in the next section where we generalize the resolution to wider non stationary arrivals.

4.3 Generalization to non stationary arrivals

We now assume that the arrival process is not stationary. More precisely we consider a Switched Bernoulli Batch Process (SBBP): the size of the batch is modulated by an auxiliary discrete time finite Markov chain [HTS91]. Let $\{\Phi_n\}$ be the modulating Markov chain, N be its number of states, M its transition matrix and ψ its transition function [Duf97]. When this chain is in state k, the batch of arrivals is distributed following distribution $B^{\{k\}}$. The arrivals are still independent but the distribution depends on phase Φ_n at time n. Clearly, using the same notation as in previous section, (Φ_n, X_n, H_n) is a DTMC whose transition is described by:

$$\left[\begin{array}{ll} \Phi_n = \psi(U, \Phi_{n-1}) & \\ X_{n+1} = min(X_n + A_n, J) & with \;\; A_n =_{st} B^{\{\Phi_{n-1}\}} \\ \text{IF} \;\; X_{n+1} > 0 & H_{n+1} = min(H_n + 1, C) \\ If \;\; (H_{n+1} = C) & Container\ Ready \\ If \;\; (X_{n+1} \geq Thr * J) & Container\ Ready \end{array}\right.$$

where U is uniform random. Again, once the "Container Ready" event occurs, we make the following transitions at the next step : $X_n = 0$ and $H_n = 0$ but the value of Φ does not change.

Due to these assumptions we can reorder the Markov chain of the model to exhibit a block structure:

$$\mathbf{P} = \begin{pmatrix} P_{11} & P_{12} & \dots & P_{1N} \\ P_{21} & P_{22} & \dots & P_{2N} \\ \vdots & \vdots & \vdots & \ddots \\ P_{N1} & P_{N2} & \dots & P_{NN} \end{pmatrix}, \tag{4.1}$$

where block $P_{i,i}$ describes the transitions when modulating chain Φ remains in state i while $P_{i,j}$ models the transitions when modulating chain Φ moves from state i to state j. Matrix $P_{i,j}$ has size $n_i \times n_j$. We have for all value i and j of Φ, and all value k and l of (X, H):

$$P_{i,j}[k,l] = M[i,j] Q_i[k,l], \tag{4.2}$$

where $Q_i[k,l]$ describes the transition from state $k = (X_{n-1}, H_{n-1})$ to $l = (X_n, H_n)$ when the phase is i at the beginning of the transition. Thus the Markov chain can be described as a generalized tensor product and some algorithms take advantage of this block decomposition [FPS98; GDF01; Ben+04].

Remember that the NGreen container size is associated with a slot equal to 10µs. Thus, if the evolution of the phase is much slower, we also have that (i.e $M(i,j) << M(i,i)$, for all $i \neq j$) and the Markov chain is Near Completely Decomposable (NCD).

Definition 4.3.1 P *is NCD if* P *is decomposed as in Equation* (4.1) *such that for all* $i \neq j$, $||P_{i,j}||_\infty << 1$.

Proposition 4.3.1 *If* $M(i,j) << M(i,i)$, *for all* $i \neq j$, *the DTMC* (Φ_n, X_n, H_n) *is NCD.*

This is a simple consequence of the multiplicative form in Equation (4.2).

There exist many algorithms which takes into account the block structure or the NCD property to compute the steady-state distribution [Ste95] after a decompositional analysis and the coupling of the sub-problems. Elimination algorithms are rather inefficient due to the complexity and the spectral properties of NCD matrices forbid to use iterative algorithms like Gauss Seidel or the Power method. We first use a decompositional method (more precisely Iterative Aggregation/Disaggregation algorithm) because our matrix exhibits a block structure decomposition: Koury, McAllister and Stewart (KMS). More precisely we begin with the KMS-BGS version described in [Ste95] (see also Algorithm 3) where a Block Gauss Seidel (BGS) is performed at each iteration (Step 6) in Algorithm 3. The steady-state distribution π is decomposed into sub-vectors π_i. Sub-vector π_i has size n_i and $n = \sum_{i=1}^{N} n_i$ is the size of the Markov chain of the model. All vectors in the following are row vectors. e is a row vector whose entries are all equal to 1 and e^T is the transpose of e.

Algorithm 3: Computation of the steady-state distribution: KMS-BGS algorithm

---

Input : Matrix P.

Output: Steady-state distribution $\pi(x, y)$.

Step 1) Choose an initial probability distribution $\pi_i^{(0)}$ for each sub-problem i, typically, the uniform distribution: $\pi^{(0)}[m] = 1/n$ for all m. Decompose $\pi^{(0)}$ into sub-vectors $\pi_i^{(0)}$.

Step 2) Compute the conditional probabilities at iteration k knowing that we are in subset i :

$$\Phi_i^{(k)} = \frac{\pi_i^{(k)}}{||\pi_i^{(k)}||_1}.$$

Step 3) Compute coupling matrix $A^{(k)}$: $A^{(k)}[i, j] = \Phi_i^{(k)} P_{i,j} e^T$.

Step 4) Compute the dominant eigenvector of matrix $A^{(k)}$: $\psi^{(k)} = \psi^{(k)} A^{(k)}$, and $\psi^{(k)} e^T = 1$.

Step 5) Obtain a new approximation of the solution based on the decomposition and the law of total probability:

$$z^{(k)} = (\psi^{(k)}[1]\Phi_1^{(k)}, \psi^{(k)}[2]\Phi_2^{(k)}, \ldots, \psi^{(k)}[N]\Phi_N^{(k)}).$$

Step 6) Perform an iteration of block Gauss Seidel to find a better iteration for $\pi_i^{(k+1)}$

$$\pi_i^{(k+1)}(Id - P_{i,i}) = \sum_{j<i} \pi_j^{(k+1)} P_{j,i} + \sum_{j>i} z_j^{(k)} P_{j,i}.$$

Step 7) Conduct a test for convergence between $\pi_i^{(k)}$ and $\pi_i^{(k-1)}$. If the accuracy is sufficient, then build solution π from sub-vectors $\pi_i^{(k)}$. Otherwise set $k = k + 1$ and go to step 2).

Unfortunately, Step 6) requires to compute exactly (or to give an approximation)

of $(\mathrm{I}d - \mathrm{P}_{i,i})^{-1}$ to perform the block Gauss Seidel iteration. Even if this matrix has to be computed once for all the iterations of the algorithm, it needs a large amount of time. As $\mathrm{P}_{i,i}$ is transient, $(\mathrm{I}d - \mathrm{P}_{i,i})^{-1}$ is the fundamental matrix associated to $\mathrm{P}_{i,i}$. The computation of the fundamental matrix is known to be numerically unstable and we use an elimination algorithm proposed by Sheskin [She95] due to its stability. Such an algorithm is also time consuming and we also use a version of KMS where the block Gauss Seidel iteration is replaced by an iteration of the Power method. This version (called KMS-Pwr) is known to need a larger number of iterations but an iteration of the Power method does not require to compute $(\mathrm{I}d - \mathrm{P}_{i,i})^{-1}$ and is therefore much faster. In KMS-Pwr algorithm, Step 6) is replaced by

$$\pi_i^{(k+1)} = \sum_j z_j^{(k)} \mathrm{P}_{j,i}.$$

Computing the steady-state distribution is really time consuming as it can be seen in Table 3.1 for KMS-BGS while the numbers of iterations increase with KMS-Pwr. Thus we develop a new algorithm which combines the mains ideas of KMS algorithm (decomposition and coupling) and Robertazzi method (simple elimination due to the graph structure). In this new algorithm (called KMS+R) we replace Step 6) of KMS algorithm and we obtain a much more efficient method as it is based on the cycles of the graph.

4.3.1 Replacement of Step 6) in our algorithm

As we state the complexity of this new algorithm, we need some notation and some assumptions about the storage of matrix P.

- Matrix P is stored by column.
- Matrix P is decomposed into blocks called $\mathrm{P}_{i,j}$ for $i = 1..\mathrm{N}$, and $j = 1..\mathrm{N}$. Diagonal block i (i.e. $\mathrm{P}_{i,i}$) contains n_i nodes and m_i non zero elements.
- Furthermore $n = \sum_i n_i$. Let m the number of non zero transitions in P.
- π_i^k represents the value of sub-vector π for subset i at iteration k. It is a vector of size n_i.
- Diagonal block $\mathrm{P}_{i,i}$ has the following structure (see previous results): $\mathrm{P}_{i,i} = \mathrm{C}_i + \mathrm{U}_i$, where:
  - C_i is a matrix whose first column is positive and all the other entries are equal to 0.
  - U_i is a strictly upper diagonal matrix.
  - $\mathrm{P}_{i,i}[1,1] = 0$.

Instead of computing matrices $(\mathrm{I} - \mathrm{P}_{i,i})^{-1}$ for each index i, we solve the following system of linear equations, at each iteration k and for each index i, taking into account the properties of the matrices.

$$\pi_i^{(k+1)}(\mathrm{I}d - \mathrm{P}_{i,i}) = \sum_{j<i} \pi_j^{(k+1)} \mathrm{P}_{j,i} + \sum_{j>i} z_j^{(k)} \mathrm{P}_{j,i}.$$

Note that the right part can be computed before solving the linear system. Let $b_i^{(k)}$ be the right hand side. The Equation becomes: $\pi_i^{(k)} = \pi_i^{(k)} \mathrm{P}_{i,i} + b_i^{(k)}$. Using the decomposition, we get:

$$\pi_i^{(k)} = \pi_i^{(k)} (\mathrm{C}_i + \mathrm{U}_i) + b_i^{(k)} \tag{4.3}$$

Taking into account the structure, we get for $p = 2$ to n_i

$$\pi_i^{(k)}[p] = \sum_{q<p} \pi_i^{(k)}[q] \mathrm{U}_i[q,p] + b_i^{(k)}[p]$$

and

$$\pi_i^{(k)}[1] = \sum_q \pi_i^{(k)}[q] \mathrm{C}_i[q,1] + b_i^{(k)}[1]$$

We prove in the following lemma a constructive formula for $\pi_i^{(k)}[q]$ for all index i and node q.

Lemma 4.3.1 *For all $q > 1$ and i in $1..b$, we have:*

$$\pi_i^{(k)}[q] = \alpha_i[q] \pi_i^{(k)}[1] + \beta_i^{(k)}[q] \tag{4.4}$$

where $\alpha_i[q]$ and $\beta_i[q]$ are defined by the following induction:

$$\alpha_i[q] = \sum_{p<q} \alpha_i[p] \mathrm{P}_{i,i}[p,q], \tag{4.5}$$

with the following initializations: $\alpha_i[2] = \mathrm{P}_{i,i}[1,2]$, and $\alpha_i[1] = 1$. Moreover

$$\beta_i^{(k)}[q] = b_i^{(k)}[q] + \sum_{p<q} \beta_i^{(k)}[p] \mathrm{P}_{i,i}[p,q], \tag{4.6}$$

with $\beta_i^{(k)}[2] = b_i^{(k)}[2]$, and $\beta_i^{(k)}[1] = 0$. Furthermore, we have

$$\pi_i^{(k)}[1](1 - \sum_{p>1} \alpha_i[p] \mathrm{P}_{i,i}[p,1]) = b_i^{(k)}[1] + \sum_{p>1} \beta_i^{(k)}[p] \mathrm{P}_{i,i}[p,1] \tag{4.7}$$

Note that vector α_i does not change with iteration number k unlike vector $\beta_i^{(k)}$. Thus it can be computed once.

Proof: First, we prove the induction.

1. For $p = 2$, from Equation (4.3), we clearly have: $\pi_i^{(k)}[2] = \pi_i^{(k)}[1] \mathrm{P}_i[1,2] + b_i^{(k)}[2]$, because node 1 if the only predecessor of node 2. We clearly get the values for $\alpha_i[2]$ and $\beta_i[2]$.

2. For an arbitrary $p > 2$, assume the induction holds until $p-1$. We have:

$$\pi_i^{(k)}[p] = \sum_{q<p} \pi_i^{(k)}[q] \mathrm{P}_{i,i}[q,p] + b_i^{(k)}[p].$$

Using the induction for $\pi_i^{(k)}[q]$ with $q < p$, we get:

$$\pi_i^{(k)}[p] = \sum_{q<p} (\alpha_i[q] \pi_i^{(k)}[1] + \beta_i^{(k)}[q]) \mathrm{P}_{i,i}[q,p] + b_i^{(k)}[p],$$

from which one can readily obtained the induction.

Finally, we compute $\pi_i^{(k)}[1]$ using Equation (4.3):

$$\pi_i^k[1] = b_i^{(k)}[1] + \sum_{p>1} \beta_i^{(k)}[p] P_{i,i}[p,1] + \sum_{p>1} \alpha_i[p] P_{i,i}[p,1])$$

This concludes the proof and we can now derive the algorithm (called KMS+R as it mixed the KMS approach of block structured matrices with Robertazzi algorithm).

Algorithm 4: Replacement of step 6) for KMS+R algorithm

Input : KMS+R algorithm.
Output: Steady-state distribution $\pi(x, y)$.

Step 6.1) Compute the terms of the right hand side to obtain b_i^k.
Step 6.2) Using Equations 4.5 and 4.6, we get the values of $\alpha_i[p]$ and $\beta_i^{(k)}[p]$ for $p > 1$.
Step 6.3) Using Equation (4.7), we get the value of $\pi_i^{(k)}[1]$.
Step 6.4) We obtain $\pi_i^{(k)}[p]$ for all $p > 1$ using the last two steps and the induction.

Let turn to the complexity of this approach for replacing Step 6). The computation of b_i^k requires at most m products and m additions to obtain the values of b_i^k for all i (it is a block Gauss Seidel iteration) during Step 6.1). At Step 6.2) the loop iterates on p from 2 to n_i and iteration p has a complexity equal to two times the number of non zero elements in column p of matrix $P_{i,i}$. Step 6.3) requires multiplication by the first column of $P_{i,i}$ (twice). Finally for Step 6.4) we only need n_i multiplications and n_i additions. Steps 6.2 and 6.3 requires $o(m_i)$ operations while Step 6.4) needs $o(n_i)$. This is much more efficient that our version of KMS-BGS which computes the fundamental matrix of an arbitrary transient Markov chain (i.e. $(I - P_{i,i})^{-1}$) [She95] which has a complexity in $o(n_i^3)$. Note however that we only need to compute $(I - P_{i,i})^{-1}$ once.

4.3.2 Algorithm Comparison

We now compare the time needed to solve the model using a standard numerical algorithm with a cubic complexity (GTH proposed by Grassmann, Taksar and Heymann) which does not take into account the structure of the matrix, and with two versions of the algorithm for block structured chains Markov chains (including NCD chains). GTH is a very accurate direct method. A detailed presentation of this algorithm can be found in [Ste95].

KMS-BGS, KMS-Pwr and KMS+R algorithms are iterative. To stop the process we use the same condition as Stewart [Ste95] (i.e. $\|\pi - \pi P\|_2 \leq \epsilon$ with a precision ϵ equal to 10^{-15}). We check the algorithms for NCD and non NCD matrices associated with the same model for the buffer and various modulating Markov chains for the phase of the arrivals. We assume without loss of generality that the modulating Markov chain has only two phases. We check our algorithm for NCD and non NCD matrices.

4.3.2.1 NCD matrix

We have assumed the following matrix for modulation of the arrival:

$$M = \begin{pmatrix} 0.9 & 0.1 \\ 0.001 & 0.999 \end{pmatrix} \tag{4.8}$$

Clearly, such a matrix makes the Markov chain of the model near completely decomposable. The batches of arrival have the same support (with size equal to 4) but different probabilities for the two phases (see Table 4.1).

Batch size	0	1	2	3
Probability Phase 1	0.1	0.15	0.25	0.50
Probability Phase 2	0.5	0.25	0.15	0.1

Table 4.1: Parameters for the batch distribution.

We present in the following table a comparison of the execution time of the four algorithms. The computations have been performed on a multicore PC with 8 Xeon processors at 2.4 GHz with 12Go RAM. We report the results in Table 4.2. The computations are performed 30 times to obtain the 95% confidence intervals for the computation times. We only report the CPU usage and we do not take into account the IO. The numbers of iterations needed by the KMS-BGS and KMS+R algorithms are extremely low (4 to 6 iterations on most of the examples we solved) when the matrix is NCD, as already reported in [Ste95]. KMS-Pwr is less efficient for this criterion.

Size	**1000**		**1500**	
KMS+R	[0.018, 0.020]s	5	[0.019, 0.022] s	5
KMS-Pwr	[0.056 , 0.062]s	18	[0.076 , 0.079]s	18
KMS-BGS	[20.21, 20.58]s	5	[62.96, 63.90]s	5
GTH	[4.07, 4.39]s		[13.41, 13.78]s	
Size	**2000**		**5500**	
KMS+R	[0.039, 0.057]s	5	[0.15, 0.16]s	6
KMS-Pwr	[0.12, 0.15]s	17	[0.37 , 0.38]s	16
KMS-BGS	[154.68, 155.25]s	5	[2956.70, 2965.16]s	6
GTH	[33.34, 33.72]s		[629,67, 632.32]s	

Table 4.2: Computation time in seconds and number of iterations for NCD chains.

4.3.2.2 General modulating matrix

We now consider two modulating matrices which make the matrix non NCD.

$$M1 = \begin{pmatrix} 0.8 & 0.2 \\ 0.1 & 0.9 \end{pmatrix}, \quad M2 = \begin{pmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{pmatrix}$$

Our approach is again very fast as it could be seen in Table 4.3 and Table 4.4. The numbers of iterations increase for general matrices compared to NCD matrices.

Size	**1000**		**1500**	
KMS+R	[0.049, 0.054]s	18	[0.082, 0.091]s	21
KMS-Pwr	[0.074 , 0.081]s	32	[0.190 , 0.194]s	36
KMS-BGS	[20.24, 20.62]s	18	[63.00, 63.99]s	21
GTH	[4.04,4.35]s		[13.31,13.74]s	
Size	**2000**		**5500**	
KMS+R	[0.23, 0.29]s	25	[0.89, 0.91]s	41
KMS-Pwr	[0.29 , 0.30]s	41	[1.38 , 1.41]s	64
KMS-BGS	[155.11, 155.75]s	25	[2975.93, 2985.47]s	41
GTH	[33.42, 33.84]s		[630.99, 634.26]s	

Table 4.3: Computation time in seconds and number of iterations for modulating matrix M1.

	1000		**1500**	
KMS+R	[0.23, 0.25]s	66	[0.50, 0.51]s	88
KMS-Pwr	[0.36 , 0.39]s	92	[0.71 , 0.73]s	118
KMS-BGS	[20.48, 20.84]s	66	[64.12, 65.20]s	88
GTH	[4.12, 4.47]s		[13.73, 13.99]s	
	2000		**5500**	
KMS+R	[0.82, 0.94]s	113	[5.54, 5.57]s	252
KMS-Pwr	[1.13 , 1.14]s	145	[6.86 , 6.92]s	305
KMS-BGS	[157.16, 157.77]s	113	[2998.26, 3011.79]s	252
GTH	[33.37, 33.75] s		[633.92, 638.27]s	

Table 4.4: Computation time in seconds and number of iterations for modulating matrix M2.

The usual version of KMS-BGS is not efficient because the problem is not balanced: we consider a problem with two large blocks. Our implementation of KMS-BGS begins with the computation of the blocks $(I - P_{i,i})^{-1}$, a task which is very time consuming for our problem. The numerical results show that the time needed by the KMS-BGS algorithm slowly increase with the number of iterations. This is due to this constant time needed to compute the fundamental matrices associated with the diagonal blocks.

The KMS+R and KMS-Pwr algorithm have a different timing behavior. When the matrix size is constant, the times increases almost linearly with the number of iterations. This is consistent with the complexity of the problem. However our algorithm (KMS+R) requires less iterations and less computation times until convergence.

4.3.3 Example

We now consider an application to a more realistic traffic. We still assume two phases for the modulating Markov chain. During phase 1 the traffic is the same than the one we consider in section 4.2.3.2 while during phase 2 we assume heavier batches of arrivals. More precisely we keep the same support for the batch distribution but now the probability distribution is $(0.15, 0.1, 0.2, 0.3, 0.25)$. We still assume that the buffer size is 250 chunks of 50 bytes. We consider a deadline equal to 40 and the same threshold as in section 2.3. Note that due to the more frequent arrival of large batches, the average size of the batches is 11.1 chunks. The modulating chain is NCD and equal to $\begin{pmatrix} 0.999 & 0.001 \\ 0.001 & 0.999 \end{pmatrix}$.

The chain has 15340 states and 135160 transitions. The steady state distribution is computed by KMS+R using 4 iterations and 0.37s. We observe from the distribution that the average size of the container is 10093 bytes (i.e. 201.86 chunks) and the mean inter arrival time of the containers at the insertion node is 19.18 time slots. The average time to fill a container is 18.009 slots. The distributions are depicted in Fig. 4.14 and Fig. 4.15. Due to the heavy arrivals in phase 2 the containers are much more filled.

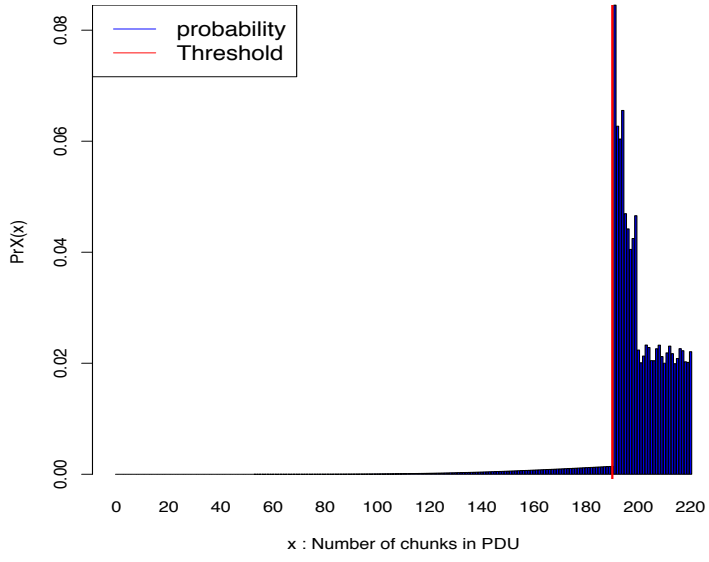


Figure 4.14: ModelSBBP: Distribution of the PDU size at release time (chunks of 50 bytes).

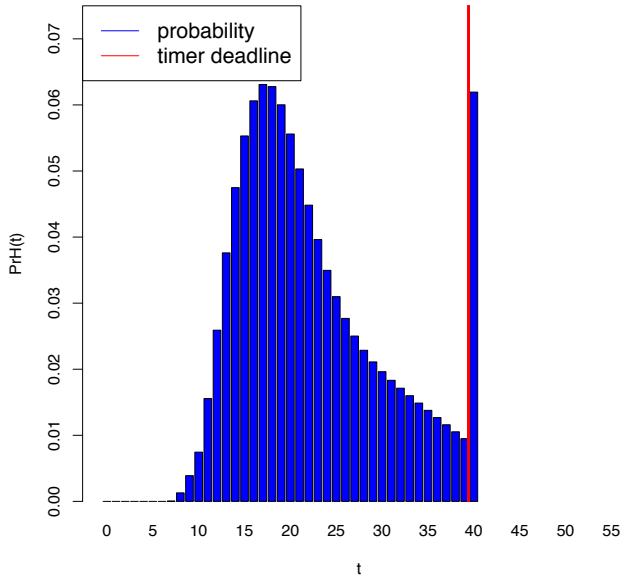


Figure 4.15: ModelSBBP: Distribution of the Timer at release time.

4.4 Modeling the container insertion on the optical ring

We first explain some constraints due to the technology we used to design the NGREEN network. First, the network is an optical ring where optical containers are inserted by stations. The ring network is synchronized and divided into time slots. The communication mode is the so-called Broadcast and Select. Under this mode, the optical container may contain SDU destined to various stations. All the stations copy the container to be converted to electronic packets. The only station which can free the slot on the ring from the container is the sending station. Therefore a container moving on the ring makes one turn and is copied by all the stations.

The third important feature already mentioned in the introduction is the use of WDM packets. The NGREEN network is built on slots: 1-µs long in the time domain and covering 10 WDM channels in the spectral domain. The optical container has 10µs long time duration (i.e. 12500 bytes, for an emission at 10 Gbs). The container is filled by aggregating different service data units and the optical container is spread over the 10 wavelengths. Thus, the WDM packet has a duration which is 10 times smaller than the optical container previously filled (this is illustrated in Fig. 4.16).

The slot assignment to containers may be opportunistic or based on a reservation [Gra+18]. Next, we analyze these two insertion modes and give numerical results of the delays. The container once it has been released by the filling mechanism waits for an empty slot on the ring. It must also wait 10 slots after the last insertion on the ring. This is a consequence of the conversion of a 10µs PDU into a WDM packet

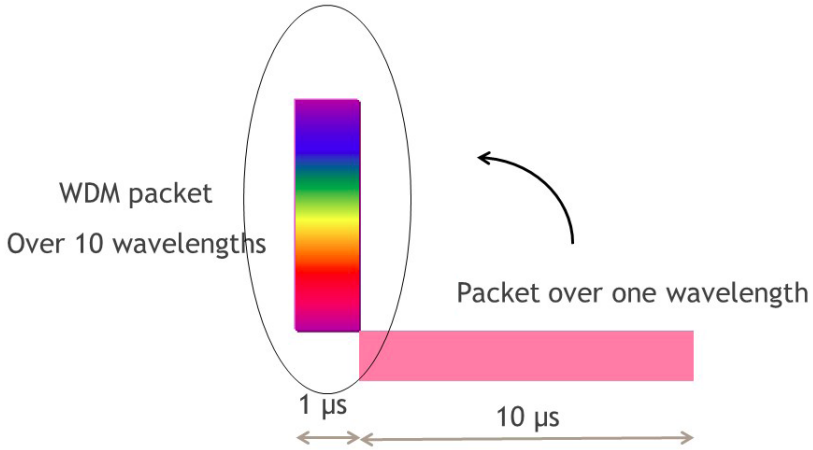


Figure 4.16: The optical conversion at the insertion at a NGREEN node (from [D C17]).

of 1µs. Remember that the slot is freed by the node which has used it for emission, after one turn of the ring. A container can be immediately inserted at its arrival if the conditions of insertions are respected. We consider two scenarios which differ by the kind of the insertion mode of the PDU into the ring : opportunistic or slot reservation. We first begin with the opportunistic insertion mode, by a simulation on the ring to find the number of stations which makes the network stable with distribution of PDU inter-arrival time given in Fig.4.9. The results are depicted in Fig. 4.17.

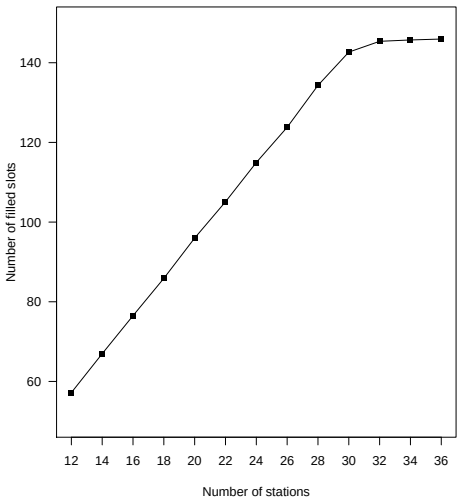


Figure 4.17: Average ring occupancy versus number of stations.

We also check with the number of containers waiting for insertion. Clearly, the system is stable up to 30 stations. In the following we will study a network with 22 stations (when the load on the ring is low) and a network with 28 stations which provide a high load to the ring.

4.4.1 Scenario A : latency with opportunistic insertion mode

We build a discrete time simulation engine to study the insertion delay for a container to enter the optical ring. As we model an Optical Slot Switching system, a simulator based on discrete time is easier to build and to use than a continuous event engine. The main loop of the simulator proceeds as described in Algorithm 5.

Algorithm 5: Discrete time simulator

Input : The number of stations, and the inter-arrival distribution of containers.

Output: Statistics on the ring occupancy, and insertion time distribution.

```
Initiate the simulation time t ← 0;
while !(simulator stop condition) do
    // − − − Make the ring turn a slot − − −
    Increase the global clock t ← t + 1;
    for all the stations do
        1) Free the slot in front of the station if it was occupied by the
         station for a container;
        2) If the slot is empty and if Δ = 0, and if there is a container waiting
         for insertion, then: put the container on the ring, remove the PDU
         for the queue, perform some statistics on the insertion delay, and
         let Δ = 10;
        3) If an arrival of a fresh PDU occurs, then compute the next arrival
         instant using an inverse transform method based on the discrete
         distribution obtained in section 4.2;
        4) Perform some statistics on the queue occupancy ;
        5) Let Δ = Δ − 1;
    end
end
Derive some statistics as the mean insertion delay for all stations, and the
 ring occupancy.
```

We assume that all the nodes receive the same traffic which obeys the same model studied in section 4.2.3.1. We suppose initially that 22 insertion nodes are connected to the ring. The distribution of the time between two arrivals of container is given by the distribution depicted in Fig. 4.9.

In Fig. 4.18, we show the distribution of the insertion time in slots for the first station. The number of containers waiting for insertion is extremely small: during the simulation we observe between 0 and 1 containers waiting. Note that due to the scheduling used in the simulation a container may be inserted immediately at its arrival. The probability to observe 2 containers waiting is less that 10^{-4} with a confidence interval of 95%. Thus a very small buffer will be sufficient in the line card. The utilisation of the ring is depicted in Fig. 4.20. After a small transient period, the ring occupancy has small oscillations around 105 used slots (remember that the ring contains 150 optical slots). We remove the first part of the sample path (typically for time smaller than 1000 slots) to avoid the transient phenomenon. The

simulations are long enough to reach steady-state. We propose now to increase the number of insertion nodes to 28, and we see in Fig. 4.19, the probability distribution reachs higher insertion delays due to a higher load, typically around 135 used slots among 150. For this case, the number of containers waiting is up to 6 and the probability to observe 6 containers waiting is about 10^{-2}.

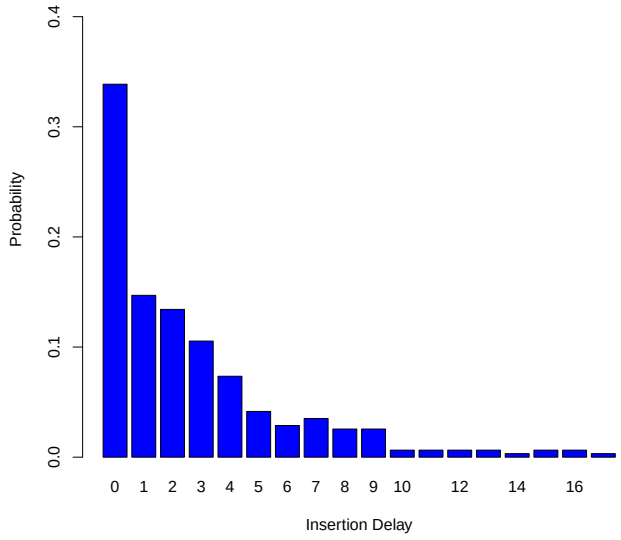


Figure 4.18: $\mathscr{I}$: Distribution of the insertion time (in slots) for the first station of 22 stations.

We now compute $\mathscr{E}2\mathscr{E}$ the distribution of the end to end delay. Remember that the duration to fill a PDU is $\mathscr{D}$. As the arrivals of SDU are supposed to be independent, the arrival instants of optical containers are independent. Therefore the distribution of end to end delay for a PDU is the convolution of $\mathscr{D}$, $\mathscr{I}$ (the insertion delay) and $\mathscr{T}$ (the transportation delay on the ring). Transportation delay is upper bounded by the size of the ring as we do not make any assumption on the spatial distribution on insertion stations along the ring. See Fig. 4.21 and Fig. 4.22 for $\mathscr{E}2\mathscr{E}$ distribution in case of 22 (resp. 28) stations in the ting..

$$\mathscr{E}2\mathscr{E} = \mathscr{D} \otimes \mathscr{I} \otimes \mathscr{T}$$

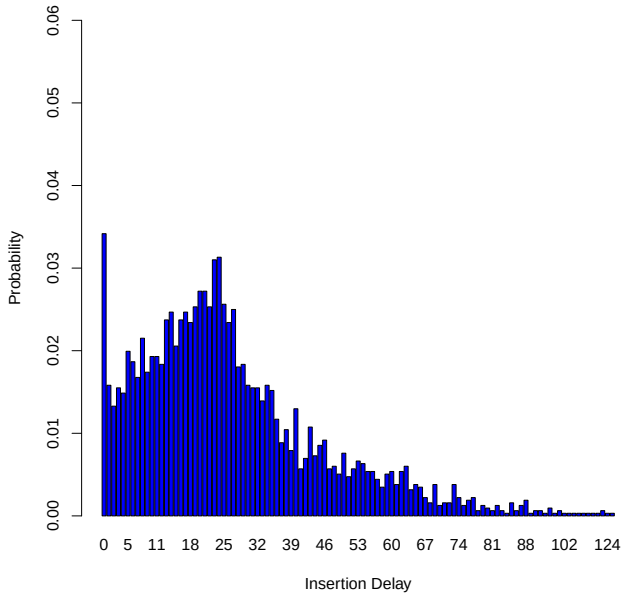


Figure 4.19: $\mathscr{I}$: Distribution of the insertion time (in slots) for the first station of 28 stations.

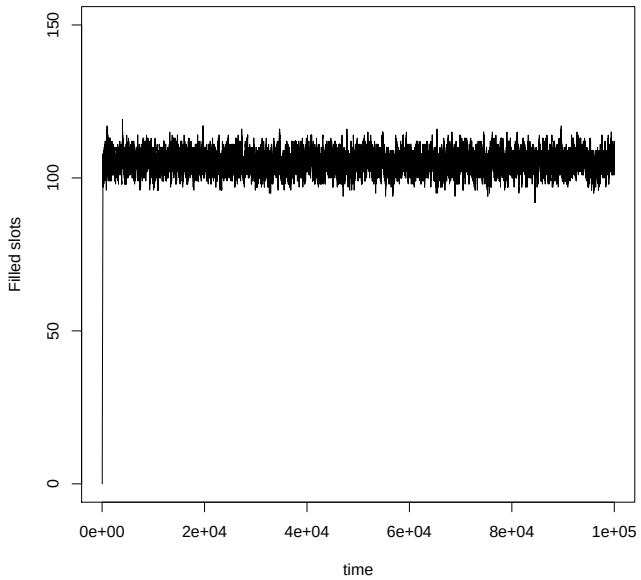


Figure 4.20: Ring occupancy versus simulation time.

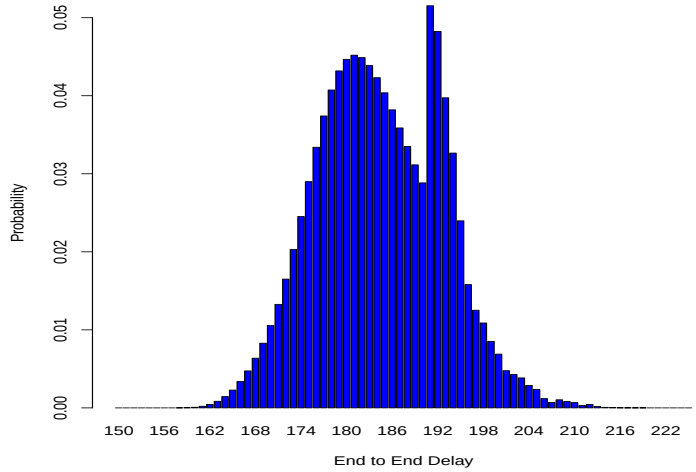


Figure 4.21: E2E: distribution of the end to end delay, case of 22 stations.

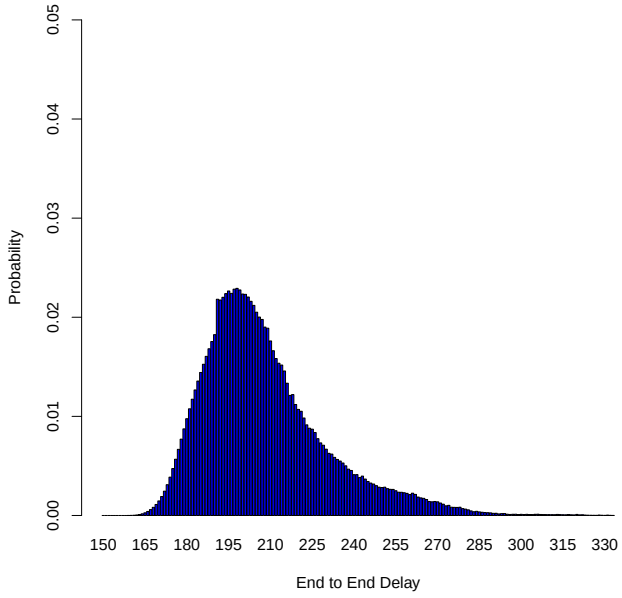


Figure 4.22: E2E: distribution of the end to end delay, case of 28 stations.

Next, we propose to analyze another insertion mode, allowing a greater stability when the number of insertion nodes increases.

4.4.2 Scenario B : guarantee latency with slot reservation insertion mode

We model the slot reservation mode. A station can only use the slots that are periodically reserved for it. We assume that a slot is reserved every D slots on the ring. The arrivals are independent and the inter-arrivals of containers follow the random process, the distribution of which was computed in Section II-C.

We consider a discrete time model, the time unit of which is the WDM slot duration. The model is based on two clocks: H1 to model the arrival of a reserved slot on the ring and H2 to model the inter-arrival of an optical container. Both clocks are synchronized. However clock H1 makes a deterministic jump of length D after the arrival of the reserved slot while clock H2 makes a random jump after an arrival of a container. The length of the jump is a random variable distribution of which is given by the steady-state distribution computed in Section 4.2.3.1 (Fig. 4.9).

We also take into account the fact that an arriving container may be inserted immediately if the reserved slot is available. This is modeled by the scheduling used in the dynamics of the clock (i.e. H2 first). We assume that the arrival of a container or a slot on the ring are modeled by the transition of the clocks out of 0. The model also contains the population of container waiting for insertion. Let X be this random variable. We assume that the buffer size is B. Clearly $(\mathrm{H1},\mathrm{H2},\mathrm{X})_n$ is a Discrete Time Markov Chain. More precisely the dynamics are the following at each time slot.

- First, if Clock H2 is positive, then it is decreased, otherwise as it is equal to 0, it jumps to state $\mathrm{K}-1$ while component X is increased if it is not equal to B. If an arrival occurs while the buffer size is reached, the incoming customer is lost. This probability will be computed in the following. K is the inter-arrival delay. It is distributed as $\mathscr{F}$.

- Second, if Clock H1 is positive, then it is decreased, otherwise it is equal to 0 and it jumps to state $\mathrm{D}-1$. During the same transition, component X is decreased by 1 if is not 0.

The model is generated using XBorne and it is solved numerically using a standard numerical technique (i.e. Sparse version of the Grassman, Taksar and Heyman Algorithm [Wil94]). As the matrix is very sparse, the computations only require a few seconds for a matrix of size 8500. Once we have obtained the steady state probability (say $\mathrm{P}r(\mathrm{H1},\mathrm{H2},\mathrm{X})$), one can numerically compute the following quantity:

- The loss probability for a container :

$$\mathrm{P}_{Loss} = \sum_i \mathrm{P}r(i,0,\mathrm{B}) \tag{4.9}$$

- The buffer population :

$$E[X] = \sum_i \sum_j X \; Pr(i, j, X) \tag{4.10}$$

- The distribution of the insertion time for an arriving container: when a container arrives at state $(j, 0, Y)$ (with $Y < B$, otherwise it is lost), it has to wait $(Y * D + j)$ time units. Note that we must consider in the distribution the conditional probabilities knowing that a new container arrives :

$$\frac{Pr(j, 0, Y)}{\sum_i \sum_{X<B} Pr(i, 0, X)} \tag{4.11}$$

The Markov chains are solved for the same traffic assumptions we already use for the simulation. Furthermore the parameter D is supposed to be equal to the number of stations to share equally the bandwidth among the stations. The buffer size is supposed to be small. We consider a buffer which can contain up to 4 containers. The numerical analysis show that this buffer is sufficient to avoid losses (see Tables in Fig. 4.23).

#Stations	**22**	**28**
Losses	10^{-17}	10^{-8}

Optical containers	**0**	**1**	**2**	**3**	**4**
22 Stations	0.65	0.34	10^{-3}	10^{-7}	10^{-12}
28 Stations	0.46	0.50	0.03	10^{-4}	10^{-5}

Figure 4.23: Loss probability & Distribution of the optical containers in the buffer

In the case of the slot reservation insertion mode, we can see in Fig.4.24 (resp. Fig. 4.25), the distribution of the insertion delays of the first station for 22 (resp. 28) nodes. If we compare with the opportunistic insertion mode in Fig. 4.18 and Fig. 4.19, we can remark that insertion delays are smaller in the opportunistic insertion mode, for 22 nodes, and higher for 28 nodes. In fact, for 28 stations, the number of containers waiting in the node is up to 6 in the opportunistic insertion mode, against up to 4 in the slot reservation mode. So the slot reservation insertion mode is really interesting when the number of nodes increases.

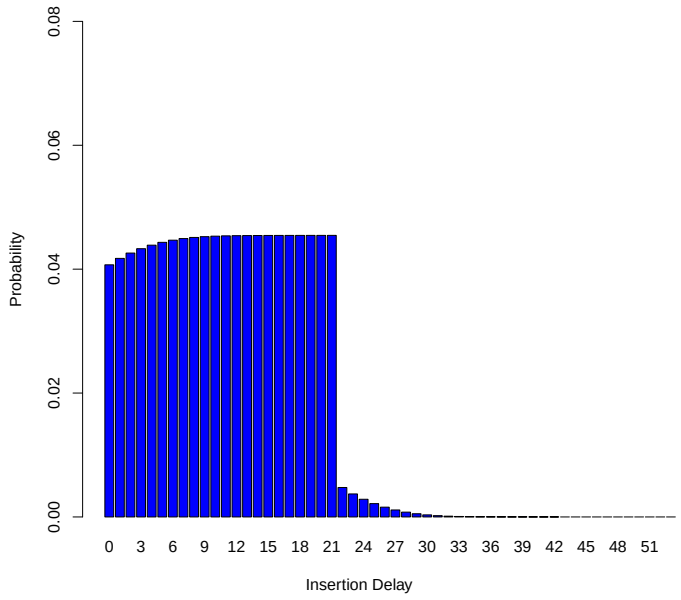


Figure 4.24: $\mathscr{I}$: Distribution of the insertion time (in slots) for the first station of 22 stations.

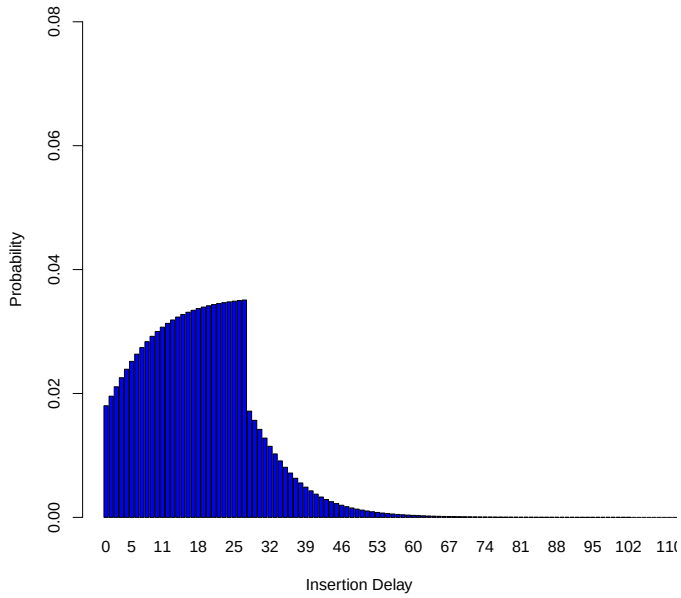


Figure 4.25: $\mathscr{I}$: Distribution of the insertion time (in slots) for the first station of 28 stations.

4.5 Energy efficiency and latency analysis

We consider a scenario with 10 stations, with the opportunistic insertion mode into the ring, and we try to analyze both end to end delays, and energy efficiency. In the NGREEN project, the energy consumed with respect to the different equipment crossed is 0.8 nJoule/bit. For the transmission of an optical packet of 12500 bytes we need $12500 * 8 * 0.8$ nJoules, so the energy efficiency represents the ratio between 80 µJoules and the payload of the PDU which depends on the filling. We give the energy efficiency as

$$\frac{ContainerCapacity * 8 * 0.8 * 10^{-9}\ (Joules)}{Container\ occupancy\ (bit)} = \frac{80 * 10^{-6}\ (Joules)}{Container\ occupancy\ (bit)}.$$

In Fig. 4.26, we have represented energy efficiency for different filling ratios and deadlines. Obviously, the larger the thresholds and the deadlines, the larger the occupancy of the container, then the smaller (and better) the energy efficiency. We have also represented in Fig. 4.27, the end to end delays according to the deadlines and the thresholds ratios, in order to analyze the impact of the traffic aggregation on the end to end performances. So we can see that the higher threshold ratio (0.90) gives the higher end to end delay due to the high delays to fill the PDU (as shown in Fig.4.10), but the lowest and the best energy efficiency. These results are obtained for the opportunistic insertion mode, and allow to provide the thresholds and the deadlines which guarantee both energy efficiency and end to end delays. In the case of the slot reservation insertion mode, the energy efficiency will be the same as it depends essentially on the parameters of the filling process (threshold and deadline). The insertion delays will be higher as the number of stations is small.

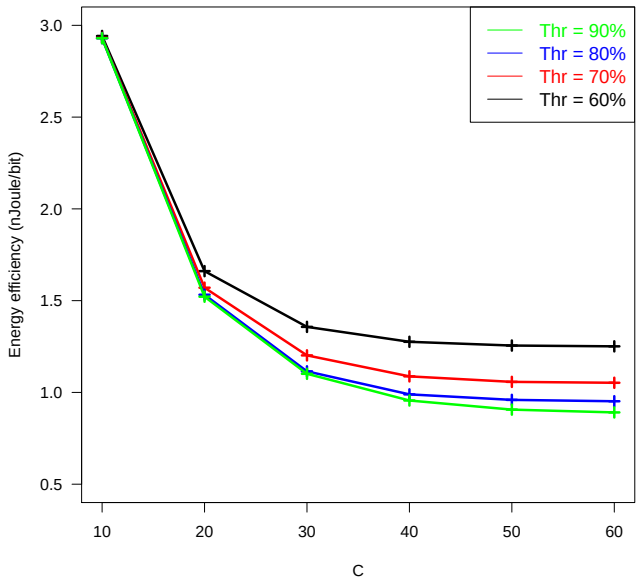


Figure 4.26: Energy efficiency versus deadline.

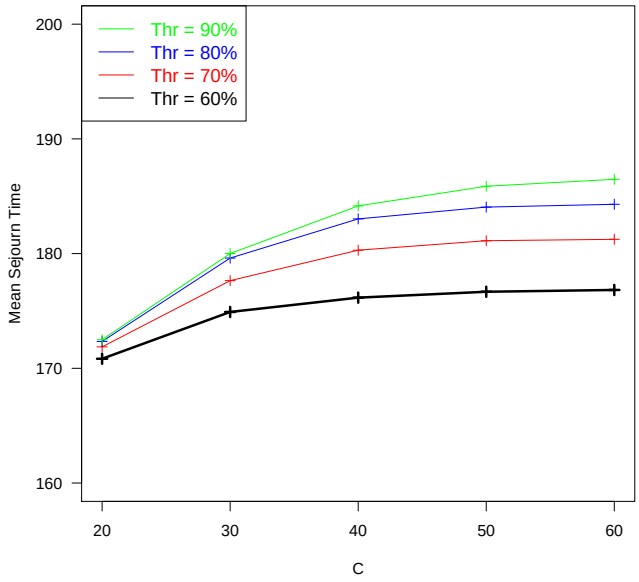


Figure 4.27: End to end delays versus deadline.

Remark

The optical container filling model considered in this work is much wider. The model supports any filling system subject to a global clock. For instance, the charging of a battery with energy packets.

Part III

Analytic analysis of power consumption in a cloud center

Chapter 5

State of the art

Contents

Half of the power consumption of a data center is due solely to the cooling of computer devices, particularly servers. The need to reduce the energy consumed by servers is constantly increasing. Servers consume 60% of their maximum energy by being inactive. Turning on an off server consumes even more energy. Most companies are reluctant to shut down inactive servers because the cost of powering them up is very high, typically 200sec, while at most one task is processed in 1 second ([Kri10] and [DeC07]). So before making the right decision, studies must be done to obtain an estimate of costs, energy, losses and other performance indicators.

In 2015, in France, digital technology consumed approximately 56 TWh of electricity out of a total of 476 TWh, i.e. about 12% of the country's electricity consumption. According to [dec] calculations, since electricity accounts for approximately 25% of final energy consumption in France, digital technology now accounts for approximately 3% of the country's final energy consumption. Details are in Fig. 5.1.

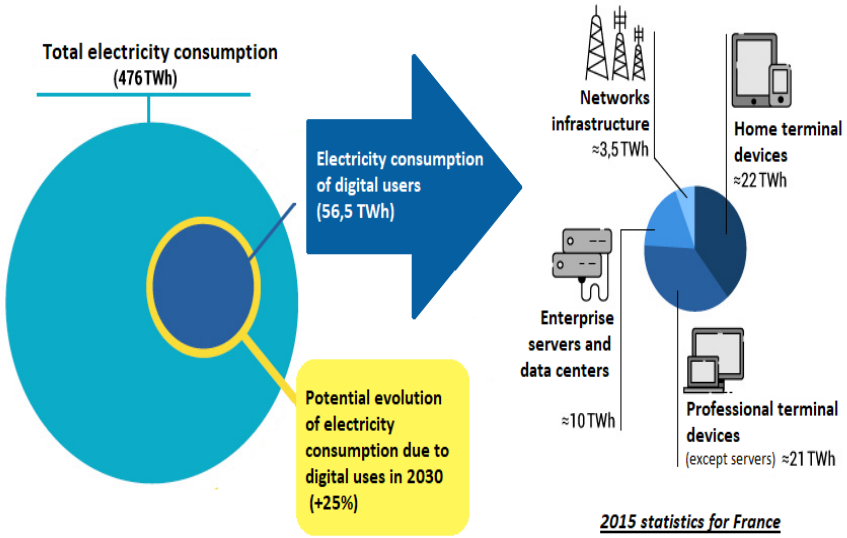


Figure 5.1: Statistics of electricity consumption in France, 2015 (Source: [dec]).

In order to reduce high operating costs, as well as CO2 emissions, one of the greatest future challenges is the improvement of energy efficiency in data centers. In [A+17], the authors analyze using traces some shutdown policies in order to study their impact on energy saving. Shutdown policies are often combined with consolidation algorithms that gather the load on a few servers to favor the shutdown of the others. The resources (servers, Virtual Machines (VMs)) can be in one of the following states: OFF, ON, and the ON state is divided into two sub-states: Idle or Run. The power consumption for OFF state is low (sometimes considered zero), but in the Idle state as the resource is powered On, then the consumption could reach half of its maximum consumption when it is in activity. In the Run state, the consumption increases with the resource utilization. Moreover, energy consumption for turning ON, and turning OFF is not negligible and has to be taken into account for the shutdown policies. So turning OFF resources could be energy saving only if the idle periods are long enough to make the transitions ON/OFF

and OFF/ON. Migration of tasks or VMs can also be used in order to balance the load between overused servers to unused servers [LA10]. This helps to preserve the nodes from overheating and to prevent overused nodes to have technical problems prematurely. It is linked to the principle that it is better to have a working node than a turned ON node which is doing nothing. Several papers have addressed the problem of load balancing in distributed networks to avoid server overload, and to improve the response time [ASJ17; JJ06; MN91]. Another approach, is to use consolidation [A13] in order to overload some servers and to shut down others. It could be efficient for long shut down periods. In [A+15] this approach is used with PIKA simulation framework aiming at reducing the brown energy consumption (i.e. from non-renewable energy sources), and improving the usage of renewable energy for small mono-site data centers. Simulators as "CloudSim, SimGrid, ns-2 and ns-3, ecofen (for wired networks) [AC+17] ..." have been widely used in the last decade to evaluate cloud performance and energy consumption. Virtualization in cloud data centers allows one to create several VMs on a physical server, and therefore, reduce the amount of hardware in use and improve the utilization of resources. Many studies as consolidation algorithms, load balancing algorithms, power saving under performance/power budget/CO_2 emission constraints are discussed in [A13][A+15].

5.1 Load balancing of tasks

Load balancing technique consists on Balancing the workload among different servers, in order to (i) achieve some equilibrium load between servers so that no servers are in Idle state, this solution clearly increases the performance and could be energy efficient depending on the energetic equation considered. (ii) Another concern of the load balancing is the consolidation, it could be efficient to gather the workload into a single physical server in order to shutdown the sending load servers for a long period. In [KR13] authors presents an energy conscious, power aware load balancing strategy based on adaptive migration of VMs. This strategy will be applied to virtual machines on cloud, considering higher and lower thresholds for migration of VMs on the servers also they consider RAM and Bandwidth for better performance and load balancing. Load balancing is very common in queueing theory [ASJ17; JJ06].

Many communication networks and computer systems use load balancing to improve performance and resource utilization. The ability to efficiently divide service requests among system resources can have a significant effect on performance and energy consumption. Static [SM91; Sou+12] load balancing mainly based on the information about the average of the system work load. It does not take the actual current system status into account. In [AD85] static load balancing algorithms aim at finding optimal customer routing to optimize the throughput and other performance indices under certain constraints. Dynamic or adaptive load balancing policies are the most efficient ones. The system reacts dynamically to the network state and traffic is directed to routes with less load or extra load, depending on a cost function. Dynamic load balancing algorithms are usually classified into two families: receiver-initiated and sender-initiated [DEJ86]. In the first case, an overloaded node decides to send some of its job to another node that receives them and

tries to process or reallocate them elsewhere. In the latter case, it is the receiver that decides when it can poll another node to import some of its jobs. In [DEJ86] also in [RDJ90] for heterogeneous systems, authors compare sender- and receiver-initiated strategies and conclude that, under heavy load, receiver-initiated algorithms give lower expected response time than sender-initiated one. Also In [ASJ17], authors address the problem of dynamic load balancing for networks with open topology where an arbitrary number of nodes implement a receiver-initiated dynamic load balancing algorithm. The algorithm's point is to compute the polling rates among the network stations that ensure both the network to be in product-form and that the sets of specified stations have their load balanced.

Regarding the literature on product-form analysis (see e.g. [SJH13; SA13; GM15; Gar+16]) for some works in the field, LB networks present important particularities. Compared to the literature on signals in G−networks and similar models ([XMM99; Gel93a; Gel93b; Gel93d; FV95; AB12]), LB networks have the property that the network population is preserved by node interactions. In Chapter 6, we present a study based on the load balancing between physical servers, in order to reduce energy consumption of the system.

5.2 Server's power consumption

The power consumption of a server depends on the server's components. It is well known that a server has a basic or static consumption (consumed even when the server is in an idle state), and a dynamic consumption. Dynamic power consumption is the power needed by a server to execute a task. Static and dynamic consumption depends on the configuration of the server, for instance, for a physical server the power consumption will depend on the number of VMs hosted, the number of cores in an single VMs, cores speed, cores/VMs states considered, and the VMs migration cost if this technique is considered. The power consumption of a server is generally related to its workload. When the server is not virtualized and the applications it runs are known, the term server load has a meaning relative to the applications present. For example, the load of a web server can be defined as the number of requests it receives per second. While the literature does not provide a general model of servers consumption, it does, however, provide different approaches some of them are presented in Table 5.1.

Works	**Characteristics.**
[Gao+13; BAB12; Bas+11]	Linear power model.
[BBA10; Lim+11]	Non linear power model based on mathematical integration.
[GNS11; MD15]	Non linear model based on queueing theory.
[WTB10; Inc05]	Non linear model based on the CPU utilization
[G14]	Quadratic power model.
[Zha+13]	Cubic polynomial power model

Table 5.1: Server's power modeling.

Chapter 6

Analysis of power consumption in cloud/data center with tasks migrations

Contents

6.1 Introduction

In this study, we use the load balancing as a resource management, in order to see the impact on the power consumption. We use Jackson multi-server network to model the servers of the data center. Queueing theory and Mean Field analysis have already been used for performance analysis of clouds. Models include M/M/C queues to study the minimum of resources required for the performance [G+16] or the reliability of the cloud [B+09]. Infrastructure as a service (IAAS) clouds have been modeled by M/G/m/m+k queues in [X+16]. Mean Field approximation have been proposed for the asymptotic analysis of large scale load balancing network with general distribution [RLK15]. Dealing with general arrivals and services processes, one can also use a diffusion approximation ([W04],[T+09]). Such an approach was published recently to build a model which explicitly represent energy and data in sensor networks with energy harvesting [HE16].

In this work, we study how to optimaly minimize the mean power consumption using an exact analysis of the queueing network with customer migration. We use the multi-server Jackson network to represent the behavior of the data center. Each physical server is represented by a multi-server station, and the VM (Virtual machines) are the servers of the stations. The routing between the stations represents the task migrations. The exact distribution provided by Jakson model allows to compute the optimal migration rate. An extension of this work consists in using the new product form result published by Gelenbe and his colleagues ([EE15],[EE16]). This model explicitly represents the energy needed by data transfer and computation.

The chapter is organized as follows: next, we give the power consumption equations computed in a data center, then in section 6.3 we derive the mathematical formula of the mean power consumption and we give the upper bound of the migration power which reduces the global power consumption. We give the migration rate which minimizes the mean power consumption. In section 6.4, we show with a numerical example how to choose the parameters (power migration, migration rates) in order to minimize the power consumption.

6.2 Power consumption model

In a data center, the power consumption can be divided into static and dynamic parts. The static parts are the base costs of running the data center when being idle and the dynamic costs depend on the current usage [MAA16]. For the data center with n servers, we denote s_i the power consumption of server i, for $i = 1, \ldots, n$. Each server hosts a set of (VMs) Virtual Machines that perform tasks, and m represents the number of VMs hosted by a server.

Let $\mathrm{P}wr_{total}$ be the total power consumed in the data center, and $\mathrm{P}wr(s_i)$ the power consumed by a physical server. From [G14] we have:

$$\mathrm{P}wr_{total} = \sum_{i=1}^{n} \mathrm{P}wr(s_i).$$

The power consumed by a server is the sum of static power (δ) when the server

is powered ON (and idle) and the dynamic part (β and α) depends on the CPU resource utilization of the VMs hosted in the server when they are performing tasks. As the CPU is one of the highest power consumers on a node, in many studies this is the most considered (compared to the RAM or disk). In [G14], the consumption of a server is defined as a polynomial function of degree two of the CPU load of this server. It is therefore an extension of the linear model, which is a polynomial of degree one.

$$\mathrm{P}wr(s_i) = \delta + \mathrm{CPU}(s_i) * \beta + \mathrm{CPU}^2(s_i) * \alpha, \tag{6.1}$$

We consider that $\mathrm{CPU}(s_i)$ is the CPU utilization ratio of the server i, which is divided between each VM of the server:

$$\mathrm{CPU}(s_i) = \sum_{j=1}^{m} \mathrm{CPU}(vm_j),$$

where $\mathrm{CPU}(vm_j)$ is the CPU ratio allocated to the vm_j. We suppose that all the virtual machines are homogeneous in terms of CPU utilization [X+16; R+13]. That means that the physical resources allocated to each VM are of the same capacity and type. We consider a single-task policy, each job or task is served by one virtual machine, so the mean number of activated VMs corresponds to the mean number of tasks in service. As the virtual machines are homogeneous, then $\mathrm{CPU}(vm_j) = \mathrm{CPU}(vm)$, for all j. And if $\mathbb{E}(n_{s,i})$ is the mean number of tasks in service (or activated VMs), then:

$$\mathrm{CPU}(s_i) = \mathbb{E}(n_{s,i}) * \mathrm{CPU}(vm).$$

As an example, if we consider a data center with 2 physical servers, then Equation (6.1) gives, for $i \in \{1,2\}$:

$$\mathrm{P}wr(s_i) = \delta + \mathbb{E}(n_{s,i}) * \mathrm{CPU}(vm) * \beta + (\mathbb{E}(n_{s,i}) * \mathrm{CPU}(vm))^2 * \alpha.$$

The consumption of a physical server is upper bounded by $\delta + \beta + \alpha$: let's m be the number of VMs in a physical server, $\mathrm{CPU}(vm) = \frac{1}{m}$ since the VMs are all homogeneous and receive the same utilisation ratio as mentioned above. As $\mathbb{E}(n_{s,i}) \leq m$ then we deduce that $\delta + \beta + \alpha$ is an upper bound of $\mathrm{P}wr(s_i)$.

We note that in this section, we have not considered migrations of tasks and therefore power consumption due to migrations, but it will be added in the following. Next, we compute analytically the power consumption in a data center using multi-server Jackson network with task migrations for load balancing.

6.3 Jackson network model for task migrations

We consider a data center system with two physical servers. We suppose now that the system performs migrations from overloaded to unloaded server in order to avoid to shutdown the unloaded servers. We model the system with a Jackson network. Each station represents a physical server, where tasks are either waiting in the queue, or in activity (in service or in migration). This limitation to system with two servers is due to the optimization process detailed in section 6.3.3. Of course

the queueing model can accommodate larger systems. In section 6.4 we extend the study to models with $N > 2$ physical servers.

A task can migrate from one station to another one. We assume that for each station i ($i = 1, 2$), the external arrivals of tasks follow independent Poisson processes with rate λ_i . We also assume that the services and the migration durations are independent and distributed according to exponential distributions with rate μ_i ($i = 1, 2$) and $\gamma_{i,j}$ ($i, j = 1, 2$, and $i \neq j$) for the task migration from station i to station j.

We suppose that each physical server is associated to m logical servers (VMs) as in [G+16]. If the number of tasks is larger than m, then they wait in the queue, and only m among them are in service or involved in a migration. We also assume that the queueing capacity is infinite. No task can be rejected at its arrival.

The logical service rate in a queue could be a real service or a migration. As both activities are independent and exponentially distributed, their parallel composition can be seen as an exponential duration with rate $\mu_i + \gamma_{i,j}$ ($i, j = 1, 2$ and $j \neq i$). After this logical service, a task can leave the system (being stopped) or stochastically route to the other queue (migrate). This is typically a race condition between two events as in a stochastic Petri net. The probabilities of these events are given in Lemma 6.3.1.

6.3.1 Analytic results for the queues

The state of station i is x_i, which represents the number of tasks waiting or in activity. The task scheduling discipline is FCFS (First Come First Serve). We assume that both stations contain m logical servers. Therefore $min(m, x_i)$ task are in service or in a migration activity while $(x_i - m)^+$ are queued. Each queue in isolation can be seen as a M/M/m queue, using Kendall notations. Under the classical assumptions we mention earlier, $X_t = (x_1, x_2)_t$ is a Markov chain, and the transitions are as follows:

$$
\begin{aligned}
(x_1, x_2) \quad &\rightarrow \quad (x_1 + 1, x_2) \text{ with rate } \lambda_1, \\
&\rightarrow \quad (x_1, x_2 + 1\}), \text{ with rate } \lambda_2, \\
&\rightarrow \quad (\max\{0, x_1 - 1\}, x_2), \text{ with rate } min\{x_1, m\} \cdot \mu, \\
&\rightarrow \quad (\max\{0, x_1 - 1\}, x_2 + 1), \text{ with rate } min\{x_1, m\} \cdot \gamma_{1,2}, \\
&\rightarrow \quad (x_1, \max\{0, x_2 - 1\}) \text{ with rate } \quad min\{x_2, m\} \cdot \mu, \\
&\rightarrow \quad (x_1 + 1, \max\{0, x_2 - 1\}) \text{ with rate } min\{x_2, m\} \cdot \gamma_{2,1} \quad .
\end{aligned}
$$

We know that under ergodicity conditions, Jackson networks have a steady-state distribution $\Pi(x_1, x_2) = \Pi_1(x_1)\Pi_2(x_2)$ which has a product form [Jai92]. $\Pi_1(x_1)$ and $\Pi_2(x_2)$ are steady-state distributions of two $M/M/m$ queues the parameters of which are given by the flow equations between traffic intensities ρ_1 and ρ_2.

Theorem 6.3.1 *Let $(x)_t$ be the Markov chain that represents a M/M/m queue. x is the number of tasks in the station. Let $\rho < 1$ be the traffic intensity and $\Pi(x)$ the steady-state distribution. Then from [Jai92], we have:*

$$
\Pi(x) = \begin{cases} \Pi(0) \frac{(m\rho)^x}{x!}, & x < m \\ \\ \Pi(0) \frac{\rho^x m^m}{m!}, & x \geq m \end{cases} \tag{6.2}
$$

and

$$\Pi(0) = \left[1 + \frac{(m\rho)^m}{m!(1-\rho)} + \sum_{x=1}^{m-1} \frac{(m\rho)^x}{x!}\right]^{-1}. \tag{6.3}$$

It remains to write the flow Equation for our model.

Lemma 6.3.1 *In the networks we consider, the fresh arrivals at station i ($i = 1, 2$) follow Poisson processes with rate λ_i. The parallel composition of service and migration results in an activity with an exponential duration with rate $\mu_i + \gamma_{i,j}$ ($i, j = 1, 2$ and $i \neq j$), followed by a departure with probability $\frac{\mu_i}{\mu_i+\gamma_{i,j}}$ ($i, j = 1, 2$, $i \neq j$) or a migration with probability $\frac{\gamma_{i,j}}{\mu_i+\gamma_{i,j}}$ ($i, j = 1, 2$, $i \neq j$). Using the flow equations for Jackson networks, we get:*

$$\rho_1 = \frac{\lambda_1 + m(\mu 2 + \gamma_{2,1})\rho_2 \frac{\gamma_{2,1}}{(\mu 2+\gamma_{2,1})}}{m(\mu 1 + \gamma_{1,2})}, \quad \rho_2 = \frac{\lambda_2 + m(\mu 1 + \gamma_{1,2})\rho_1 \frac{\gamma_{1,2}}{(\mu 1+\gamma_{1,2})}}{m(\mu 2 + \gamma_{2,1})}.$$

And after simplifications we obtain the flow Equation of the model with migration:

$$\rho_1 = \frac{\lambda_1 + m\rho_2\gamma_{2,1}}{m(\mu 1 + \gamma_{1,2})} \quad \textit{and} \quad \rho_2 = \frac{\lambda_2 + m\rho_1\gamma_{1,2}}{m(\mu 2 + \gamma_{2,1})}. \tag{6.4}$$

We now define some notations we will use in the following of the study. Consider a task in a station, it is either served or implied in a migration to another station. Let P_1 (resp. P_2) be the probability of service and 1-P_1 (resp. 1-P_2) the probability of migration:

$$P_1 = \frac{\mu_1}{\mu_1 + \gamma_{1,2}}, \; P_2 = \frac{\mu_2}{\mu_2 + \gamma_{2,1}}. \tag{6.5}$$

In order to compute power consumption of the system using the relations mentioned in the previous section, we need the mean number of tasks in service which correspond to the mean number of active VMs. Since our system has a product form solution, the rewards will be calculated separately for each station. Let $\mathbb{E}(n_{s,1})$ (resp. $\mathbb{E}(n_{s,2})$) be the mean number of tasks in service in station 1 (resp. station 2). It is well-known that (see for instance [Jai92] for a proof):

$$\mathbb{E}(n_{s,1}) = m\rho_1, \quad \textit{and} \quad \mathbb{E}(n_{s,2}) = m\rho_2. \tag{6.6}$$

Lemma 6.3.2 *The total power is the summation of the power consumed by each station plus the power Pm needed for the migrations:*

$$Pwr_{Jackson} = Pwr(s_1) + Pwr(s_2) + Pm.$$

Combining the power model (in Equation 6.1), the value of the average queue size (Equation 6.6) and the probability of serving (resp. migrating) (Equation 6.5), we obtain for physical server $i \in \{1, 2\}$:

$$Pwr(s_i) = \delta + \mathbb{E}(n_{s,i}) * \mathrm{CPU}(vm) * P_1 * \beta + (\mathbb{E}(n_{s,i}) * \mathrm{CPU}(vm) * P_1)^2\alpha, \tag{6.7}$$

while for the power consumed by the migration:

$$\mathrm{P}m = p_m \mathbb{E}(n_{s,1})(1-\mathrm{P}_1) + p_m \mathbb{E}(n_{s,2})(1-\mathrm{P}_2), \tag{6.8}$$

where p_m is the migration power for a task.
Combining Equations 6.7, and 6.8, we finally obtained after summation:

$$\mathrm{P}wr_{Jackson} = 2\delta + \left[\mathrm{P}_1\mathbb{E}(n_{s,1}) + \mathrm{P}_2\mathbb{E}(n_{s,2})\right]\mathrm{CPU}(vm)\beta + \left[\mathbb{E}(n_{s,1})^2\mathrm{P}_1^2 + \mathbb{E}(n_{s,2})^2\mathrm{P}_2^2)\right]\mathrm{CPU}(vm)^2\alpha + \mathrm{P}m.$$

Remember that, $\mathrm{CPU}(vm) = \frac{1}{m}$ since the VMs are supposed homogeneous as already mentioned in section 2. Then, we get:

$$\begin{aligned} \mathrm{P}wr_{Jackson} &= 2\delta + \left(\rho_1\mathrm{P}_1 + \rho_2\mathrm{P}_2\right)\beta + \left[(\rho_1\mathrm{P}_1)^2 + (\rho_2\mathrm{P}_2)^2\right]\alpha \\ &+ mp_m\rho_1(1-\mathrm{P}_1) + mp_m\rho_2(1-\mathrm{P}_2). \end{aligned} \tag{6.9}$$

Corollary 6.3.1 *The system without migration is modeled with migration rates equal to 0:*

$$\gamma_{1,2} = \gamma_{2,1} = 0.$$

Each physical server is an independent M/M/m queue. Let r_i be the traffic intensity of station i ($i \in \{1,2\}$) in the system without migration, then :

$$r_i = \frac{\lambda_i}{m\mu_i}, \tag{6.10}$$

And the mean number of tasks in service are $\mathbb{E}(n_{s,i}) = mr_i$. Since $\gamma_{1,2} = \gamma_{2,1} = 0$, we get $\mathrm{P}_1 = \mathrm{P}_2 = 1$ (see Equation (6.5)) and the power consumption of this system without migration is:

$$\mathrm{P}wr_{\mathrm{MM}m} = 2\delta + (r_1 + r_2)\beta + \left[r_1^2 + r_2^2\right]\alpha. \tag{6.11}$$

6.3.2 Systems comparison

In this section we compare the power consumption in both models. Clearly if migration cost is very high, there is no benefit to perform migration.

Assumption 6.3.1 *We consider that the two physical servers have different traffic intensities. More precisely and without loss of generality, we assume a heavy traffic in station 2 and a low traffic in server 1: $r_1 < r_2 < 1$. To minimize the power consumption, we assume that the migrations only occur from the heavy traffic station (station 2) to the low traffic one (station 1). Thus, $\gamma_{1,2} = 0$.*

Lemma 6.3.3 *Under this assumption, we have $\mathrm{P}_1 = 1$ and $\mathrm{P}_2 = \frac{\rho_2}{r_2}$.*

Proof. *As $\mathrm{P}_1 = \frac{\mu_1}{\mu_1+\gamma_{1,2}}$ the assumptions clearly implies that $\mathrm{P}_1 = 1$. Furthermore we have the following relations:*

$$\mathrm{P}_2 = \frac{\mu_2}{\mu_2+\gamma_{2,1}}, \quad r_2 = \frac{\lambda_2}{m\mu_2}, \quad \rho_2 = \frac{\lambda_2}{m(\mu 2+\gamma_{2,1})}, \quad \rho_1 = \frac{\lambda_1 + m\rho_2\gamma_{2,1}}{m(\mu 1)}. \tag{6.12}$$

Clearly $\mathrm{P}_2 = \frac{\rho_2}{r_2}$ holds.

Lemma 6.3.4 *Assume that VMs are homogeneous and the servers are equivalent (i.e.* $\mu_1 = \mu_2 = \mu$*), then the sum of the mean number of activated VMs in each model is a constant value.*

$$m\rho_1 + m\rho_2 = mr_1 + mr_2 = m\mathrm{K}, \qquad where \quad \mathrm{K} \in [0,2]. \tag{6.13}$$

Proof. *By solving the Equation system in Equation (6.4) and using* $\gamma_{1,2} = 0$*, we get:*

$$\rho_1 = \frac{\lambda_2\gamma_{2,1} + \lambda_1(\mu + \gamma_{2,1})}{m\mu(\mu + \gamma_{2,1})}, \qquad and \qquad \rho_2 = \frac{\lambda_2}{m(\mu + \gamma_{2,1})}. \tag{6.14}$$

So:

$$\Rightarrow \ \rho_1 + \rho_2 = \frac{\lambda_2\gamma_{2,1} + \lambda_1(\mu + \gamma_{2,1}) + \lambda_2\mu}{m\mu(\mu + \gamma_{2,1})} = \frac{\lambda_2(\mu + \gamma_{2,1}) + \lambda_1(\mu + \gamma_{2,1})}{m\mu(\mu + \gamma_{2,1})}.$$

$$\Rightarrow \ \rho_1 + \rho_2 = \frac{\lambda_2(\mu + \gamma_{2,1}) + \lambda_1(\mu + \gamma_{2,1})}{m\mu(\mu + \gamma_{2,1})}.$$

$$\Rightarrow \ \rho_1 + \rho_2 = \frac{\lambda_2 + \lambda_1}{m\mu} = \frac{\lambda_1}{m\mu} + \frac{\lambda_2}{m\mu}.$$

$$\Rightarrow \ \rho_1 + \rho_2 = r_1 + r_2.$$

Lemma 6.3.5 *Assuming that* $\gamma_{2,1} \geq 0$ *then the following relations hold:*

$$\rho_2 \leq r_2 \quad and \quad \rho_1 \geq r_1.$$

Proof. *From Equation (6.10) and (6.14) we have:*

$$\rho_1 - r_1 = \frac{\lambda_2\gamma_{2,1} + \lambda_1(\mu + \gamma_{2,1})}{m\mu(\mu + \gamma_{2,1})} - \frac{\lambda_1}{m\mu} = \frac{\lambda_2\gamma_{2,1}}{m\mu(\mu + \gamma_{2,1})} \geq 0,$$

$$\rho_2 - r_2 = \frac{\lambda_2}{m(\mu + \gamma_{2,1})} - \frac{\lambda_2}{m\mu} = \frac{-\lambda_2\gamma_{2,1}}{m\mu(\mu + \gamma_{2,1})} \leq 0.$$

Lemma 6.3.6 *Under Assumption 6.3.1, if the following relation on the power for a migration holds,*

$$0 \leq p_m \leq \left\lceil \frac{(r_2 + r_2\mathrm{P}_2)(2\alpha + \alpha\mathrm{P}_2^2) - 2\mathrm{K}\alpha + \mathrm{P}_2\beta}{m\mathrm{P}_2} \right\rceil, \tag{6.15}$$

then

$$f(\rho_1, \rho_2) - g(r_1, r_2) \leq 0.$$

This gives an upper bound on the power needed for a migration which reduces the global power consumption.

Proof. *Let* $f(\rho_1, \rho_2)$ *(resp.* $g(r_1, r_2)$*) be the power consumption function in the task migration model (resp. no task migration model). From Equation (6.9) and (6.11) and Assumption 6.3.1 we obtain:*

$$\left[\begin{array}{rcl} g(r_1, r_2) & = & 2\delta + (r_1 + r_2)\beta + \left[r_1^2 + r_2^2\right]\alpha, \\ f(\rho_1, \rho_2) & = & 2\delta + \left(\rho_1 + \rho_2\mathrm{P}_2\right)\beta + \left[\rho_1^2 + (\rho_2\mathrm{P}_2)^2\right]\alpha + mp_m\rho_2(1 - \mathrm{P}_2). \end{array} \right. \tag{6.16}$$

Using $P_2 = \frac{\rho_2}{r_2}$ *and* $r_1 + r_2 = \rho_1 + \rho_2 = K$ *(see Lemma (6.3.4)) then:*

$$\begin{aligned} f(\rho_1,\rho_2) - g(r_1,r_2) &= P_2(\rho_2 - r_2)\beta + \left(\frac{\rho_2^4 - r_2^4}{r_2^2}\right)\alpha \\ &+ (r_2 - \rho_2)\left[(2K - \rho_2 - r_2)\alpha + mp_m P_2\right]. \end{aligned}$$

After factorization:

$$\begin{aligned} f(\rho_1,\rho_2) - g(r_1,r_2) &= (r_2 - \rho_2) \\ &* \left[mp_m P_2 + (2K - \rho_2 - r_2)\alpha - P_2\beta - \frac{(\rho_2 + r_2)(\rho_2^2 + r_2^2)\alpha}{r_2^2}\right]. \end{aligned}$$

Remember that $\rho_2 \leq r_2$ *(see Lemma 6.3.5) and after some algebraic manipulations, the proof is complete.*

Remark that the best case is when $p_m = 0$ but it's not realistic, as migration of tasks costs power.

6.3.3 Optimization of power consumption

Next, we study the optimal solution of the equation describing power consumption for the model with tasks migrations (see Equation (6.9)). We know that $\rho_1 = K - \rho_2$ (see Lemma (6.3.4)), $P_2 = \frac{\rho_2}{r_2}$ and $P_1 = 1$ (see Lemma 6.3.3). Then we can express Equation (6.9) as a function of one variable ρ_2. Let $f()$ be this function:

$$f(\rho_2) = 2\delta + \left(K - \rho_2 + \rho_2 P_2\right)\beta + \left((K - \rho_2)^2 + (\rho_2 P_2)^2\right)\alpha + mp_m\rho_2(1 - P_2).$$

After some algebraic manipulations, we get the expression of $f()$ as a degree 4 polynomial:

$$f(\rho_2) = \left(\frac{\alpha}{r_2^2}\right)\rho_2^4 + \left(\frac{\alpha r_2 - mp_m + \beta}{r_2}\right)\rho_2^2 + (mp_m - \beta - 2\alpha K)\rho_2 + K^2\alpha + K\beta + 2\delta. \quad (6.17)$$

Let us now study the domain of definition (the possible values of ρ_2). The set of possible values for (ρ_1,ρ_2) comes from all the constraints we got:

$$\left[\begin{array}{lll} \rho 1 & \leq & 1, \\ \rho 1 & \geq & r_1 \geq 0, \\ \rho 2 & \geq & 0, \\ \rho 2 & \leq & r_2 \leq 1, \\ \rho 1 + \rho_2 & = & K = r_1 + r_2. \end{array}\right.$$

Let $\mathbb{S}$ be this set. It is a compact set of $\mathbb{R}^2$. Let $\mathbb{D}$ be its projection for the second component. By construction $\mathbb{D}$ is a compact of $\mathbb{R}$. Depending of the value of K, $\mathbb{D}$ can be:

$$\left[\begin{array}{ll} [0,K] & if\ 0 < K < 1, \\ [0,1] & if\ K = 1, \\ [K-1,1] & if\ K > 1. \end{array}\right.$$

Theorem 6.3.2 *As a polynomial function, f is continuous and it takes value from a compact set $\mathbb{D}$ to the real numbers. According to Weierstrass extreme value theorem, it has a minimal and a maximal values on the compact set. Therefore it exists a value of ρ_2 in the domain of definition which minimizes the power consumption.*

Lemma 6.3.7 *If the following condition on the parameters of the power function holds:*

$$mp_m - \beta - \alpha r_2 \leq \frac{6\alpha}{r_2}((\mathrm{K}-1)^+)^2, \tag{6.18}$$

then, f is convex on $\mathbb{D}$.

Proof. *The first and second derivative of f are:*

$$f'(\rho_2) = \left(\frac{4\alpha}{r_2^2}\right)\rho_2^3 + \left(\frac{2\alpha r_2 - 2mp_m + 2\beta}{r_2}\right)\rho_2 + \left(mp_m - \beta - 2\alpha\mathrm{K}\right),$$

and,

$$f''(\rho_2) = \left(\frac{12\alpha}{r_2^2}\right)\rho_2^2 + \left(\frac{2\alpha r_2 - 2mp_m + 2\beta}{r_2}\right),$$

while

$$f'''(\rho_2) = \left(\frac{24\alpha}{r_2^2}\right)\rho_2.$$

Function $f''()$ is a degree 2 polynomial which is increasing on $\mathbb{D}$ as $f'''()$ is positive on $\mathbb{R}^+$ and $\mathbb{D} \subset \mathbb{R}^+$. To prove that $f''(\rho_2) \geq 0$ on $\mathbb{D}$, we just have to compute $f''(\rho_2)$ for the smallest element of $\mathbb{D}$. We have two cases:

- *if $\mathrm{K} \leq 1$, the minimal element of $\mathbb{D}$ is 0. We obtain the following condition*

  $$mp_m - \beta - \alpha r_2 \leq 0,$$

  which is consistent with our claim, as $(\mathrm{K}-1)^+ = 0$ in that condition.

- *if $\mathrm{K} > 1$, the minimal element is $\mathrm{K}-1$ and we get:*

  $$mp_m - \beta - \alpha r_2 \leq \frac{6\alpha}{r_2}(\mathrm{K}-1)^2,$$

  which is equivalent to our claim as $(\mathrm{K}-1)^+ = \mathrm{K}-1$ when $\mathrm{K} > 1$.

And the proof is complete.

Let us state how we can obtain the optimal solution for the power consumption by looking at function $f'()$ and its zero. We look for the value of ρ_2 which satisfies the following equation:

$$\rho_2^3 + \left(\frac{(\alpha r_2 - mp_m + \beta)r_2}{2\alpha}\right)\rho_2 + \left(\frac{(mp_m - \beta - 2\alpha\mathrm{K})r_2^2}{4\alpha}\right) = 0\ , \tag{6.19}$$

Let x (resp. y) be the minimal (resp. maximal) value on $\mathbb{D}$: $if(k \leq 1)$ then x = 0 and y = K, else x = K - 1 and y = 1.

- If this Equation has no solution in $\mathbb{D}$ then:

$$\rho_{2,opt} = \rho_2^{'} \quad where \quad f(\rho_2^{'}) = min(f(x), f(y)).$$

- Otherwise, Let $\rho_{2,j} \; with \; j \in \{0,1,2\}$ (at most the three solutions are in $\mathbb{D}$) be solutions that satisfies $\rho_{2,j} \in \mathbb{D}$ and let define $\rho_{2,t}$ such as:

$$f(\rho_{2,t}) = min(f(\rho_{2,j})) \quad \forall j \in \{0,1,2\},$$

then:

$$\rho_{2,opt} = \rho_2^{'}$$

where

$$f(\rho_2^{'}) = min(f(x), f(y), f(\rho_{2,t})).$$

Lemma 6.3.8 *When* $K \leq 1$ *and* f *is convex, then the global minimum of the power function (Equation (6.17)) is unique.*

Proof. *Using Cardan formula, to solve Equation (6.19), we first compute* Δ*:*

$$\Delta = -\left[4\left(\frac{(\alpha r_2 - m p_m + \beta) r_2}{2\alpha}\right)^3 + 27\left(\frac{(m p_m - \beta - 2\alpha K) r_2^2}{4\alpha}\right)^2\right].$$

Theorem 6.3.2 assures that f *admits at least one global minimum on* $\mathbb{D}$*. Condition in Lemma 6.3.7 states that* f *is convex and that* $\Delta \leq 0$*.* f *is convex guaranties that global minimum is not on the reachable bounds of* $\mathbb{D}$ *and* $\Delta < 0$ *implies that f' admits one real solution. We conclude that the unique solution of* f' *is the global minimum of* f*.*

Now we can deduce $\gamma_{2,opt}$ the migrations rate that optimizes power consumption. So we have:

$$\gamma_{2,opt} = max\left(0, \frac{\lambda_2}{m\rho_{2,opt}} - \mu\right) \quad with \quad \rho_{2,opt} \in \mathbb{D}, \tag{6.20}$$

is the optimal solution of:

$$f(\gamma_{2,1}) = 2\delta + \left(K - \frac{\lambda_2}{m(\mu+\gamma_{2,1})} + \left(\frac{\lambda_2}{m(\mu+\gamma_{2,1})}\right)^2 r_2^{-1}\right)\beta \; + \left(\left(K - \frac{\lambda_2}{m(\mu+\gamma_{2,1})}\right)^2 + \left(\frac{\lambda_2}{m(\mu+\gamma_{2,1})}\right)^4 r_2^{-2}\right)\alpha \; +$$

$$m p_m \left(\frac{\lambda_2}{m(\mu+\gamma_{2,1})} - \left(\frac{\lambda_2}{m(\mu+\gamma_{2,1})}\right)^2 r_2^{-1}\right). \tag{6.21}$$

Notice that $\rho_{2,opt} = 0$ could be possible (in theory), since $\gamma_{2,1} \in [0, +\infty[$.

Corollary 6.3.2 *Equation (6.20) shows that if* $\frac{\lambda_2}{m\rho_{2,opt}} < \mu$ *then the migration of tasks will only increase power consumption of the data center. Therefore the optimal solution is to not perform the migration (*$\gamma_{2,opt} = 0$*) under the current power parameters. Otherwise, it means that an optimal solution that acutely reduce power consumption exists and its value is* $\frac{\lambda_2}{m\rho_{2,opt}} - \mu$*.*

6.3.4 Generalization to larger scale systems

In this section, we study the general case of migrations between servers. Each server $i \in [1,N]$ can serve a task (resp. Migrate the task to a server $j \in [1,N]$) with a probability P_i (resp. $1-P_i$). Let $\gamma_{i,j}$ be the migration rate from the physical server i to the physical server j.

$$P_i = \frac{\mu_i}{\mu_i + \sum_{j=1}^{N} \gamma_{i,j}}.$$

First, we notice that Lemma. 6.3.4 still holds for N servers.

Lemma 6.3.9 *When VMs are homogeneous, then the sum of the mean number of activated VMs in each model is a constant value (property of the conservation of the sum of the workload).*

$$\forall\ i\ \in [1,N]\ ,\quad \mu_i = \mu \quad \Rightarrow \quad m\sum_{i=1}^{N} \rho_i = m\sum_{i=1}^{N} r_i = mK \tag{6.22}$$

Proof. *Equation (6.4) for each server $i \in [1,N]$:*

$$\forall\ i, \qquad \rho_i = \frac{\lambda_i + \sum_{j=1}^{N} m\rho_j\gamma_{j,i}}{m(u + \sum_{j=1}^{N} \gamma_{i,j})}$$

$$\Rightarrow \quad \mu m\rho_i + m\rho_i \sum_{j=1}^{N} \gamma_{i,j} = \lambda_i + \sum_{j=1}^{N} m\rho_j\gamma_{j,i}$$

$$\Rightarrow \quad \rho_i = \frac{\lambda_i}{m\mu} + \frac{\sum_{j=1}^{N} m\rho_j\gamma_{j,i}}{m\mu} - \frac{\sum_{j=1}^{N} m\rho_i\gamma_{i,j}}{m\mu}$$

$$\Rightarrow \quad \sum_{i=1}^{N} \rho_i = \sum_{i=1}^{N} \frac{\lambda_i}{m\mu} + \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} \rho_j\gamma_{j,i} - \sum_{i=1}^{N}\sum_{j=1}^{N} \rho_i\gamma_{i,j}}{\mu}$$

$$\Rightarrow \quad \sum_{i=1}^{N} \rho_i = \sum_{i=1}^{N} r_i$$

Lemma 6.3.10 *Let $f(\rho_1,\rho_2,\ldots,\rho_N)$ (resp. $g(r_1,r_2,\ldots,r_N)$) the energetic function for the model with task migration (resp. model without migration of tasks). We propose a sufficient condition on $e_{m,i}$ the migration cost of the sending queues :*

$$f(\rho_1,\ldots,\rho_N) - g(r_1,\ldots,r_N) \le 0$$

$$\Rightarrow \sum_{i=1}^{N} (\rho_i P_i - r_i)\beta + \sum_{i=1}^{N} \left[(\rho_i P_i)^2 - r_i^2\right]\alpha + m\sum_{i=1}^{N} \rho_i e_{m,i}(1-P_i) \le 0$$

If : $\forall i,\ \ e_{m,i} = p_m$ then:

$$p_m \le \frac{\sum_{i=1}^{N} (r_i - \rho_i P_i)(\beta + r_i\alpha + \rho_i P_i\alpha)}{m\sum_{i=1}^{N} \rho_i(1-P_i)} \tag{6.23}$$

If the energetic cost per task migrated is the same for all physical servers, then the condition (Equation (6.23)) must be checked to reduce power consumption in the data center. Otherwise, in the Equation (6.24) we propose a sufficient condition on each physical server migration cost :

$$\forall i, \quad e_{m,i} \leq \frac{(r_i - \rho_i P_i)(\beta + r_i \alpha + \rho_i P_i \alpha)}{m \rho_i (1 - P_i)} \tag{6.24}$$

Next we propose two heuristics that improve data center consumption during task migrations for N physical servers. The heuristics are based on the optimization process detailed in section 6.3.

6.3.4.1 Heuristic 1

- We propose an iterative algorithm (Algorithm 6) that reduces power consumption step by step. At each iteration, we optimize the power consumption between the physical server that consumes the most power and the one that consumes the least.
- A physical server can be either a transmitter, a receiver or "not defined yet" during the algorithm.
- A sending server sends part of its load to a receiving server or to a server whose status has not yet been defined.
- The algorithm stops if the maximum load of the transmitters is lower than the minimum load of the receivers, or if the selected couple (Max_Transmitters, Min_Receivers) has already been processed in the last iterations.

6.3.4.2 Heuristic 2

- In this algorithm (Algorithm 7), at each iteration, the most consuming physical server distributes part of its load to a group of servers with a minimal load. This group includes servers whose load is lower than the total average load.
- The load is distributed in order to obtain the average load for each of the servers in the group (starting with those with low load in the group).
- The transmitter sends the load from the least loaded in the group to the most loaded.
- A server in the group does not receive tasks if the sender no longer has tasks to migrate.
- The algorithm stops when all the transmitters performed the migration.

Algorithm 6: Heuristic 1: Computing the power consumption in a data center with $N > 2$ physical servers

Input : p_m, γ, β, α and $r_i \quad \forall_i \in \{1,\ldots,N\}$.
Output: Pwr_{total} and $\rho_i \ \forall_i \in \{1,\ldots,N\}$.

Step 1) Initialization : $k = 0$, $\mathbb{S} = \emptyset$ and $\forall\ i \quad \rho_i = r_i$, computing of $Pwr_i(\rho_i, \gamma, \beta, \alpha)$ and $State_i = N$.
Step 2) Select $\rho_{i,max}$ (resp. $\rho_{i,min}$) the maximum (resp. minimum) servers consumption which state $State_i \in \{E, N\}$ (resp. $State_i \in \{R, N\}$).
Step 3) If $(S_{i,max}, S_{i,min}) \notin \mathbb{S}$, then perform the optimal migration between the server $S_{i,max}$ and $S_{i,min}$, and put $State_{i,max} = E$, $State_{i,min} = R$ and $\mathbb{S} = \mathbb{S} \cup (S_{i,max}, S_{i,min})$. Otherwise the algorithm stops.
Step 4) After performing the migration, we update $\rho_{i,max}$, $\rho_{i,min}$, $Pwr_{i,max}$, $Pwr_{i,min}$ and $Pwr_{totale} = \sum_{i=1}^{N} Pwr_i(\rho_i, \gamma, \beta, \alpha)$.
Step 5) Do $\rho_{i,max} = \rho_{i,max} P_{i,max}$ where $P_{i,max}$ is the probability of executing a task in the server $S_{i,max}$.
Step 6) Print (ρ_i, Pwr_i) for each server i in $\{1,\ldots,N\}$.
Step 7) If the maximal load of the transmitters is less than the minimal load of the receivers and that this couple (Max_Transmitters, Min_Receivers)$\notin \mathbb{S}$, then k++ and go to step 3). Otherwise, the algorithm stops.

Algorithm 7: Heuristic 2: Computing the power consumption in a data center with $N > 2$ physical servers

Input : p_m, γ, β, α, θ and $r_i \quad \forall_i \in \{1,\ldots,N\}$.
Output: Pwr_{total} and $\rho_i \ \forall_i \in \{1,\ldots,N\}$.

Step 1) Initialization : $k = 0$ and $\forall\ i \quad \rho_i = r_i$, computing of $Pwr_i\ (\rho_i, \gamma, \beta, \alpha)$ and let $Etat_i = N$.
Step 2) Computation of the average load $m = \sum_{i=1}^{N} \frac{\rho_i}{N}$.
Step 3) Do $State_i = E$ for each server S_i with $\rho_i > m + \epsilon$. Let T be the number of these servers.
Step 4) Do $\mathbb{S} = \emptyset$ and select $\rho_{i,max}$ the maximum's servers consumption with $State_i = E$ which has not performed the migration yet.
Step 5) Do $\mathbb{S} = \mathbb{S} \cup S_j$ for each server S_j which $\rho_j + \theta < m$ and $State_j \in \{N, R\}$.
Step 6) For all $S_j \in \mathbb{S}$, starting with low loaded servers ρ_j. If $\rho_{i,max} - (m - \rho_j - \theta) > 0$, then perform the migration of $m - \rho_j - \theta$ from $S_{i,max}$ to S_j, which means, put $\rho_{i,max} = \rho_{i,max} - (m - \rho_j - \epsilon)$ and $\rho_j = \rho_j + m - \rho_j - \epsilon$.
Step 7) Do $State_j = R$ for each server $S_j \in \mathbb{S}$ receiving tasks.
Step 8) Updating $Pwr_{i,max}$, $Pwr_j \ \forall S_j \in \mathbb{S}$ and Pwr_{totale}.
Step 9) Print (ρ_i, Pwr_i) for each server i in $\{1,\ldots,N\}$.
Step 10) The algorithm stops if $k < T$, otherwise k++ and go to step 3).

Remark 6.3.1 *In heuristics 1, for each pair (transmitter, receiver) we optimize the migration, i.e. if the transmitter is very loaded and the receiver is almost empty, the transmitter sends part of its load to the receiver. In heuristics 2, we considered that a transmitter distributes the load to be sent to a group of low receivers and not only to a single receiver.*

6.4 Numerical results

Let's consider the following power parameters: $\delta = 95W$, $\beta = 25W$, $\alpha = 100W$ so the maximal power that can be consumed by a physical server without migration is $220W$. These power consumption parameters are inspired from Taurus server [MAA16] consumption. We use the total power given by Equation (6.11) for the system without migration, while for the system with migrations we have also to consider the migration power Em given by Equation (6.8), so it results Equation (6.9).

6.4.1 A data center with two physical servers

We suppose that $m = 20$ VMs on each physical server so the CPU ratio of each VM is $CPU(vm) = \frac{1}{20} = 0.05$. Arrival rate in server1 (resp. server2) is $\lambda_1 = 2$ (resp. $\lambda_2 = 17$), and service rate is $\mu = 1$. So traffic intensity is $r_2 = 0.85$ and $r_1 = 0.1$. Migration of tasks will only be performed from server2 to server1 (see Assumption 6.3.1) so $\gamma_{1,2} = 0$. When $\gamma_{2,1} = 0$, then no migrations are performed in the system (see Corollary 6.3.1).

Taking $p_m = 3W$ and varying $\gamma_{2,1}$, we can see in Fig. 6.1 that data center power consumption (green curve, circle points) is a convex function since numerical parameters satisfies condition in Lemma 6.3.7. The behavior of black curve (rectangular points) and red curve (star points) clearly shows the effect of tasks migrations from server 2 to server 1, also shown in Fig. 6.2. Migration rate should be cleverly chosen in order to reduce data center consumption. Let's analyze optimal case: using Cardan formula to solve Equation (6.19) that becomes $\rho_2^3 + 0.2125\rho_2 - 0.2799 = 0$, we have $\Delta \leq 0$, then we obtain one solution $\rho_{2,opt} = 0.5305$ in $[0\ ,\ 0.95]$ so $\gamma_{2,opt} = 0.553$ which corresponds to an power consumption of $249.20W$ instead of an power consumption of $287W$ in the model without migrations. Then power gain is 13.16%. Notice that, in Fig. 6.2, the load of expected number of activated VMs in server2 (10.94) is not equal to the expected number of activated VMs in server1 (8.05), when the total power consumption is minimized.

In Fig. 6.3, we have calculated optimal power consumption for several value of p_m. We can see that the data center consumption increases with migration cost until some point ($p_m = 18$) where the migration is no more efficient, then $\forall\ \ p_m \geq 18$ optimal solution of Equation (6.21) is $\gamma_{2,opt} = 0$. The best case, but not realistic, is when the migration does not cost power then we achieve an power gain of 17.31% and the worst case is when the power is costly but in that case no migrations are performed so the power gain is 0% as shown in Fig. 6.3 and Fig. 6.4.

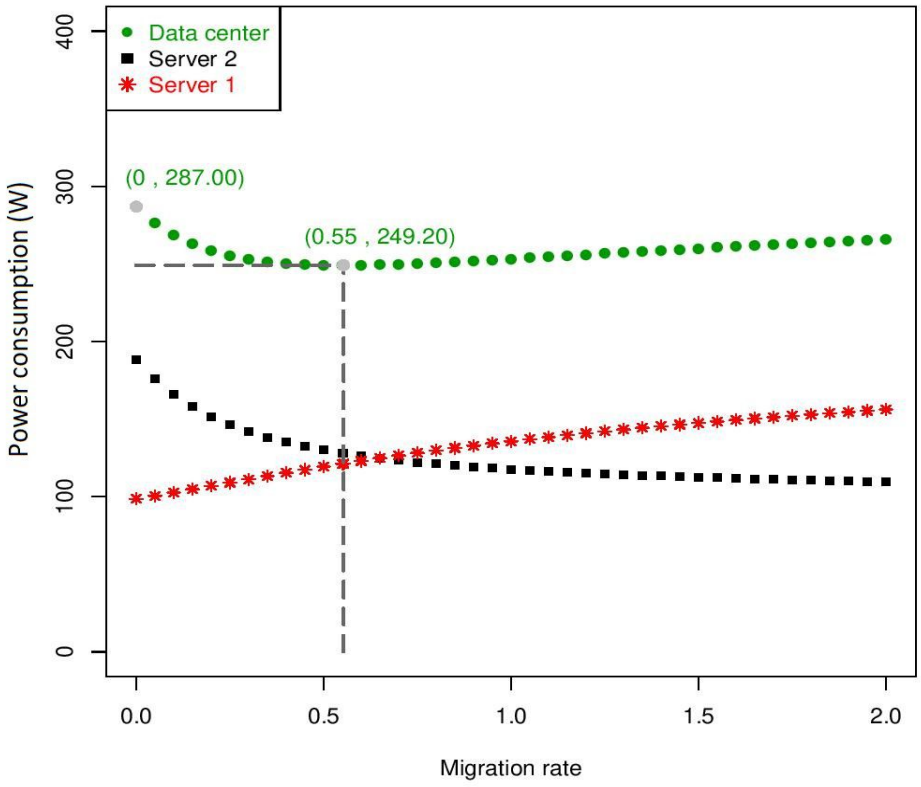


Figure 6.1: Power consumption under $\gamma_{2,1}$ variation

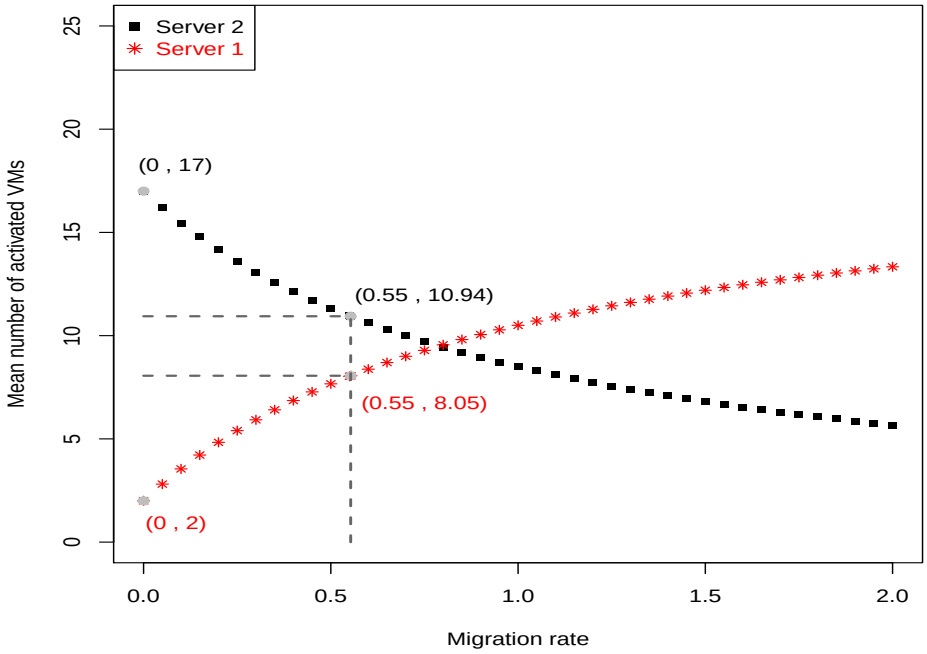


Figure 6.2: The mean number of ativated VMs under $\gamma_{2,1}$ variation

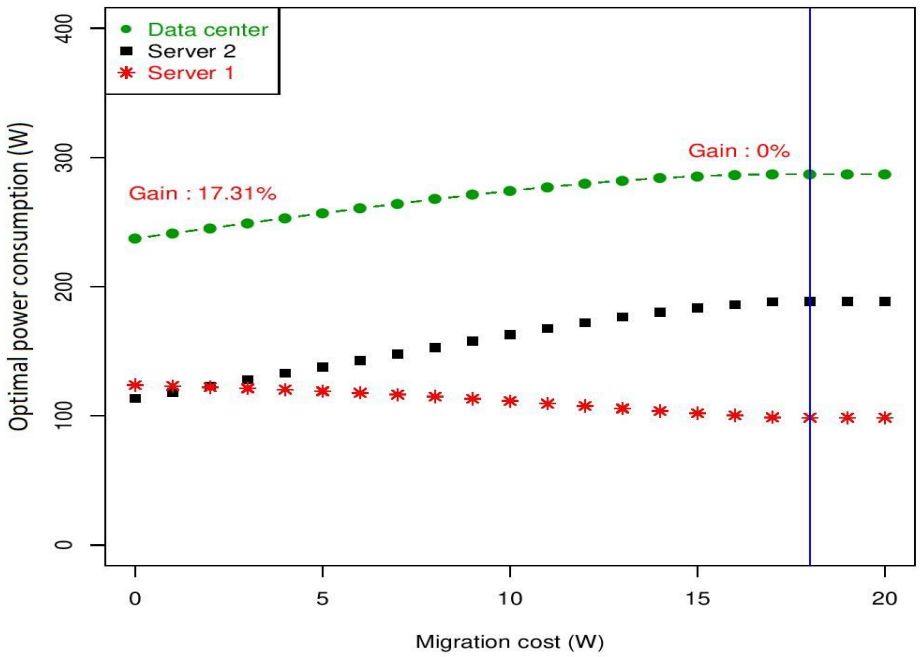


Figure 6.3: Power consumption in optimal case, under p_m variation

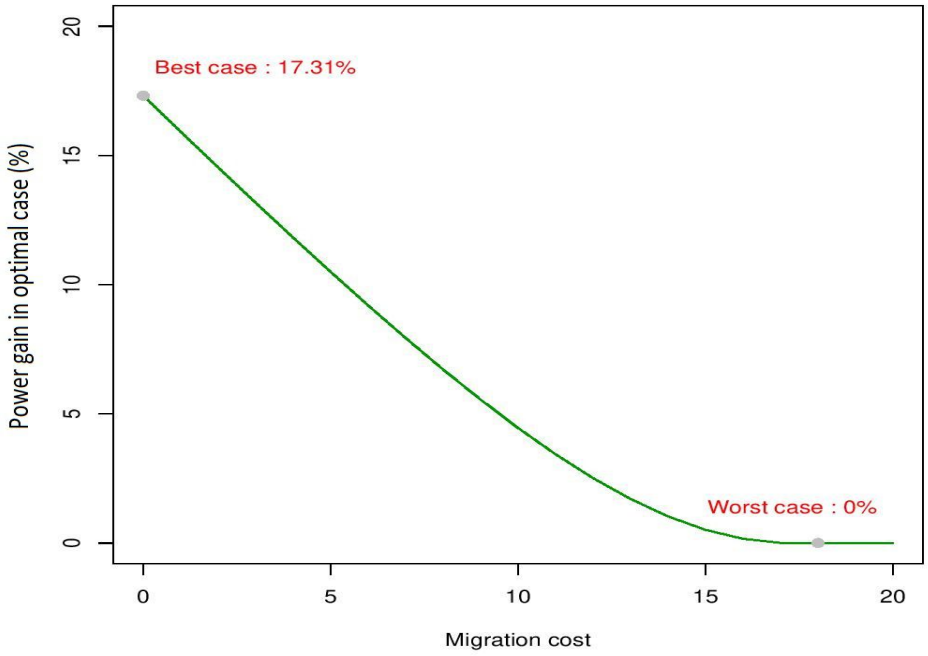


Figure 6.4: Power gain in optimal case, under p_m variation

6.4.2 A data center with N physical servers

Next we fix the parameters : $p_m = 1\text{W}$ and $\forall\ i \in [1 \ldots \text{N}]\quad \mu_i = 1$.

6.4.2.1 Experiment 1

We consider a data center with $\text{N} = 5$ physical servers and $m = 50$ VMs per physical server. The arrival rates are $\lambda = \{49, 1, 1, 1, 1\}$ respectively in server $\text{S}_1 \ldots \text{S}_5$.

	S1		**S2**		**S3**		**S4**		**S5**		$\mathbf{Pwr_{totale}}$
k = 0	0.98	215.54	0.02	95.54	0.02	95.54	0.02	95.54	0.02	95.54	597.70
k = 1	0.35	128.51	0.40	121.77	0.02	95.54	0.02	95.54	0.02	95.54	536.91
k = 2	0.14	104.87	0.40	121.77	0.15	101.02	0.02	95.54	0.02	95.54	518.75
k = 3	0.079	99.01	0.40	121.77	0.15	101.02	0.05	96.77	0.02	95.54	514.12
k = 4	0.071	97.49	0.40	121.77	0.15	101.02	0.05	96.77	0.024	95.65	512.72

Table 6.1: Experiment1, Heuristic1: Load and power consumption in each server during iterations 'k'. The power gain = 14.21%.

	S1		**S2**		**S3**		**S4**		**S5**		$\mathbf{Pwr_{totale}}$
k = 0	0.98	215.54	0.02	95.54	0.02	95.54	0.02	95.54	0.02	95.54	597.70
k = 1	0.25	144.05	0.202	104.13	0.202	104.13	0.202	104.13	0.202	104.13	560.57

Table 6.2: Experiment1, Heuristic2, $\epsilon = 10^{-2}$: Load and power consumption in each server during iterations 'k'. The power gain = 6.2%

The table 6.1 represents the power consumption and load in each server during heuristic 1. The $k = 0$ line represents the data center without migration. The migration process starts from $k = 1$ and in each line we have the load $\rho_i * \text{P}_i$ (the actual load in a S_i server after the optimization process, see Step 5) in Algorithm 8) and the power consumed by the corresponding server. We can see that the data center's power consumption decreases (column $\text{P}wr_{total}$) until it reaches an power gain of 11.34 %. The table 6.2 represents the actual load ρ_i in a server after migration (no optimization process is considered here) and the power consumed. The algorithm stops after an iteration since there is only one transmitter S_1. In the following example, we will increase the number of physical servers and transmitter servers to see the impact on heuristics 2.

6.4.2.2 Experiment 2

Now, we consider a data center with $N = 7$ physical servers, we keep $m = 50$ VMs per physical server. The arrival rates are $\lambda = \{49, 49, 49, 1, 1, 1, 1\}$ respectively in server $S_1 \ldots S_7$.

	k = 0		**k = 1**		**k = 2**		**k = 3**		**k = 4**		**k = 5**		**k = 6**	
S1	0.98	215.54	0.35	128.51	0.35	128.51	0.35	128.51	0.14	104.87	0.14	104.87	0.14	104.87
S2	0.98	215.54	0.98	215.54	0.35	128.51	0.35	128.51	0.35	128.51	0.20	107.46	0.20	107.46
S3	0.98	215.54	0.98	15.54	0.98	215.54	0.35	128.51	0.35	128.51	0.35	128.51	0.23	09.26
S4	0.02	95.54	0.40	121.77	0.40	121.77	0.40	121.77	0.40	121.77	0.40	121.77	0.40	121.77
S5	0.02	95.54	0.02	95.54	0.40	121.77	0.40	121.77	0.40	121.77	0.40	121.77	0.40	121.77
S6	0.02	95.54	0.02	95.54	0.02	95.54	0.40	121.77	0.40	121.77	0.40	121.77	0.40	121.77
S7	0.02	95.5	0.02	95.54	0.02	95.54	0.02	95.54	0.15	101.02	0.24	106.76	0.30	112.10
E	1028.78		967.99		907.20		846.42		828.26		812.95		799.03	

Table 6.3: Experiment2, Heuristic1: Load and power consumption in each server during iterations 'k'. The power gain = 22.33%

	k = 0		**k = 1**		**k = 2**		**k = 3**	
S1	0.98	215.54	0.39	149.53	0.39	149.53	0.39	149.53
S2	0.98	215.54	0.98	215.54	0.40	150.07	0.40	150.07
S3	0.98	215.54	0.98	215.54	0.98	215.54	0.54	159.77
S4	0.02	95.54	0.02	95.54	0.42	123.29	0.42	123.29
S5	0.02	95.54	0.02	95.54	0.19	103.81	0.42	123.29
S6	0.02	95.54	0.20	104.46	0.20	104.46	0.42	123.29
S7	0.02	95.54	0.42	123.29	0.42	123.29	0.42	123.29
E	1028.78		999.46		970.03		952.57	

Table 6.4: Experiment2, Heuristic2, $\epsilon = 10^{-2}$: Load and power consumption in each server during iterations 'k'. The power gain = 7.4%

We can see in the tables 6.1, 6.2, 6.3 and 6.4 that heuristic 1 gives better results. The power gain reaches **22.33%** in the table 6.3 which could be very significant in a huge data center. Of course the results depend on the initial load in the servers and

the power parameters. The power of heuristic1 is that it could admit that some data are actually migrating (see the optimization process in Section 6.3) and then cost only the cost of migration which is normally quite lower than the cost of service. In heuristics 2, the tasks are either in the transmitter server or in the receiver server and the migration cost is also included in the energetic function.

Part IV

Analytical analysis of EPN networks and G-networks

Chapter 7

State of the art

Contents

7.1 Energy packet networks

Energy Packet Networks (EPNs) were recently introduced by Gelenbe and his colleagues [Gel11; Gel12; Gel14a; GC15]. They are used to study interactions between IT devices consuming energy like sensors, cpu, storage systems and networking elements and the flow of intermittent sources of energy like batteries and solar or wind based generators.

Energy Packet Networks [EC16] model energy packets (EPs) as discrete units (a simplified version of the actual continuous flow of energy) of energy, e.g. X Joules or Watts, which is represented as arriving in "one chunk". An energy storage unit (ES) such as a battery is modelled as a queue of EPs that are waiting to be used. The ES is replenished by a flow of EPs from some external source including an energy harvesting unit, and it can be depleted both when energy is forwarded to a consumer, and through losses that represent leakage. The sources of power, the ESs and the consumers are interconnected by Power Switches, which dynamically connects the sources of power to the ESs, and the sources of power and ESs to the consumers. The consumers will request for EPs from either a power switch or an energy storage, and these requests will typically be intermittent, since they are a function of the work that these consumers accomplish with the energy. A typical example of such consumers are ICT systems which intermittently receive computational work to accomplish, and which in turn require energy to accomplish this work.

Some EP Networks (EPN in the following) are linked to the theory of G-networks (more details are in the next section) of queues and signals, which was introduced by the seminal papers by Gelenbe on networks of queues with positive and negative customers [Gel91]. Since then, many networks of queues with various signals (Triggers for one the first papers [Gel93a] and Adders for a recent one [FG17]) where proved to have a product form for the steady-state distribution of positive customers in the queues under some classical assumptions. From the product form of the steady-state distribution a closed form function can be derived for optimization purposes. In survey article [Ray19], authors have provided a bibliographic taxonomy on EPNs and their domain of application. The EPN model was thus proposed to formulate and optimize the dynamic behavior of energy consumption by the ICT-based infrastructures as Wireless Sensor Networks (WSNs), wired networks, servers, management of energy/Renewable energy in clouds/IOTs (more details in Fig. 7.1).

Figure 7.1: Bibliographic taxonomy of EPNs, [Ray19].

7.2 The evolution of G-networks

The theory of queues with signals (or G−networks) have received a considerable attention since the seminal papers on positive and negative customers [Gel91; Gel89] published by Gelenbe more than 20 years ago. The G−networks are motivated by neural networks where each queue represents a neuron, and excitation signals represents the movement of positive customers from a queue to another. Negative customers represents inhibition signals. In Fig. 7.2 a simple G−network of three queues is depicted. $P(i,j)$ (resp. $E(i,j)$) is the routing probability from a queue i to a queue j for positive (resp. negative) customers. d_i is the departure probability of a customer from a queue i. Let N be number of queues, then for all $i \in \{1 \ldots N\}$: $d_i + \sum_{j=1}^{N} P(i,j) + E(i,j) = 1$.

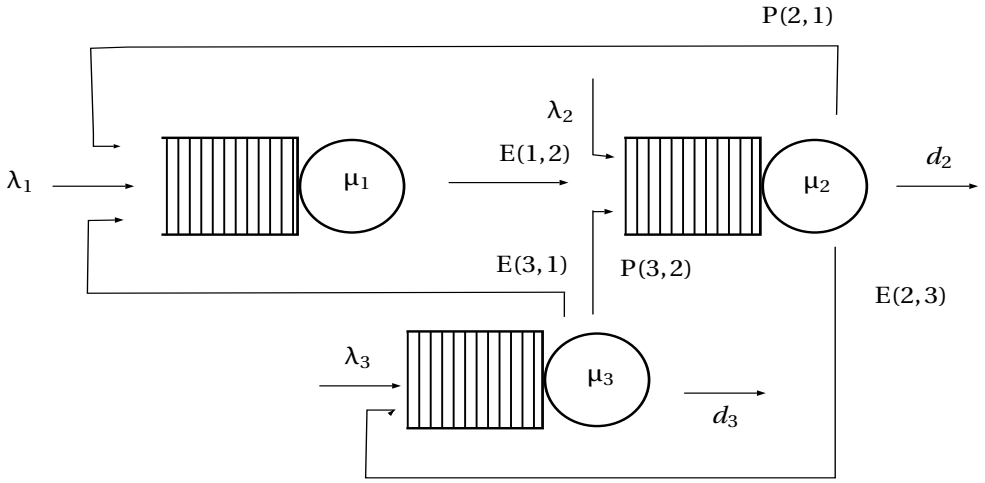


Figure 7.2: G−network with positive and negative customers

Traditional queueing networks models are used to represent contention among customers for a set of resources. Customers move from server to server where they wait for service according to a scheduling discipline. Apart this competition, they do not interact among themselves.

In a network of queues with signals, signals interact at their arrival into a queue with the queue or with customers already backlogged. Signals are never queued. At their arrival, they try to interact immediately. After their trial of interaction, they disappear irrespective of their failure or success or they can migrate to another queue. Furthermore, customers are allowed to change into signals at the completion of their service. This mechanism allows to model complex interaction between customers in several queues.

Despite this deep modification of the classical queueing network model, G-networks still preserve the product form property for the steady-state distribution of some Markovian queueing networks under some technical conditions on the processes involved (Poisson arrivals for signals and customers from the outside, exponential service times for customers, Markovian routing of customers and signals, open topology, independence).

The first type of signal was introduced as negative customers in [Gel91]. A negative customer deletes a positive customer at its arrival at a non empty queue.

Positive customers are usual customers which are queued and receive service or are deleted by negative customers. It must be clear that the results are more complex than Jackson's networks. The G-networks flow equations exhibit some uncommon properties: they are neither linear as in closed queueing networks nor contracting as in Jackson queueing networks. Therefore the existence of a solution had to be proved [GS92] by new techniques from the theory of fixed point equation and numerical algorithms had to be proved to solve the flow equations [Fou91; FQ06]. Many types of signal have been studied and they all lead to product form solution: triggers which redirect other customers among the queues [Gel93a], catastrophes which flush all the customers out of a queue and batches of deletion [Gel93b], resets [GF02]. Extensions with multiple classes of customers have also been derived [GL98].

G-networks had also motivated many new important results in the theory of queues. As negative customers lead to customer deletions, the original description of quasi-reversibility by arrivals and departures does not hold anymore and a new version had been proposed by Chao and his co-authors in [CMP99]. A different approach, based on Stochastic Process Algebra, was proposed by Harrison [Har03; Har04]. The main results (CAT and RCAT theorems and their extensions [BHM10; Har03; Har04]) give some sufficient conditions for product form stationary distributions. This technique clearly has a different range of applications as it allows to represent component based models which are much more general and more detailed than networks of queues.

Network of positive and negative customers were introduced to model neural networks where neurons exchange inhibitory and exciting signals [Gel94; GF99]. G-networks and Random Neural Networks were also used in the design of the learning process for Cognitive Packet Networks [GLX01] or application of the Random Neural Networks to quality of service [MRV04] or to model the interaction between energy and the Data plane in telecom networks [Gel14b; GC16]. Currently there are several hundred references devoted to the subject and two books [GM10; CMP99] provide insight into some of the research issues, developments and applications in the area of networks of queues with customers and signals.

Most of single class G−networks have a product-form [MAR16] which is based on the product of the marginal geometric distributions of the single queues. In Table 7.1 we present a chronological listing about G−networks. The analytical methods involved depend on the nature and complexity of the G−network, and the existence of a product-form for the steady-state distribution are subject to assumptions. A standard approach consists in guessing the expression of the equilibrium distribution and verifying that it satisfies the system of global balance equations (GBEs) in general this method does not require introducing formalism to specify the synchronizations [JF06; HM14b] of the stations of the queueing network. In Chapter 9 we propose a product form solution for a G−network with processor sharing and catastrophes signals, our analytical approach is based on guessing/verifying the GBEs. Other approaches based on a description formalism of the network are proposed in the literature: " RCAT " Reversed Compound Agent Theorem introduced by Harrison [HM14b; Har03] which rely on the verification of some sufficient conditions for the product-form, "PEPA " Performance Evaluation Process Algebra introduced by Hillston in [Hil94], " PITs " Propagation of Instantaneous Transitions for synchro-

nizations in G−networks [HM14b].

Year	Works	Characteristics.
1989, 1991	Gelenbe [Gel91; Gel89]	First apparition of G−networks, in those works Gelenbe proposed a product form solution for networks with positive and negative customers.
1993	[Gel93c]	G−networks with triggered customer movement.
1993	[Gel93a]	G−networks with instantaneous customer movement.
1993	[Gel93b]	G−networks witch batch removal.
1994	[Gel94]	G−networks with unifying a model for queuing networks and neural networks.
1995, 1999	[JLF95; XMM99]	G−networks with catastrophe's signals.
1996, 1998	[FGS96; GL98]	G−networks with multiple classes of signals and positive/negative customers.
2000	[JLF00; JLD00]	G−networks with multiple class and iterated (resp. list-oriented) probabilistic deleting mechanism.
2002, 2004, 2007	[EJ02; PG04; Fou07]	G−networks with resets.
2006	[JF06]	G−networks with synchronized partial flushing.
2014	[HM14b]	G−networks with multi-way (chains of pairwise [Kel87; Mun72]) synchronisations.

Table 7.1: Related G−networks models.

Below we recall some definitions related to networks topology. We are using these definitions in the following chapters in order to reflect the features of the network in question. By "node" we refer to: a DP-queue for EPNs (Chapter 8), or a PS queue when dealing with G-networks (Chapter 9), or .

Definition 7.2.1 *Let* $\mathbf{A}$ *be a binary operator which represents a network's (G-network or EPN network) connectivity . We note by* $i\mathbf{A}j$ *the existence of a directed path from node* i *to node* j*.*

Definition 7.2.2 *We use '0' to indicate an external source/destination. A network is open iff each node i verifies* $i\mathrm{A}0$ *and* $0\mathrm{A}i$*.*

Definition 7.2.3 *Let* $\mathrm{P}(i,j)$ *be the routing matrix of usual costumers when dealing with G-networks, or routing matrix of data packets when dealing with EPNs. Then, a network with routing matrix* P *is connected if for all couple of nodes* (i,j)*,* $i\mathrm{A}j$ *or* $j\mathrm{A}i$ *is verified. Hence, we say that* P *is irreducible.*

Remark 7.2.1 *The topology reported in Definition 7.2.3 is not the only topology for interconnected G-networks, the queues can also be connected through signals.*

Chapter 8

Energy Packet Networks with general service time distribution

Contents

8.1 Introduction

In the literature, product form results were obtained under classical assumptions: Poisson arrivals of packets (both EP and data), exponential duration for service and leakage, independence, Markovian routing. Other stochastic processes, like Interrupted Poisson Process [Fou20], have been considered to model energy packet generation variation during a day. Here, we extend the results to general service time distribution. We assume that the service times of the Energy Packets which trigger the data processing follow a Coxian distribution. Since all non-negative distributions can be arbitrarily closely approximated by Coxian distributions [Bar76], we can use them to model EP networks with general service times for both the energy consumption of energy packets and data packets processing

The technical part of this work is as follows. In section 8.2, we describe the EPN model. We give the proof of the product form steady-state distribution. We also study the existence of a solution to the flow equations. In Section 8.3, we focus on performance evaluation and energy losses rate. Finally, in section 8.4, we show how to optimize a sensor network with a solar panels harvesting capacity and we present two methods to distribute the panels among the stations to optimize the average delay. This illustrates one of the main advantages of EP networks models. Based on its closed form solution, it is possible to conduct an optimization of the systems based on some utility function like mean response time or loss rates [GC15; FMB16; GZ19; GA18].

8.2 Model Description and Markov chain analysis

We consider a distributed system (as a fog [DF19] or a sensor system) consuming energy provided by intermittent sources. It is represented by an EPN (see Fig. 8.1) which is an open network with N cells, where each of the cells is represented by one queue that stores DPs (Data Packets) and one battery that stores EPs (Energy Packets). Both EP and DP queues have an infinite capacity. DPs arrive to each cell i following a Poisson distribution with parameter λ_i, and EPs arrive to cell i following also a Poisson distribution with rate $\alpha_i > 0$. We suppose that queuing discipline is Processor Sharing, and the service is applied on EP queues and follows the Cox probability distribution law with K phases (see Fig. 8.2). Each phase $n \in \{1,\dots,K\}$ of cell $i \in \{1,\dots,N\}$ generates a transition with an exponential distribution with parameter $\mu_{i,n} > 0$. At each phase n there is a probability $p_{i,n}$ that the corresponding exponential transition occurred, and a probability $1 - p_{i,n}$ that the process ends. Also note that $p_{i,K} = 0$ and we suppose that $0 \leq p_{i,n} \leq 1$ for all $n \in \{1,\dots,K-1\}$. The process could reach the last phase K that generates the last transition $\mu_{i,k}$. We consider that there are energy leakages with exponential times and only occur in the first phase of the Cox process. Let $\gamma_i > 0$ be the leakage rate of one EP in the battery of cell i.

The service takes place in EP queues, and is described as follows for the cell i and at each phase n $(0 \leq n < K)$:

- With rate $\mu_{i,n}(1 - p_{i,n})$ an EP packet is consumed and a data packet is sent;

- With rate $\mu_{i,n} p_{i,n}$ the EP service continues, to go to the next phase.

Note that the triggered data packet either goes outside (with probability d_i) or is routed to another cell j $(1 \leq j \leq \mathrm{N})$ with probability $\mathrm{P}(i,j)$ of a DP. Hence, for all i $(1 \leq i \leq \mathrm{N})$:

$$d_i + \sum_{j=1}^{\mathrm{N}} \mathrm{P}(i,j) = 1. \tag{8.1}$$

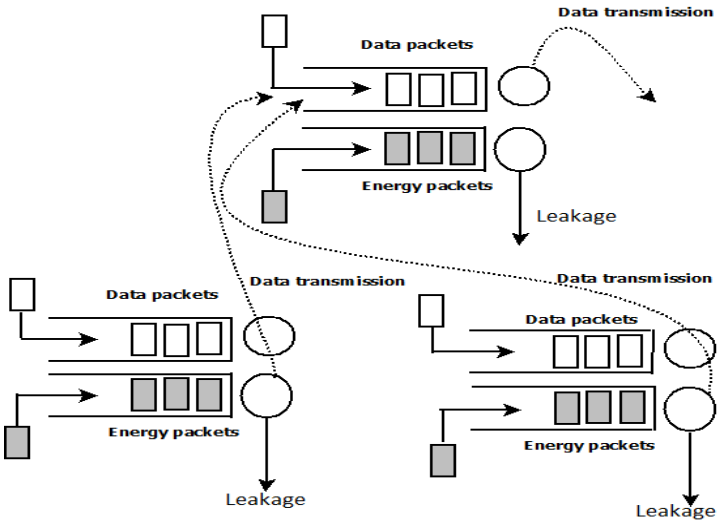


Figure 8.1: An EPN network with three cells.

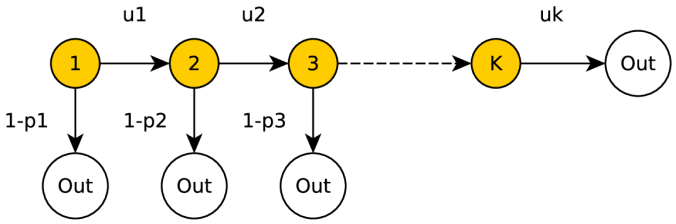


Figure 8.2: Cox process with K phases.

We consider an EPN where the EPs initiates the transfer. This means that, upon service in cell i, an EP is sent from the battery to the DP queue of the same cell. If the DP queue is empty, the energy packet is lost. If no EPs are present in the battery of a cell then DPs are blocked in the queue until the next external arrival of an EP. This model represents two important features of the model: first without energy, the data packets are not served, and second even if there is no data packets in the server, there is still some energy consumption.

8.2.1 Markov chain analysis

In this section we propose to analyse the system with a Markov chain model. For each cell i we have the following components:

- X_i is the number of data packets at the DP queue of cell i;
- $Y_{i,n}$ is the number of energy packets in phase n at the battery of cell i.

Let us note by $\mathbf{X} = (X_1, \ldots, X_N)$ and $\mathbf{Y_n} = (Y_{1,n}, \ldots, Y_{N,n})$. Under the assumptions about the arrivals and the services, $\{(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})_t, t \geq 0\}$ is a continuous-time Markov chain where:

- on $\mathbf{X}$ component, transitions are due to (i) external arrivals of DPs, (ii) service departure of a DP in phase n ($n \in \{1, \ldots, K\}$) EP, and (iii) transit of DPs after routing between the cells;
- on $\mathbf{Y_1}$ component, transitions are caused by (i) external arrivals of EPs, (ii) Leakage of an EP, (iii) transformation to phase 2 of an EP after a service, and (iiii) the triggering (departure/routing) of a DP after a service;
- on $\mathbf{Y_n}$ for all $2 \leq n \leq K-1$ components, transitions are due to (i) arrivals of EPs by transformation from phase $n-1$, (ii) the transformation of an EP to phase $n+1$ after a service, and (iii) the triggering (departure/routing) of a DP after a service.
- on $\mathbf{Y_K}$ component, transitions are (i) arrivals of EPs by transformation from phase $K-1$, and (ii) the triggering of a DP (departure/routing) after a service.

In the next theorem, we prove that the Markov chain $\{(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})_t,\ t \geq 0\}$ has a product form steady-state distribution if the flow equations have a solution which satisfies the stability constraints. Let us first introduce the flow equations, for all $i \in \{1, \ldots, N\}$:

$$\beta_{i,1} = \frac{\alpha_i}{\gamma_i + \mu_{i,1}}\ ,\ \beta_{i,n} = \Big(\frac{p_{i,n-1}\mu_{i,n-1}}{\mu_{i,n}}\Big)\beta_{i,n-1} \tag{8.2}$$

which is equivalent to

$$\beta_{i,1} = \frac{\alpha_i}{\gamma_i + \mu_{i,1}}\ ,\ \forall n \in \{2, \ldots, K\}\quad \beta_{i,n} = \frac{\mu_{i,1}\beta_{i,1}}{\mu_{i,n}} \prod_{r=1}^{n-1} p_{i,r}.$$

Also we have

$$\rho_i = \frac{\lambda_i + \sum_{n=1}^{K}\sum_{j=1}^{N}(1 - p_{j,n})\mu_{j,n}\rho_j\beta_{j,n}P(j,i)}{\sum_{n=1}^{K}(1 - p_{i,n})\mu_{i,n}\beta_{i,n}}. \tag{8.3}$$

Lemma 8.2.1 *Energy packets are cell unique contrary to DP that are routed to other cells. So the equation flow of EPs in each cell i is*

$$\gamma_i\beta_{i,1} + \sum_{n=1}^{K}(1 - p_{i,n})\mu_{i,n}\beta_{i,n} = \alpha_i. \tag{8.4}$$

Where the l.h.s of the equation represents the total rate for the departure of an EP from a cell i and the r.h.s stands for the arrivals rate of an EP to cell i.

Proof. *By developing the second term of the l.h.s of Equation (8.4), we have*

$$\begin{aligned}
&\gamma_i\beta_{i,1} + \sum_{n=1}^{K}(1-p_{i,n})\mu_{i,n}\beta_{i,n} = \gamma_i\beta_{i,1} + \mu_{i,1}\beta_{i,1} \\
&-p_{i,1}\mu_{i,1}\beta_{i,1} + \mu_{i,2}\beta_{i,2} - p_{i,2}\mu_{i,2}\beta_{i,2} + \cdots + \\
&\mu_{i,K-1}\beta_{i,K-1} - p_{i,K-1}\mu_{i,K-1}\beta_{i,K-1} + \mu_{i,K}\beta_{i,K} \\
&-p_{i,K}\mu_{i,K}\beta_{i,K}.
\end{aligned}$$

By substituting using Equation (8.2) and using that $p_{i,K} = 0$

$$\begin{aligned}
&\gamma_i\beta_{i,1} + \sum_{n=1}^{K}(1-p_{i,n})\mu_{i,n}\beta_{i,n} = \beta_{i,1}(\gamma_i + \mu_{i,1}) \\
&-p_{i,1}\mu_{i,1}\beta_{i,1} + p_{i,1}\mu_{i,1}\beta_{i,1} - p_{i,2}\mu_{i,2}\beta_{i,2} + \cdots + \\
&p_{i,K-2}\mu_{i,K-2}\beta_{i,K-2} - p_{i,K-1}\mu_{i,K-1}\beta_{i,K-1} + \\
&p_{i,K-1}\mu_{i,K-1}\beta_{i,K-1}.
\end{aligned}$$

All r.h.s terms of the last equation are eliminated except the first one, so

$$\gamma_i\beta_{i,1} + \sum_{n=1}^{K}(1-p_{i,n})\mu_{i,n}\beta_{i,n} = \beta_{i,1}(\gamma_i + \mu_{i,1}) = \alpha_i.$$

The proof is complete.

8.2.2 Product form of the EPN network

Theorem 8.2.1 *Assume that Markov chain* $(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})_t$ *is ergodic. If the flow equations (8.2) and (8.3) have a fixed point solution* $(\rho_i, \beta_{i,1}, \ldots, \beta_{i,K})$ *such that* $\rho_i < 1$ *and* $\sum_{n=1}^{K}\beta_{i,n} < 1$ *, then the steady-state distribution is*

$$\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K}) = G\prod_{i=1}^{N}(\rho_i)^{X_i}||Y_i||!\prod_{n=1}^{K}\frac{(\beta_{i,n})^{Y_{i,n}}}{Y_{i,n}!} \tag{8.5}$$

where $||Y_i|| = \sum_{n=1}^{K} Y_{i,n}$*. See Lemma 8.2.2 for the value of* G*.*

Proof. *The proof of this theorem is based on the analysis of the global balance equation. First let us introduce some notations. As usual* $\mathbf{e_i}$ *will denote a vector whose components are all 0 except component with index i which is 1. Moreover* 1_c *be the step function equal to 1 when condition C is true and 0 otherwise. Let us now write the Chapman-Kolmogorov equation at steady-state:*

$$\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})\left[\sum_{i=1}^{N}\left(\lambda_i + \alpha_i + \gamma_i\frac{Y_{i,1}}{||Y_i||}1_{||Y_i||>0} + \sum_{n=1}^{K}\mu_{i,n}\frac{Y_{i,n}}{||Y_i||}1_{||Y_i||>0}\right)\right] \tag{8.6}$$

$$
\begin{aligned}
&= \sum_{i=1}^{N} \lambda_i \pi(\mathbf{X} - \mathbf{e_i}, \mathbf{Y_1}, \ldots, \mathbf{Y_K}) 1_{X_i>0} && [1]\\
&+ \sum_{i=1}^{N} \alpha_i \pi(\mathbf{X}, \mathbf{Y_1} - \mathbf{e_i}, \mathbf{Y_2}, \ldots, \mathbf{Y_k}) 1_{Y_{i,1}>0} && [2]\\
&+ \sum_{i=1}^{N} \gamma_i \frac{Y_{i,1}+1}{||Y_i||+1} \pi(\mathbf{X}, \mathbf{Y_1} + \mathbf{e_i}, \mathbf{Y_2}, \ldots, \mathbf{Y_k}) && [3]\\
&+ \sum_{n=1}^{K-1} \sum_{i=1}^{N} p_{i,n} \mu_{i,n} \frac{Y_{i,n}+1}{||Y_i||} \pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_n} + \mathbf{e_i},\\
&\mathbf{Y_{n+1}} - \mathbf{e_i}, \ldots, \mathbf{Y_K}) 1_{Y_{i,n+1}>0} && [4]\\
&+ \sum_{n=1}^{K} \sum_{i=1}^{N} (1 - p_{i,n}) d_i \mu_{i,n} \frac{Y_{i,n}+1}{||Y_i||+1} \pi(\mathbf{X} + \mathbf{e_i}, \mathbf{Y_1}, \ldots,\\
&\mathbf{Y_n} + \mathbf{e_i}, \ldots, \mathbf{Y_K}) && [5]\\
&+ \sum_{n=1}^{K} \sum_{i=1}^{N} \sum_{j=1}^{N} (1 - p_{i,n}) \mu_{i,n} P(i,j) \frac{Y_{i,n}+1}{||Y_i||+1}\\
&* \pi(\mathbf{X} + \mathbf{e_i} - \mathbf{e_j}, \mathbf{Y_1}, \ldots, \mathbf{Y_n} + \mathbf{e_i}, \ldots, \mathbf{Y_K}) 1_{X_j>0} && [6]\\
&+ \sum_{n=1}^{K} \sum_{i=1}^{N} (1 - p_{i,n}) \mu_{i,n} \frac{Y_{i,n}+1}{||Y_i||+1} \pi(\mathbf{X}, \mathbf{Y_1}, \ldots,\\
&\mathbf{Y_n} + \mathbf{e_i}, \ldots, \mathbf{Y_K}) 1_{X_i=0}. && [7]
\end{aligned}
$$

Let's describe the right-hand side of the balance Equation (8.6). Term [1] (resp. Term [2]) represents an external arrival of a DP (resp. EP) in the DP (resp. EP) queue of the cell i. Term [3] represents the leakage of an EP from the battery of cell i. Term [4] represents the transformation phase of a DP from phase each phase n ($\forall\ n \in \{1, \ldots, K-1\}$) to phase $n+1$ after a service in cell i. Term [5] (resp. Term [6]) is the departure (resp. routing transitions from a cell i to a cell j) of a DP after a service in each phase n, the triggering EP also vanishes. The last term describes the consumption of an EP while the DP queue is empty.

We divide both sides of Equation (8.6) by $\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})$ and take into account the product form of the solution and the simplifications

$$
\begin{aligned}
\frac{\pi(\mathbf{X}-\mathbf{e_i}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})}{\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})} &= \frac{1}{\rho_i},\\
\frac{\pi(\mathbf{X}, \mathbf{Y_1}-\mathbf{e_i}, \mathbf{Y_2}, \ldots, \mathbf{Y_k})}{\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})} &= \frac{1}{\beta_{i,1}} \frac{Y_{i,1}}{||Y_i||},\\
\frac{\pi(\mathbf{X}, \mathbf{Y_1}+\mathbf{e_i}, \mathbf{Y_2}, \ldots, \mathbf{Y_k})}{\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})} &= \beta_{i,1} \frac{||Y_i||+1}{1+Y_{i,1}},\\
\frac{\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_n}+\mathbf{e_i}, \mathbf{Y_{n+1}}-\mathbf{e_i}, \ldots, \mathbf{Y_K})}{\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_k})} &= \frac{\beta_{i,n}}{\beta_{i,n+1}} \left(\frac{Y_{i,n+1}}{1+Y_{i,n}} \right),\\
\frac{\pi(\mathbf{X}+\mathbf{e_i}, \mathbf{Y_1}, \ldots, \mathbf{Y_n}+\mathbf{e_i}, \ldots, \mathbf{Y_K})}{\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})} &= \rho_i \beta_{i,n} \frac{||Y_i||+1}{1+Y_{i,n}},\\
\frac{\pi(\mathbf{X}+\mathbf{e_i}-\mathbf{e_j}, \mathbf{Y_1}, \ldots, \mathbf{Y_n}+\mathbf{e_i}, \ldots, \mathbf{Y_K})}{\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})} &= \frac{\rho_i \beta_{i,n}}{\rho_j} \frac{||Y_i||+1}{1+Y_{i,n}},\\
\frac{\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_n}+\mathbf{e_i}, \ldots, \mathbf{Y_K})}{\pi(\mathbf{X}, \mathbf{Y_1}, \ldots, \mathbf{Y_K})} &= \beta_{i,n} \frac{||Y_i||+1}{1+Y_{i,n}}.
\end{aligned}
\tag{8.7}
$$

We obtain:

$$
\sum_{i=1}^{N} \left(\lambda_i + \alpha_i + \gamma_i \frac{Y_{i,1}}{||Y_i||} 1_{||Y_i||>0} + \sum_{n=1}^{K} \mu_{i,n} \frac{Y_{i,n}}{||Y_i||} 1_{||Y_i||>0} \right) \tag{8.8}
$$

$$
\begin{aligned}
&= \sum_{i=1}^{N} \frac{\lambda_i}{\rho_i} 1_{X_i>0} && [1]\\
&+ \sum_{i=1}^{N} \frac{\alpha_i}{\beta_{i,1}} \frac{Y_{i,1}}{||Y_i||} 1_{||Y_i||>0} && [2]\\
&+ \sum_{i=1}^{N} \gamma_i \beta_{i,1} && [3]\\
&+ \sum_{n=1}^{K-1} \sum_{i=1}^{N} \frac{p_{i,n}\mu_{i,n}\beta_{i,n}}{\beta_{i,n+1}} \frac{Y_{i,n+1}}{||Y_i||} 1_{||Y_i||>0} && [4]\\
&+ \sum_{n=1}^{K} \sum_{i=1}^{N} (1-p_{i,n}) d_i \mu_{i,n} \rho_i \beta_{i,n} && [5]\\
&+ \sum_{n=1}^{K} \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{(1-p_{i,n})\mu_{i,n}\rho_i\beta_{i,n}P(i,j)}{\rho_j} 1_{X_j>0} && [6]\\
&+ \sum_{n=1}^{K} \sum_{i=1}^{N} (1-p_{i,n}) \mu_{i,n} \beta_{i,n} 1_{X_i=0}. && [7]
\end{aligned}
$$

In Term [4] of the r.h.s of Equation (8.8), we apply an index change on n. In Term [7], we substitute $1_{X_i}=0$ with $1-1_{X_i>0}$ and we move the negative terms in the l.h.s. To get:

$$
\begin{aligned}
&\sum_{i=1}^{N} \left(\lambda_i + \alpha_i + \left(\gamma_i \frac{Y_{i,1}}{||Y_i||} + \sum_{n=1}^{K} \mu_{i,n} \frac{Y_{i,n}}{||Y_i||} \right) 1_{||Y_i||>0} \right.\\
&\left. + \sum_{n=1}^{K} (1-p_{i,n}) \mu_{i,n} \beta_{i,n} 1_{X_i>0} \right) = \sum_{i=1}^{N} \frac{\lambda_i}{\rho_i} 1_{X_i>0}\\
&+ \sum_{i=1}^{N} \gamma_i \beta_{i,1} + \sum_{i=1}^{N} \frac{\alpha_i}{\beta_{i,1}} \frac{Y_{i,1}}{||Y_i||} 1_{||Y_i||>0}\\
&+ \sum_{n=2}^{K} \sum_{i=1}^{N} \frac{p_{i,n-1}\mu_{i,n-1}\beta_{i,n-1}}{\beta_{i,n}} \frac{Y_{i,n}}{||Y_i||} 1_{||Y_i||>0}\\
&+ \sum_{n=1}^{K} \sum_{i=1}^{N} (1-p_{i,n}) d_i \mu_{i,n} \rho_i \beta_{i,n}\\
&+ \sum_{n=1}^{K} \sum_{i=1}^{N} (1-p_{i,n}) \mu_{i,n} \beta_{i,n}\\
&+ \sum_{n=1}^{K} \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{(1-p_{i,n})\mu_{i,n}\rho_i\beta_{i,n}P(i,j)}{\rho_j} 1_{X_j>0}.
\end{aligned}
\tag{8.9}
$$

We decompose Equation (8.9) into three equation systems based on the step functions they expose:

$$
\left[
\begin{aligned}
&\sum_{i=1}^{N} \sum_{n=1}^{K} (1-p_{i,n}) \mu_{i,n} \beta_{i,n} 1_{X_i>0} = \sum_{i=1}^{N} \left(\frac{\lambda_i}{\rho_i} \right.\\
&\left. + \sum_{n=1}^{K} \sum_{j=1}^{N} (1-p_{i,n}) \mu_{j,n} P(j,i) \frac{\rho_j \beta_{j,n}}{\rho_i} \right) 1_{X_i>0}
\end{aligned}
\right.
\tag{8.10}
$$

and

$$
\left[
\begin{aligned}
&\left[\sum_{i=1}^{N} \gamma_i \left(\frac{Y_{i,1}}{||Y_i||} \right) + \sum_{i=1}^{N} \mu_{i,1} \left(\frac{Y_{i,1}}{||Y_i||} \right) + \right.\\
&\left. \sum_{n=2}^{K} \sum_{i=1}^{N} \mu_{i,n} \left(\frac{Y_{i,n}}{||Y_i||} \right) \right] 1_{||Y_i||>0} = \left[\sum_{i=1}^{N} \frac{\alpha_i}{\beta_{i,1}} \left(\frac{Y_{i,1}}{||Y_i||} \right) \right.\\
&\left. + \sum_{n=2}^{K} \sum_{i=1}^{N} \frac{p_{i,n-1}\mu_{i,n-1}\beta_{i,n-1}}{\beta_{i,n}} \left(\frac{Y_{i,n}}{||Y_i||} \right) \right] 1_{||Y_i||>0}
\end{aligned}
\right.
\tag{8.11}
$$

and

$$
\left[
\begin{aligned}
&\sum_{i=1}^{N} (\lambda_i + \alpha_i) = \sum_{n=1}^{K} \sum_{i=1}^{N} (1-p_{i,n}) d_i \mu_{i,n} \rho_i \beta_{i,n}\\
&+ \sum_{n=1}^{K} \sum_{i=1}^{N} (1-p_{i,n}) \mu_{i,n} \beta_{i,n} + \sum_{i=1}^{N} \gamma_i \beta_{i,1}.
\end{aligned}
\right.
\tag{8.12}
$$

The values of $(\rho_i, \beta_{i,1}, \ldots, \beta_{i,K})$ we are looking for have to satisfy the three equations (8.10),(8.11) and (8.12).

First by the factorisation of ρ_i *in Equation* (8.10) *we get, for all* i

$$\rho_i = \frac{\lambda_i + \sum_{n=1}^{K}\sum_{j=1}^{N}(1-p_{j,n})\mu_{j,n}\rho_j\beta_{j,n}P(j,i)}{\sum_{n=1}^{K}(1-p_{i,n})\mu_{i,n}\beta_{i,n}}. \tag{8.13}$$

Next we decompose Equation (8.11)*, into two equations. And we verify that two equations are satisfied. This decomposition:*

$$\begin{cases} \sum_{i=1}^{N}(\gamma_i+\mu_{i,1})\left(\frac{Y_{i,1}}{||Y_i||}\right)1_{||Y_i||>0} = \sum_{i=1}^{N}\frac{\alpha_i}{\beta_{i,1}}\left(\frac{Y_{i,1}}{||Y_i||}\right)1_{||Y_i||>0}, \\ \textit{and} \\ \sum_{n=2}^{K}\sum_{i=1}^{N}\mu_{i,n}\left(\frac{Y_{i,n}}{||Y_i||}\right)1_{||Y_i||>0} = \sum_{n=2}^{K}\sum_{i=1}^{N}\frac{p_{i,n-1}\mu_{i,n-1}\beta_{i,n-1}}{\beta_{i,n}}\left(\frac{Y_{i,n}}{||Y_i||}\right)1_{||Y_i||>0}. \end{cases}$$

By the elimination of the multiplicative terms on both sides of the equations. We get in the first equation for all $i \in \{1,\dots,N\}$

$$\beta_{i,1} = \frac{\alpha_i}{\gamma_i+\mu_{i,1}}, \tag{8.14}$$

and in the second equation for all $i \in \{1,\dots,N\}$ *and* $n \in \{2,\dots,K\}$

$$\beta_{i,n} = \frac{p_{i,n-1}\mu_{i,n-1}\beta_{i,n-1}}{\mu_{i,n}}. \tag{8.15}$$

Finally, we only have to check that the values of $(\rho_i,\beta_{i,1},\dots,\beta_{i,K})$ *in Equation* (8.13)*,* (8.14) *and* (8.15) *also satisfies Equation* (8.12)*.*

For Equation (8.12)*. We use Lemma 8.2.1 and Equation* (8.1) *(which states that* $\forall i \;\; d_i = 1-\sum_{j=1}^{N}P(i,j)$*). By moving negative terms to the l.h.s, Equation* (8.12) *is reduced to* $0=0$*. Therefore, all equations that constitute the global balance equation at steady-state are satisfied and the proof of the theorem is complete.*

Lemma 8.2.2 *The normalization constant is equal to*

$$G = \prod_{i=1}^{N} G_i. \tag{8.16}$$

Where

$$G_i = (1-\rho_i)(1-\sum_{n=1}^{K}\beta_{i,n}). \tag{8.17}$$

Proof. *We compute the value of* G *using that* $\sum_{X,Y_1,\dots,Y_K}\pi(\mathbf{X},\mathbf{Y_1},\dots,\mathbf{Y_K}) = 1$ *and Equation* (8.5)*, we get that*

$$\begin{aligned} &\sum_{X,Y_1,\dots,Y_K} G\prod_{i=1}^{N}(\rho_i)^{X_i}||Y_i||!\prod_{n=1}^{K}\frac{(\beta_{i,n})^{Y_{i,n}}}{Y_{i,n}!} = 1 \\ \Rightarrow\; &G\prod_{i=1}^{N}\sum_{X_i,Y_{i,1},\dots,Y_{i,K}}(\rho_i)^{X_i}||Y_i||!\prod_{n=1}^{K}\frac{(\beta_{i,n})^{Y_{i,n}}}{Y_{i,n}!} = 1 \\ \Rightarrow\; &G\prod_{i=1}^{N}\sum_{X_i}(\rho_i)^{X_i}\sum_{Y_{i,1},\dots,Y_{i,K}}||Y_i||!\prod_{n=1}^{K}\frac{(\beta_{i,n})^{Y_{i,n}}}{Y_{i,n}!} = 1. \end{aligned}$$

From the assumption of the theorem $\rho_i < 1$, *then the sum on* X_i *converge*

$$G\prod_{i=1}^{N}\frac{1}{1-\rho_i}\sum_{Y_{i,1},\ldots,Y_{i,K}}||Y_i||!\prod_{n=1}^{K}\frac{(\beta_{i,n})^{Y_{i,n}}}{Y_{i,n}!}=1.$$

We partition the summation on $Y_{i,1},\ldots,Y_{i,K}$ *according to it's norm* $||Y_i||$

$$G\prod_{i=1}^{N}\frac{1}{1-\rho_i}\sum_{m=0}^{\infty}\sum_{Y_{i,1},\ldots,Y_{i,K}/||Y_i||=m}||Y_i||!\prod_{n=1}^{K}\frac{(\beta_{i,n})^{Y_{i,n}}}{Y_{i,n}!}=1.$$

Substitute $||Y_i||$ *by* m *in the previous equation. Remember the definition of the multinomial theorem*

$$\sum_{Y_{i,1},\ldots,Y_{i,K}/||Y_i||=m}||Y_i||!\prod_{n=1}^{K}\frac{(\beta_{i,n})^{Y_{i,n}}}{Y_{i,n}!}=\Big(\sum_{n=1}^{K}\beta_{i,n}\Big)^{m}.$$

After substitution we obtain:

$$G\prod_{i=1}^{N}\frac{1}{1-\rho_i}\sum_{m=0}^{\infty}\Big(\sum_{n=1}^{K}\beta_{i,n}\Big)^{m}=1.$$

As by assumption $\sum_{n=1}^{K}\beta_{i,n}<1$, *the sum converges and we get*

$$G\prod_{i=1}^{N}\frac{1}{(1-\rho_i)(1-\sum_{n=1}^{K}\beta_{i,n})}=1.$$

$$\Rightarrow\ G=\prod_{i=1}^{N}(1-\rho_i)(1-\sum_{n=1}^{K}\beta_{i,n}).$$

The proof is complete.

8.2.3 Existence of a fixed point solution

The main result of our work states that a product form of the EPN exists if the solution of the fixed point problem defined in Equations (8.2) and (8.3) exists. First of all we remind some properties related to the network.

Proposition 8.2.1 *Assume that* P *is a nonnegative matrix, whose row sums are bounded by 1. Let* Id *be the identity matrix, therefore if* P *is irreducible and substochastic (i.e at least one row sum is smaller than 1) then* $(Id-P)$ *is nonsingular.*

The proof of this proposition is based on Perron-Frobenius theory. (details are in [DMM14] Proposition (2.26)).

Assumption 8.2.1 *We consider an open and connected EPN.*

Lemma 8.2.3 *Let* P *be the routing matrix of an open connected EPN. Then* $(Id-P)$ *is non singular i.e. there exists the inverse matrix* $(Id-P)^{-1}$.

Proof. *An open and connected EPN ensures that (a) it exists at least one cell* i *in the network such that* $d_i > 0$ *(a data packet in cell* i *leaves the system with probability* d_i*). Therefore, for at least one cell* i *we have* $\sum_{j=1}^{N} P(i,j) < 1$*. That proves that* P *is substochastic. (b) The EPN is connected as defined in Definition 7.2.3 then* P *is irreducible. From Statements (a) and (b) we have that routing matrix* P *is substochastic and irreducible, we conclude (using Proposition 8.2.1) that* $(Id - P)$ *is nonsingular.*

Lemma 8.2.4 *For all* i *and* j *in* $\{1,\dots,N\}$ *and* n *in* $\{1,\dots,K\}$*, we have* $\beta_{i,n}$ *defined in Equation (8.2) and* θ_j *exists i.e.* $\beta_{i,n} > 0$ *and* $\theta_j > 0$*. Note that*

$$\theta_j = \sum_{n=1}^{K} (1 - p_{j,n}) \mu_{j,n} \beta_{j,n}. \tag{8.18}$$

Proof. *By definition, we have for all* $i \in \{1,\dots,N\}$ *and* $n \in \{1,\dots,K\}$*,* $\mu_{i,n} > 0$*,* $\alpha_i > 0$ *and* $\gamma_i > 0$*, then* $\beta_{i,n}$ *expressed in Equation (8.2) exists and* $\beta_{i,n} > 0$*.*
$\theta_j > 0$ *since (i)* $\mu_{j,n} > 0$ *, (ii)* $\beta_{j,n} > 0$ *and (iii) we know that at least the last term* $(1 - p_{j,K}) \mu_{j,K} \beta_{j,K} > 0$ *as* $p_{j,K} = 0$*.*

Lemma 8.2.5 *Equation (8.3) that represents flow equations of data packets in each cell* i*, has a fixed point* ρ_i*.*

Proof. *By substitution using Equation (8.18), then Equation (8.3) becomes* $\forall\ i \in \{1,\dots,N\}$

$$\rho_i = \frac{\lambda_i + \sum_{j=1}^{N} \theta_j \rho_j P(j,i)}{\theta_i}, \tag{8.19}$$

$$\Rightarrow\ \rho_i \theta_i - \sum_{j=1}^{N} \theta_j \rho_j P(j,i) = \lambda_i. \tag{8.20}$$

Which represents the flow equations in Jackson networks [CY01].
Let $\vec{\rho\theta} = (\rho_1\theta_1,\dots,\rho_N\theta_N)$*,* $\vec{\lambda} = (\lambda_1,\dots,\lambda_N)$ *and Id the identity matrix, then Equation (8.20) for vectors*

$$\vec{\rho\theta}\ (Id - P) = \vec{\lambda}. \tag{8.21}$$

Using Lemma 8.2.3, (Id - P) admits an inverse

$$\Rightarrow\ \vec{\rho\theta}\ = \vec{\lambda}(Id - P)^{-1}.$$

This proves the existence of the vector $\rho\theta$ *therefore* ρ_i *exists since* θ_i *exists as stated in Lemma 8.2.4. The proof is complete.*

8.3 Performance and Energy evaluation

In this section, we compute both energy and performance measures. Since our system has a product form solution, the measures can be computed as expectations of rewards separately on each cell. Note that Equation (8.5) can be expressed as

$$\pi(\mathbf{X}, \mathbf{Y_1}, \dots, \mathbf{Y_K}) = \prod_{i=1}^{N} \pi(X_i, Y_{i,1}, \dots, Y_{i,K}), \tag{8.22}$$

and

$$\pi(X_i, Y_{i,1}, \ldots, Y_{i,K}) = \pi(X_i)\pi(Y_{i,1}, \ldots, Y_{i,K})$$

where, by the substitution of G_i

$$\pi(X_i) = (1-\rho_i)(\rho_i)^{X_i} \quad , \quad \pi(0) = (1-\rho_i) \tag{8.23}$$

and

$$\pi(Y_{i,1}, \ldots, Y_{i,K}) = (1 - \sum_{n=1}^{K} \beta_{i,n})||Y_i||! \prod_{n=1}^{K} \frac{(\beta_{i,n})^{Y_{i,n}}}{Y_{i,n}!}. \tag{8.24}$$

8.3.1 Loss rate of energy packets

For each cell in the EPN, energy packets are lost either when there is no data packets in the DP queue, or due to battery leaks which occur in phase 1 of the Cox process.

Lemma 8.3.1 *Let* $Loss^T$ *be the total loss rate of EPs in the network, then:*

$$Loss^T = \sum_{i=1}^{N} \sum_{||Y_i||>0} \pi(Y_{i,1}, \ldots, Y_{i,K}) \Big[\pi(0) r_1(Y_{i,1}, \ldots, Y_{i,K}) + \sum_{X_i} \pi(X_i) r_2(Y_{i,1}, \ldots, Y_{i,K}) \Big]. \tag{8.25}$$

Where $r_1(Y_{i,1}, \ldots, Y_{i,K})$ *(resp.* $r_2(Y_{i,1}, \ldots, Y_{i,K})$ *) is described in Equation (8.29) (resp. Equation (8.32)).*

Proof. *Let* $Loss^{(i)}$ *be the loss rate of EPs in cell* i*. As the EPN network has a product-form solution, then :*

$$Loss^T = \sum_{i=1}^{N} Loss^{(i)}. \tag{8.26}$$

Let $Loss_1^{(i)}$ *(resp.* $Loss_2^{(i)}$*) be the loss rate of EPs due to emptiness of DP queue (resp. battery leakage) in cell* i*, then:*

$$Loss^{(i)} = Loss_1^{(i)} + Loss_2^{(i)}. \tag{8.27}$$

States that correspond to $Loss_1^{(i)}$*, in Equation (8.28), are those where,* $X_i = 0$ *and at least one EP in the battery is ready to trigger a DP. This explains the sum on* $||Y_i|| > 0$*. Also, for each state, we multiply by the appropriate reward* $r_1(Y_{i,1}, \ldots, Y_{i,K})$*. The reward in this case, Equation (8.29), represents the rate at which a DP is triggered in each phase of the Cox process. So*

$$Loss_1^{(i)} = \sum_{Y_{i,1},\ldots,Y_{i,K}/||Y_i||>0} \pi(0, Y_{i,1}, \ldots, Y_{i,K}) r_1(Y_{i,1}, \ldots, Y_{i,K}), \tag{8.28}$$

where

$$r_1(Y_{i,1}, \ldots, Y_{i,K}) = \sum_{n=1}^{K} \frac{Y_{i,n}}{||Y_i||} (1 - p_{i,n}) \mu_{i,n}. \tag{8.29}$$

Using the separation simplifications in Equation (8.22), (8.23) and (8.24), $Loss_1^{(i)}$ *can be expressed as*

$$Loss_1^{(i)} = \pi(0) \sum_{||Y_i||>0} \pi(Y_{i,1},\ldots,Y_{i,K}) r_1(Y_{i,1},\ldots,Y_{i,K}). \tag{8.30}$$

For $Loss_i^{(2)}$*, we use the same approach as for* $Loss_i^{(1)}$*. This means that we multiply the steady state probability of the appropriate states by the corresponding rewards* $r_2(Y_{i,1},\ldots,Y_{i,K})$*. In this case, we are interested in the rate of EP loss due to battery leakage that occurs in the first phase of the Cox process with the rate* $\frac{Y_{i,1}}{||Y_i||}\gamma_i$*. Note that we can use either the summation with* $||Y_i|| > 0$ *or* $Y_{i,1} > 0$*, the final result of* $Loss_2^{(i)}$ *remains the same. We use* $||Y_i|| > 0$ *for simplification purposes. So*

$$Loss_2^{(i)} = \sum_{X_i,Y_{i,1},\ldots,Y_{i,K}/||Y_i||>0} \pi(X_i, Y_{i,1},\ldots,Y_{i,K}) r_2(Y_{i,1},\ldots,Y_{i,K}), \tag{8.31}$$

where

$$r_2(Y_{i,1},\ldots,Y_{i,K}) = \frac{Y_{i,1}}{||Y_i||}\gamma_i. \tag{8.32}$$

Also, using the separation simplifications in Equation (8.22), (8.23) and (8.24) then

$$Loss_2^{(i)} = \sum_{X_i} \pi(X_i) \sum_{||Y_i||>0} \pi(Y_{i,1},\ldots,Y_{i,K}) r_2(Y_{i,1},\ldots,Y_{i,K}), \tag{8.33}$$

Finally, by the substitution of $Loss_1^{(i)}$ *and* $Loss_2^{(i)}$ *in Equation (8.27). And using Equation (8.26), we obtain Equation (8.25) and the proof is complete.*

8.3.2 Waiting time and total number of data packets

Lemma 8.3.2 *Let* $E[X]$ *be the total number of data packets in the system, then*

$$E[X] = \sum_{i=1}^{N} \Big[\frac{\lambda_i + \sum_{j=1}^{N} \theta_j \rho_j P(j,i)}{\theta_i - (\lambda_i + \sum_{j=1}^{N} \theta_j \rho_j P(j,i))} \Big]. \tag{8.34}$$

See Equation (8.18) for the value of θ_i*.*

Proof. *We first calculate* $E[X_i]$ *the mean number of data packets in cell* i*. As the EPN network has a product form solution, then*

$$E[X] = \sum_{i=1}^{N} E[X_i]. \tag{8.35}$$

From Equation (8.23) we can deduce that

$$E[X_i] = \frac{\rho_i}{1-\rho_i}. \tag{8.36}$$

Then by substitution of ρ_i *(see Equation (8.3)) and using Equation (8.18), we get*

$$E[X_i] = \frac{\lambda_i + \sum_{j=1}^{N} \theta_j \rho_j P(j,i)}{\theta_i - (\lambda_i + \sum_{j=1}^{N} \theta_j \rho_j P(j,i))}. \tag{8.37}$$

Note that $E[X_i] > 0$ *since from stability conditions (see Theorem 8.2.1), we have for all* i, $\rho_i < 1$ *then* $\theta_i > \lambda_i + \sum_{j=1}^{N} \theta_j \rho_j P(j,i)$. *Finally, using Equation (8.35) we obtain Equation (8.34) and the proof is complete.*

Lemma 8.3.3 *Let* $E[T_i]$ *be the mean waiting time of a data packet in cell* i, *then*

$$E[T_i] = \frac{1}{\theta_i - (\lambda_i + \sum_{j=1}^{N} \theta_j \rho_j P(j,i))}. \tag{8.38}$$

Proof. *Let* Λ_i *be the overall arrival rate to DP queue of cell* i, *including both external arrivals and internal transitions*

$$\Lambda_i = \lambda_i + \sum_{n=1}^{K} \sum_{j=1}^{N} (1 - p_{j,n}) \mu_{j,n} \rho_j \beta_{j,n} P(j,i). \tag{8.39}$$

Then from Little's law for data packets in the cell i

$$E[T_i] = \frac{E[X_i]}{\Lambda_i}. \tag{8.40}$$

By substitution, using Equation (8.36)

$$E[T_i] = \frac{\rho_i}{\Lambda_i (1 - \rho_i)}. \tag{8.41}$$

Also using Equation (8.3) to substitute ρ_i, *Equation (8.39) to substitute* Λ_i *and Equation (8.18), we obtain after simplifications*

$$E[T_i] = \frac{1}{\theta_i - (\lambda_i + \sum_{j=1}^{N} \theta_j \rho_j P(j,i))}.$$

As for $E[X_i]$ *we have* $E[T_i] > 0$ *since the denominator is a strictly positive value, and the proof is complete.*

Lemma 8.3.4 *Let* $E[T]$ *be the mean waiting time of a data packet in the EPN network, then*

$$E[T] = \sum_{i=1}^{N} \left[\frac{\lambda_i + \sum_{j=1}^{N} \theta_j \rho_j P(j,i)}{\theta_i - (\lambda_i + \sum_{j=1}^{N} \theta_j \rho_j P(j,i))} \right] \left[\sum_{i=1}^{N} \lambda_i \right]^{-1}. \tag{8.42}$$

Proof. *Let* $\chi = \sum_{i=1}^{N} \lambda_i$ *be the total arrival rate of data packets incoming from outside the network. Then from Little's law*

$$E[T] = \frac{E[X]}{\chi}.$$

Therefore using Lemma 8.3.2, we obtain Equation (8.42).

8.4 Solar panel assignment

We are now studying an EPN sensor network that collects information in a backbone network. Each sensor gets energy packets from photo-voltaic solar panels, energy flow is stored in sensor's battery. The aim is to obtain an optimal distribution of Δ solar panels over the N sensors (Δ and N being a non-zero positive integers). More precisely, we are looking for an assignment of the solar panels that minimizes the average end to end delay of data packets in the network. Also we considered the following constraints: (a) A solar panel could be assigned to only one sensor, but a sensor receives energy from at least one panel. We consider identical panels (i.e the panels generates the same rate of energy) (b) Stability constraints as described in Theorem 8.2.1. The EPs stability condition provides an upper bound on the number of panels to use, while DPs' condition of stability gives a lower bound of the quantity of panels.

Lemma 8.4.1 *Let Φ_i be the number of solar panels affected to a cell i. Then, under EPs stability conditions of the EPN, for all $i \in \{1,\dots,N\}$*

$$\Phi_i < \frac{\gamma_i + \mu_{i,1}}{\alpha}\Big(1 + \sum_{n=2}^{K} \frac{\mu_{i,1}}{\mu_{i,n}} \prod_{r=1}^{n-1} p_{i,r}\Big)^{-1}, \tag{8.43}$$

where $\alpha > 0$ is the rate of energy packets generated by a solar panel (which is supposed to be the same since the panels are identical).

Proof. *In this lemma we are interested in the stability condition of EPs (i.e. for all $i \in \{1,\dots,N\}$, $\sum_{n=1}^{K} \beta_{i,n} < 1$).*

$$\sum_{n=1}^{K} \beta_{i,n} < 1 \;\Rightarrow\; \beta_{i,1} + \sum_{n=2}^{K} \beta_{i,n} < 1.$$

By substitution using Equation (8.2)

$$\Rightarrow\; \frac{\alpha_i}{\gamma_i + \mu_{i,1}}\Big(1 + \sum_{n=2}^{K} \frac{\mu_{i,1}}{\mu_{i,n}} \prod_{r=1}^{n-1} p_{i,r}\Big) < 1. \tag{8.44}$$

The rate of energy packets coming into a cell's battery α_i is a function of the number of solar panels assigned to the cell. Then

$$\alpha_i = \Phi_i \alpha. \tag{8.45}$$

By the substitution of α_i in Equation (8.44), we obtain Equation (8.43). The proof is complete.

8.4.1 Case of a tree EPN topology

Let us now simplify the DPs flow equations due to the tree topology (see Fig. 8.3 and Fig. 8.6).

Lemma 8.4.2 *Let i be an arbitrary node of the tree, $\mathrm{A}(i)$ will denote the set of nodes in the sub-tree rooted at i (including i). Then, the flow equations of* DP *queues (Equation (8.3)) can be simplified as*

$$\rho_i = \frac{\sum_{j\in \mathrm{A}(i)} \lambda_j}{\sum_{n=1}^{\mathrm{K}} (1-p_{i,n})\mu_{i,n}\beta_{i,n}}. \tag{8.46}$$

Proof. *by induction on the node number. We assume that the nodes are numbered according to a topological ordering of the tree.*

- $i = 1$*: node* 1 *is a leaf because it has no predecessor. Therefore* $\mathrm{A}(1) = \{1\}$*. Thus*

  $$\rho_1 = \frac{\lambda_1}{\sum_{n=1}^{\mathrm{K}} (1-p_{1,n})\mu_{1,n}\beta_{1,n}},$$

  and it is also equal to $\frac{\sum_{j\in \mathrm{A}(1)} \lambda_j}{\sum_{n=1}^{\mathrm{K}} (1-p_{1,n})\mu_{1,n}\beta_{1,n}}$*. Thus the property is established for* $i = 1$*.*

- *arbitrary* i*. Let us now assume that the property is established for all* $j < i$*. Thus we have:*

  $$\rho_j \sum_{n=1}^{\mathrm{K}} (1-p_{j,n})\mu_{j,n}\beta_{j,n} = \sum_{k\in \mathrm{A}(j)} \lambda_k.$$

  Let $\Gamma^-(i)$ *be the set of predecessor of* i*. Remember that routing matrix* P *encodes the tree topology. Therefore* $\mathrm{P}(j,i) = 1$ *if* j *is a predecessor of* i *(i.e. in* $\Gamma^-(i)$*) and* 0 *otherwise. Consider the numerator of the flow equation for DP in station* i*:*

  $$\lambda_i + \sum_{j\ \in\Gamma^-(i)} \sum_{n=1}^{\mathrm{K}} (1-p_{j,n})\mu_{j,n}\beta_{j,n}.$$

  Applying the induction, the numerator becomes:

  $$\lambda_i + \sum_{j\ \in\Gamma^-(i)} \sum_{k\in \mathrm{A}(j)} \lambda_k.$$

  Due to the recursive construction of any tree, we get:

  $$\lambda_i + \sum_{j\ \in\Gamma^-(i)} \sum_{k\in \mathrm{A}(j)} \lambda_k = \sum_{j\in \mathrm{A}(i)} \lambda_j.$$

And the proof is complete.

Lemma 8.4.3 *The DPs stability condition is* $\rho_i < 1$ *for all* i*. Then, in a tree EPN, We have for all* $i \in \{1,\ldots,\mathrm{N}\}$

$$\Phi_i > \frac{(\gamma_i + \mu_{i,1})}{\alpha\mu_{i,1}} \sum_{j\in \mathrm{A}(i)} \lambda_j. \tag{8.47}$$

Proof. *From Lemma 8.2.1 we have for all* $i \in \{1,\dots,N\}$, $\sum_{n=1}^{K}(1-p_{i,n})\mu_{i,n}\beta_{i,n} = \alpha_i - \gamma_i\beta_{i,1}$ *and by the substitution of* $\beta_{i,1}$ *and* α_i *(Equation (8.45)) we obtain*

$$\sum_{n=1}^{K}(1-p_{i,n})\mu_{i,n}\beta_{i,n} = \frac{\alpha\Phi_i\mu_{i,1}}{\gamma_i+\mu_{i,1}}. \tag{8.48}$$

Using Equation (8.48) in Equation (8.46), we get

$$\rho_i = \frac{\gamma_i+\mu_{i,1}}{\alpha\Phi_i\mu_{i,1}}\sum_{j\in A(i)}\lambda_j. \tag{8.49}$$

DPs stability conditions are $\forall\ i \in \{1,\dots,N\},\ \rho_i < 1$. *Hence, by combining Equation (8.49) and that* $\rho_i < 1$ *we obtain Equation (8.47), and the proof is complete.*

8.4.1.1 Optimization problem

Let us now study the domain of definition. The set of possible values for $(\Phi_1,\dots,\Phi_N)$ comes from all the constraints we got (particularly Lemma 8.4.1, Lemma 8.4.3) : $\forall i \in \{1,\dots,N\}$

$$\begin{cases} \Phi_i < \frac{\gamma_i+\mu_{i,1}}{\alpha}\left(1+\sum_{n=2}^{K}\frac{\mu_{i,1}}{\mu_{i,n}}\prod_{r=1}^{n-1}p_{i,r}\right)^{-1}, \\ \Phi_i > \frac{(\gamma_i+\mu_{i,1})}{\alpha\mu_{i,1}}\sum_{j\in A(i)}\lambda_j > 0, \\ \sum_{i=1}^{N}\Phi_i = \Delta. \end{cases}$$

Using ρ_i in Equation (8.49), and the expression of the total mean waiting time $E[X] = \sum_{i=1}^{N}\frac{\rho_i}{1-\rho_i}$. Then, according to Little's law, we derive the mean response time function, which is equal to $\frac{E[X]}{\sum_{i=1}^{N}\lambda_i}$, by substitution we obtain the function to minimize:

$$f(\Phi_1,\dots,\Phi_N) = \sum_{i=1}^{N}\frac{(\gamma_i+\mu_{i,1})\sum_{j\in A(i)}\lambda_j}{\alpha\Phi_i\mu_{i,1}-(\gamma_i+\mu_{i,1})\sum_{j\in A(i)}\lambda_j}\left(\sum_{i=1}^{N}\lambda_i\right)^{-1}. \tag{8.50}$$

Function in Equation (8.50) can be expressed as $f(\Phi_1,\dots,\Phi_N) = \sum_{i=1}^{N}f(\Phi_i)$. As f is decreasing in each Φ_i, then to minimize Equation (8.50) we should increase Φ_i. Therefore, the optimal solution is the upper bound (excluded) for each cell i if Δ is at least equal to sum of upper bounds (excluded). Otherwise, we suggest an heuristic (Algorithm 8) and a Gradient descent algorithm (Algorithm 9).

Note that the aim is to propose a distribution of Δ panels over the N cells. The heuristic algorithm consists of 3 steps. In step 1) we calculate the bounds of the solution domain for each cell. In Step 2) we verify that the domain has at least one solution and that Δ is at least equal to the sum of the lower bound of cells. Also if Δ is equal to or greater than the sum of the upper bound of the cells, then the solution is the upper bound of the panels in each cell. In step 3) we know that the solution is between the lower (included) and the upper (excluded) bounds. So we start with a first solution which corresponds to the lower bound of panels in each cell (Lemma 8.4.3), then at each iteration we add a panel in the most loaded cell that has not reached its upper bound. The algorithm stops when we have reached Δ panels in the network. Note that the algorithm may stop in step 2) due to invalid parameters or an empty set for the solution.

In Algorithm 8, we decide to assign the new panel to the most loaded cell, while in Algorithm 9 (Gradient descent) the new panel is assigned to the cell that will minimize the most Equation (8.50). Therefore, the main difference between the heuristic and the Gradient descent is the decision made in step 3.2). Gradient descent algorithm performs the optimal local decision at each step. We have verified numerically that it achieves the optimal global distribution of panels. But we do not have a theoretical proof of this optimality.

Algorithm 8: Heuristic: Panels assignment algorithm

Input : Δ the number of panels.
Output: An assignment of the Δ panels.

Step 1) Calculation of lower and upper bound of Φ_i for each cell i. (Lemma 8.4.1 and 8.4.3).

Step 2) Verification of bounds:
Step 2.1) If there is a cell whose upper bound of the number of panels is lower than the lower bound, then print "No solution for such parameters", the algorithm stops. Otherwise go to Step 2.2).
Step 2.2) If Δ is strictly lower than the sum of all lower bounds, then print "Δ is too low", the algorithm stops. Otherwise go to Step 2.3).
Step 2.3) If Δ is equal or greater than the sum of all upper bounds, then the solution is the upper bound (Lemma 8.4.1) of each cell, the algorithm stops. Otherwise, go to step 3).

Step 3) Constructing progressively the solution:
Step 3.1) We construct a first solution, which corresponds to the lower bound of each cell.
Step 3.2) We add a panel in the most loaded cell that can receive a panel.
Step 3.3) If we achieved Δ panels in the network, then the algorithm stops. Otherwise go to Step 3.2).

Step 4) Return the solution and the corresponding delay.

Algorithm 9: Gradient descent: Panels assignment algorithm

Input : Δ the number of panels.
Output: An optimal assignment of the Δ panels.

Step 1) and Step 2) are the same as in Algorithm 8.

Step 3) Constructing progressively the solution:
Step 3.1) We construct a first solution, which corresponds to the lower bound of each cell.
Step 3.2) We add a panel in a cell that can receive a panel and that minimizes the objective function (Equation (8.50)) the most.
Step 3.3) If we achieved Δ panels in the network, than the algorithm stops. Otherwise go to Step 3.2).

Step 4) Return the solution and the corresponding delay.

8.4.1.2 A collecting sensor network of $N = 7$ cells

We consider a tree EPN topology with $N = 7$ cells (Fig. 8.3) and $K = 4$ phases. Cells $1,2,3$ and 4 are leaf cells, 5 and 6 are inter-cellular cells and the root cell is 7. All cells point towards the root. Service time is an Erlang process (i.e. $\forall i \in \{1,\ldots,N\}$ $\mu_{i,n} = \mu_i$ and $\forall n \in \{1,\ldots,K-1\}$ $p_{i,n} = 1$ and $p_{i,K} = 0$). We suppose that only leaf cells receives DPs from outside the network, arrivals rate of DPs in each cell, are respectively, $\{1,2,3,4,0,0,0\}$. Also, the service rate and leakage rate of EPs in each cell is $\forall\ i, \mu_i = 50$ and $\gamma_i = 5$. EPs rate generated by a single solar panel is $\alpha = 2$ and the number of panels is $\Delta = 41$. We made sure to consider valid parameters ($21 \leq \Delta \leq 41$) in order to perform step 3) which is the main part of both algorithms. All parameters and rewards are expressed for a unit of time.

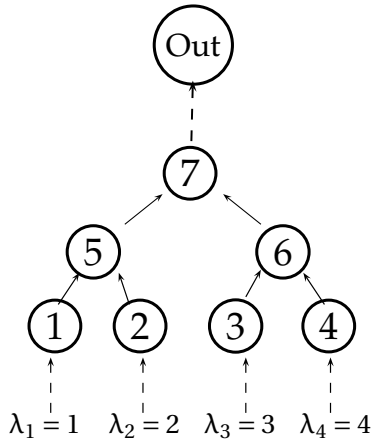


Figure 8.3: A tree EPN topology of $N = 7$ cells.

Fig. 8.4 illustrates the evolution of the mean waiting time of a DP in the network while increasing the number of panels Δ. The mean waiting time was calculated using the solution proposed by the two algorithms. We first observe that the mean waiting time decreases with the increase in the number of panels in the network (as the number of panels is at the denominator of the function to minimize i.e. Equation (8.50)). We also notice that the solutions proposed by the two methods are quasi similar. In particular, for this example we have 22 experiments (from $\Delta = 20$ to $\Delta = 41$) of which 16 experiments generate the same solution (solutions details are in Table 8.1). In the 6 other experiments ($\Delta = \{29,31,32,35,36,39\}$) the delay of the heuristic is around 10^-3 and 10^-4 above the Gradient descent. In Table 8.1 we present the solutions proposed by both algorithms for some values of Δ. In this table, an entry 'x1&x2' means that the heuristic answers $x1$ panels while Gradient descent proposes $x2$ while a single number (say 'x') means that both algorithms assign the same number of panels x to the corresponding cell.

Fig. 8.5 illustrates the loss rate of EPs (see Equation (8.25)) generated by both algorithms. We notice that the loss rate increases in most cases by increasing the number of panels in the network. Also, we observe that the heuristic gives better results for the energy loss, as the Gradient algorithm only cares about the waiting time. Therefore, the best configuration of the panels for the waiting time is obviously not the same for energy losses.

It is worthy to note that both methods are very fast. We have also verified numerically that the gradient descent algorithm generates the optimal solution, i.e. by comparing its solution to an exhaustive algorithm. Since the exhaustive method is exponential in execution time, then we can perform this exhaustive analysis up to $\Delta = 50$ panels.

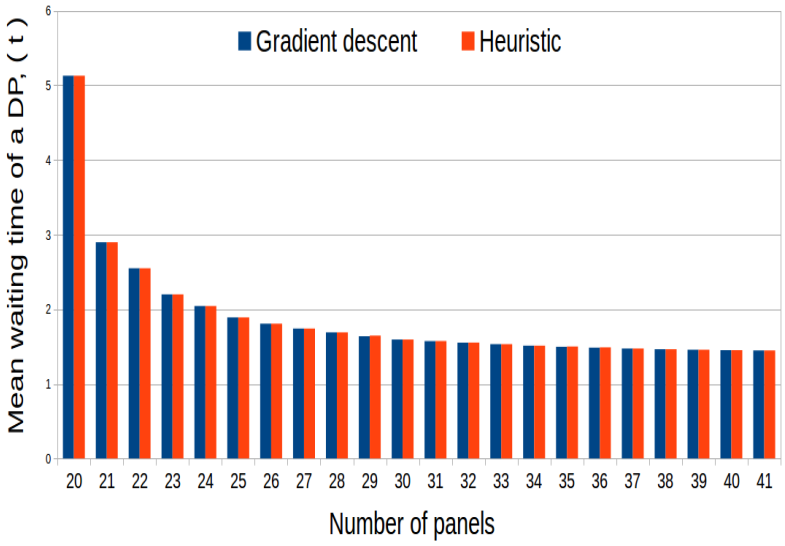


Figure 8.4: Mean waiting time of a DP in a tree EPN of $N = 7$ cells: Heuristic solution Vs Gradient descent solution .

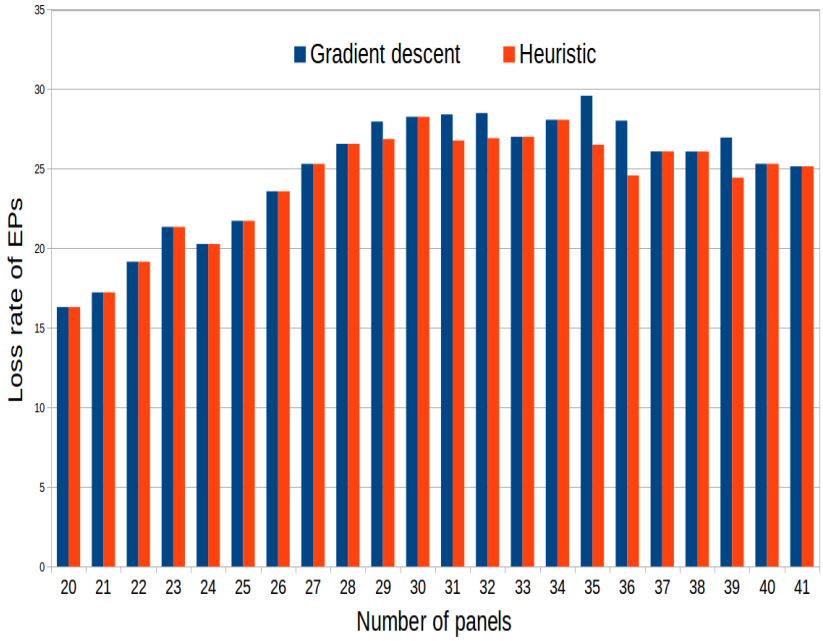


Figure 8.5: Loss rate of EPs in a tree EPN of $N = 7$ cells: Heuristic solution Vs Gradient descent solution .

Δ	Φ_1	Φ_2	Φ_3	Φ_4	Φ_5	Φ_6	Φ_7
20	1	2	2	3	2	4	6
21	1	2	2	3	2	5	6
22	1	2	3	3	2	5	6
23	1	2	3	3	3	5	6
24	1	2	3	3	3	6	6
25	1	2	3	4	3	6	6
26	2	2	3	4	3	6	6
27	2	3	3	4	3	6	6
28	2	3	4	4	3	6	6
29	2	3	4	5 & 4	3 & 4	6	6
30	2	3	4	5	4	6	6
31	2	3	4	6 & 5	4 & 5	6	6
32	2	3	5	6 & 5	4 & 5	6	6
33	2	3	5	6	5	6	6
34	2	4	5	6	5	6	6
35	2 & 3	4	6 & 5	6	5	6	6
36	2 & 3	4	6	6	6 & 5	6	6
37	3	4	6	6	6	6	6
38	3	5	6	6	6	6	6
39	3 & 4	6 & 5	6	6	6	6	6
40	4	6	6	6	6	6	6
41	5	6	6	6	6	6	6

Table 8.1: Distribution of Δ panels using Heuristic and Gradient descent algorithm, $N = 7$ cells.

8.4.1.3 A collecting sensor network of $N = 20$ cells

In this example (Fig. 8.6), we consider a wider tree EPN network ($N = 20$ cells), where the graph's structure is less regular. The service is still an Erlang process with $K = 4$ phases. We suppose that not only leaf cells receives data packets from outside the network. Cells that receive data packets are 1,2,3,4,5,6,7,8,9,10,18. Arrival rate of DPs, respectively in each cell, are $\{1, 1, 1, 1, 2, 2, 3, 3, 3, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0\}$. We suppose that service rate is $\forall\ i, \mu_i = 85$, also leakage rate of EPs is $\forall\ i, \gamma_i = 5$. EPs rate generated by a single solar panel is $\alpha = 2$. Notice that using these parameters the total number of panels to use in this topology should be in $\{51, 220\}$. That means when $\Delta \leq 50$ then the number of panels to distribute is not enough therefore both algorithms will stop in Step 2.2). And when $\Delta \geq 220$ therefore the best solution correspond to the upper bound of Φ_i in each cell, as stated in Step 2.3) in both algorithms. Hence, for this example we consider $\Delta = 90$ panels.

As in the model with $N = 7$ cells, we observe in Fig. 8.7 that the mean waiting time of a DP decreases when increasing the number of panels in the system. As DPs are triggered using energy parquets, otherwise they wait. Therefore, it is evident that the mean waiting time of a DP decreases when increasing the sources of

incoming energy in the network. Also, we observe that Gradient descent algorithm performs better than the heuristic algorithm. Nevertheless, heuristic gives almost the same results as the gradient, which is the optimal solution (verified numerically by comparing it to an exhaustive algorithm).

Fig. 8.8 illustrates the total loss rate of EPs in the network. To note that losses of EPs are due to battery leakages and emptiness of DP queue. Hence, the more panels we add, the less DPs are waiting in the queues, then, the more the queues are empty. Also the mean leakage rates in the system will increase while increasing the number of panels. That explains the increasing histograms in Fig. 8.8. We also observe that the Gradient descent is worse than heuristic algorithm, when it concerns total loss rate of EPs, that result is only an observation, as both algorithms aims to optimize the waiting time of a DP. Hence, we can deduce that optimizing the waiting time, will unfortunately, not optimize the total loss rate of EPs. In Table 8.2 we show the distribution of $\Delta = \{51, 60, 70, 80, 90\}$ panels over the network, using both algorithms.

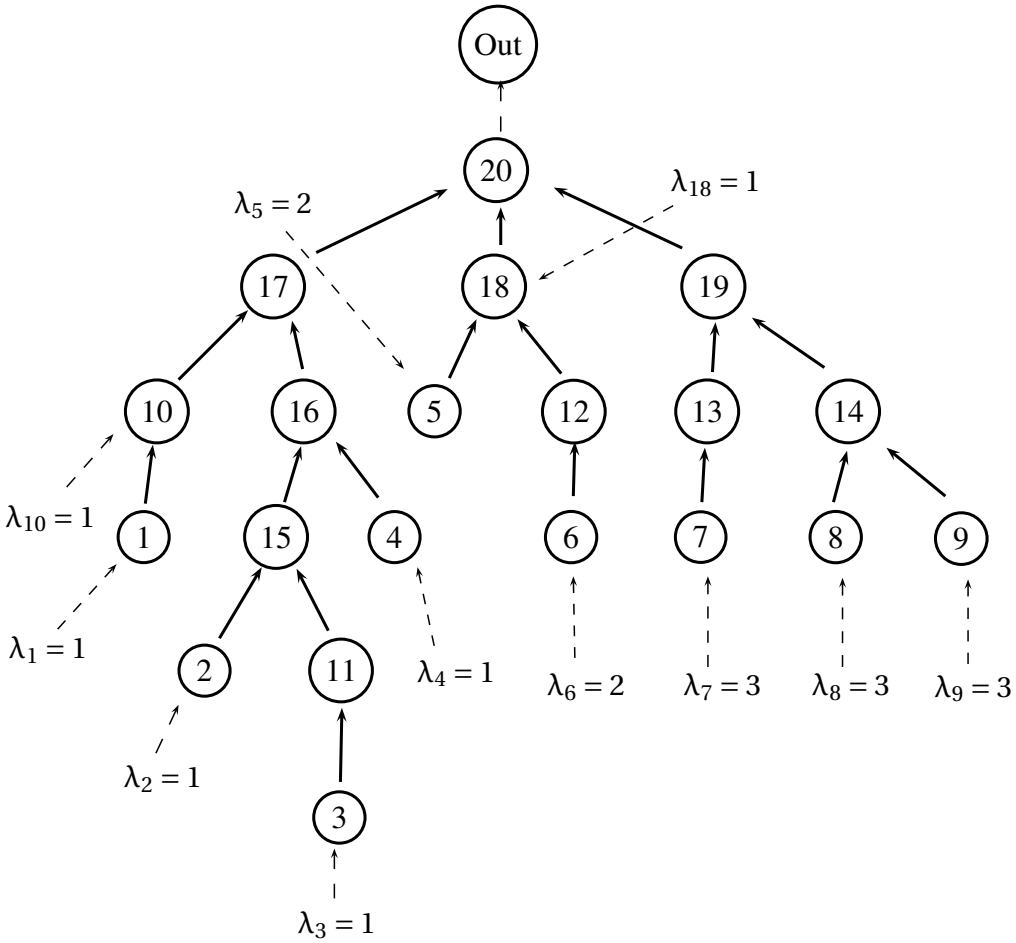


Figure 8.6: A tree EPN topology of $N = 20$ cells.

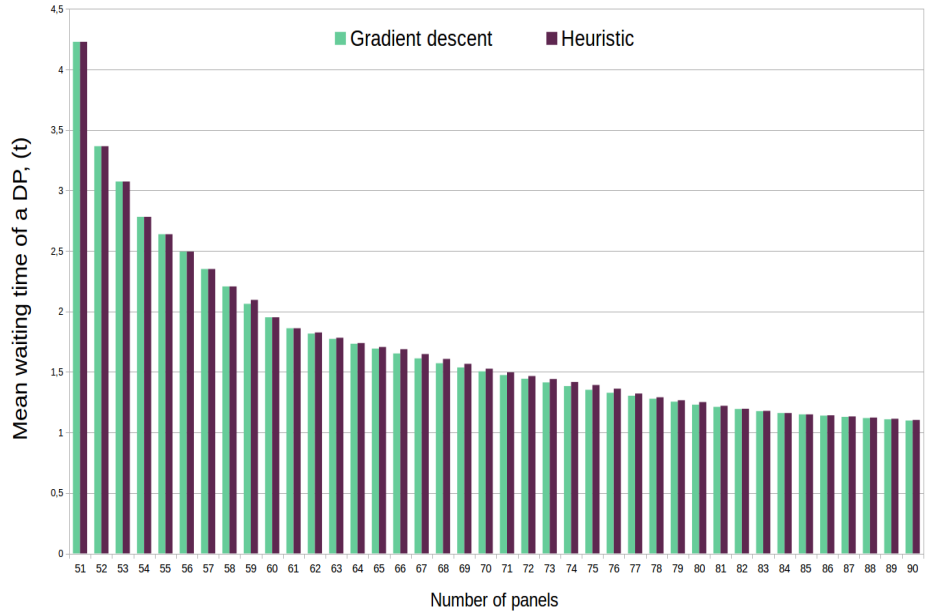


Figure 8.7: Mean waiting time of a DP in a tree EPN of N = 20 cells: Heuristic solution Vs Gradient descent solution .

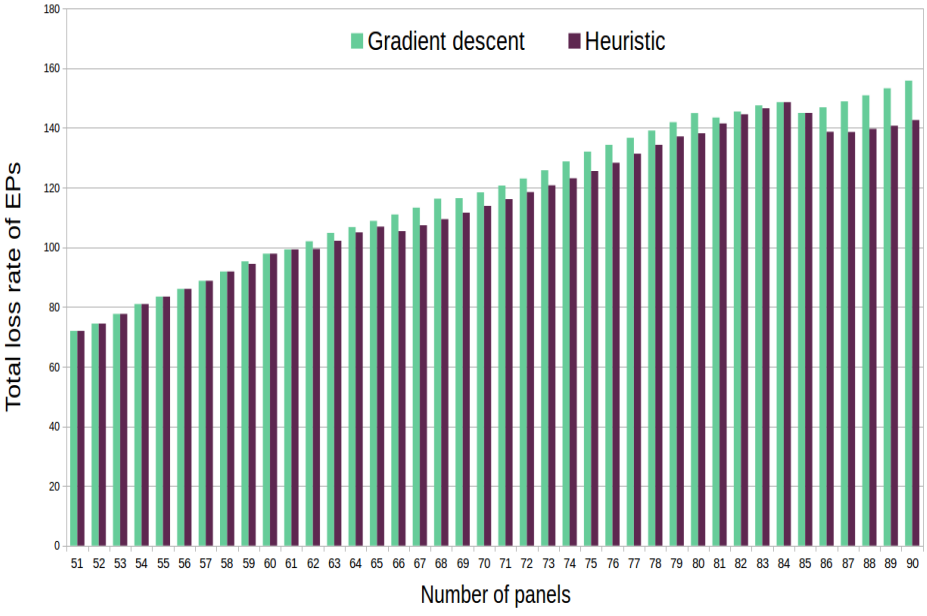


Figure 8.8: Loss rate of EPs in a tree EPN of N = 20 cells: Heuristic solution Vs Gradient descent solution .

Δ	51	60	70	80	90
Φ_1	1	1	2	2	2
Φ_2	1	1	2	2	2
Φ_3	1	1	2	2	2
Φ_4	1	1	2	2	2
Φ_5	2	2	2	3	3
Φ_6	2	2	2	3	3
Φ_7	2	3	3	4	5
Φ_8	2	3	3	4	4 & 5
Φ_9	2	3	3	4	4 & 5
Φ_{10}	2	2	2	3	3
Φ_{11}	1	1	1 & 2	2	2
Φ_{12}	2	2	2	3	3
Φ_{13}	2	3	3	4	4 & 5
Φ_{14}	4	5	6	7 & 6	8 & 7
Φ_{15}	2	2	2	2 & 3	3
Φ_{16}	2	3	3	3 & 4	4 & 5
Φ_{17}	3	4	5	5	7 & 6
Φ_{18}	3	4	5	5	7 & 6
Φ_{19}	5	6	9 & 8	9 & 8	11 & 10
Φ_{20}	11	11	11	11	11

Table 8.2: Distribution of Δ panels using Heuristic and Gradient descent algorithm, $N = 20$ cells.

8.4.2 A star EPN topology

Low Power Wide Area Network (LPWAN) technologies are characterized by long range links (several kilometers) and have star network topologies [Van+15]. These systems do not focus on enabling high data rates per device or on minimizing latency. Rather, the key metrics concerned are energy efficiency, scalability and wide-area coverage, which comes with minimum power consumption and maintenance costs [Son+17]. In LPWAN family, the most widely used technology is Long Rang (LoRa). This in mainly due to the utilization of unlicensed bands. As a result, the LoRa networks is easy to deploy over a range of several kilometers [Son+17]. The LoRa system's architecture consists of three main components [Van+15]:

- End-Devices: sensors/actuators are devices that are connected via LoRa radio interface to one or more Lora Gateways;
- Gateways: are concentrators that connect end-devices to the LoRa NetServer;
- NetServer: The network server is the central element of the network's architecture. The NetServer controls the whole network (resource management, admission control, security, etc).

LoRa is a low energy network with a battery which may be used several years. Adding an energy harvesting possibility will increase her availability. This harvesting is modelled by an EPN. In the following, we use an EPN network to model a LoRa sensor network with solar panels in order to study both energy losses of EPs and the waiting time of a DP. As depicted in Fig. 8.9, we considered a star EPN network of $N = 10$ cells. Cells $1, 2, \ldots, N-1$ are end-devices, while cell N is the LoRa gateway, also the LoRa NetServer is represented by the "Out" cell. The main difference with the first model is the topology. Here we assume that the topology is a star with symmetric directed edges as the communication between the nodes and the gateway takes place in both directions (see Fig. 8.9).

8.4.2.1 Optimization problem

Before presenting the optimization problem, we first have to adapt Lemma 8.4.2 and Lemma 8.4.3 that do not hold for the star topology:

Lemma 8.4.4 *Let i be an arbitrary node in a star EPN network. Then, the flow equations of* DP *queues (Equation (8.3)) can be simplified as, $\forall\ i \in \{1, \ldots, N\}$*

$$\rho_i = \frac{C(i)\sum_{j=1}^{N}\lambda_j}{\theta_i(1-\sum_{j=1}^{N}P(N,j))} + \frac{\lambda_i}{\theta_i}1_{i<N} \tag{8.51}$$

where

$$C(i) = \begin{cases} P(N,i) & if\ i \in \{1,\ldots,N-1\} \\ 1 & if\ i = N. \end{cases} \tag{8.52}$$

Proof. *Note that we are considering a star topology with $1, 2, \ldots, N$ cells. The middle cell is cell N, which transmit data to the NetServer (i.e. $d_N > 0$). Then*

we have $\forall i \in \{1,\ldots,N-1\}, P(i,N) = 1$ *and* $\sum_{i=1}^{N} P(N,i) < 1$ *as* $d_N > 0$. *By using these typology's features in the flow Equation* (8.3), *we obtain:*

$$\forall\ i \in \{1,\ldots,N-1\},\quad \rho_i = \frac{\lambda_i + \rho_N \theta_N P(N,i)}{\theta_i} \tag{8.53}$$

and

$$\rho_N = \frac{\lambda_N + \sum_{n=1}^{K}\sum_{j=1}^{N}(1-p_{j,n})\mu_{j,n}\rho_j\beta_{j,n}P(j,N)}{\theta_N}. \tag{8.54}$$

By substitution, using Equation (8.53), *Equation* (8.54) *becomes:*

$$\rho_N\theta_N\Big(1-\sum_{j=1}^{N}P(N,j)\Big) = \sum_{j=1}^{N}\lambda_i$$

$$\Rightarrow\quad \rho_N = \frac{\sum_{j=1}^{N}\lambda_i}{\theta_N(1-\sum_{j=1}^{N}P(N,j))}. \tag{8.55}$$

Finally by substitution of Equation (8.55), *Equation* (8.53) *becomes*

$$\forall\ i \in \{1,\ldots,N-1\},\quad \rho_i = \frac{P(N,i)\sum_{j=1}^{N}\lambda_j}{\theta_i(1-\sum_{j=1}^{N}P(N,j))} + \frac{\lambda_i}{\theta_i} \tag{8.56}$$

We have defined C(i) in Equation (8.52) *in order to express Equation* (8.55) *and* (8.56) *in a single equation for all* $i \in \{1,\ldots,N\}$, *which is Equation* (8.51). *The proof is complete.*

Lemma 8.4.5 *The DPs stability condition is* $\rho_i < 1$ *for all* i. *Then, in a star EPN, we have for all* $i \in \{1,\ldots,N\}$

$$\Phi_i > \frac{(\gamma_i + \mu_{i,1})}{\alpha\mu_{i,1}}\left[\frac{C(i)\sum_{j=1}^{N}\lambda_j}{1-\sum_{j=1}^{N}P(N,j)} + \lambda_i 1_{i<N}\right]. \tag{8.57}$$

Proof. *Recall that* $\forall\ i \in \{1,\ldots,N\}$, $\theta_i = \sum_{n=1}^{K}(1-p_{i,n})\mu_{i,n}\beta_{i,n}$ *(see Lemma 8.2.4). Also from Lemma 8.2.1 we have for all* $i \in \{1,\ldots,N\}$, $\sum_{n=1}^{K}(1-p_{i,n})\mu_{i,n}\beta_{i,n} = \alpha_i - \gamma_i\beta_{i,1}$. *Therefore* $\theta_i = \alpha_i - \gamma_i\beta_{i,1}$. *By the substitution of* $\beta_{i,1}$ *and* α_i *(Equation* (8.45)*), we obtain*

$$\theta_i = \frac{\alpha\Phi_i\mu_{i,1}}{\gamma_i + \mu_{i,1}}. \tag{8.58}$$

Using Equation (8.58) *in Equation* (8.51), *we get*

$$\rho_i = \frac{\gamma_i + \mu_{i,1}}{\alpha\Phi_i\mu_{i,1}}\left[\frac{C(i)\sum_{j=1}^{N}\lambda_j}{1-\sum_{j=1}^{N}P(N,j)} + \lambda_i 1_{i<N}\right]. \tag{8.59}$$

DPs stability conditions are $\forall\ i \in \{1,\ldots,N\}$, $\rho_i < 1$. *Hence, by combining Equation* (8.59) *and that* $\rho_i < 1$, *we obtain Equation* (8.57), *and the proof is complete.*

The Upper bound of Φ_i remains the same for any EPN topology (Lemma 8.4.1). The lower bound of Φ_i clearly depends on the topology, as the resolution of the point fix system (Equation (8.3)) can be trivial for some topologies. However, we can solve the point fix system for any topology using an iterative algorithm or using the fundamental matrix method which consists on calculating the matrix $(I-P)^{-1}$ (as stated in the point fix lemma, Lemma 8.2.5).

Remember that, we are aiming to assign Δ panels over the N cells of the network. It means that we are looking for the distribution of $(\Phi_1,\ldots,\Phi_N)$ that minimize the mean waiting time of a data packet in the network. Let us now study the domain of definition for the star EPN topology. The set of possible values for $(\Phi_1,\ldots,\Phi_N)$ comes from the constraints stated in Lemma 8.4.1 and Lemma 8.4.5 : $\forall i \in \{1,\ldots,N\}$

$$\begin{cases} \Phi_i < \frac{\gamma_i+\mu_{i,1}}{\alpha}\left(1+\sum_{n=2}^{K}\frac{\mu_{i,1}}{\mu_{i,n}}\prod_{r=1}^{n-1}p_{i,r}\right)^{-1}, \\ \Phi_i > \frac{(\gamma_i+\mu_{i,1})}{\alpha\mu_{i,1}}\left[\frac{C(i)\sum_{j=1}^{N}\lambda_j}{1-\sum_{j=1}^{N}P(N,j)}+\lambda_i 1_{i<N}\right] > 0, \\ \sum_{i=1}^{N}\Phi_i = \Delta. \end{cases}$$

We use ρ_i in Equation (8.59), and the expression of the total mean waiting time $E[X]=\sum_{i=1}^{N}\frac{\rho_i}{1-\rho_i}$, and Little's law $\frac{E[X]}{\sum_{i=1}^{N}\lambda_i}$ to derive the mean response time function to minimize: $f(\Phi_1,\ldots,\Phi_N)=\sum_{i=1}^{N}f(\phi_i)$, which is equal to

$$\begin{aligned} f(\Phi_1,\ldots,\Phi_N) &= \sum_{i=1}^{N}(\gamma_i+\mu_{i,1})\left[\frac{C(i)\sum_{j=1}^{N}\lambda_j}{1-\sum_{j=1}^{N}P(N,j)}+\lambda_i 1_{i<N}\right] \\ &\left(\alpha\Phi_i\mu_{i,1}-(\gamma_i+\mu_{i,1})\left[\frac{C(i)\sum_{j=1}^{N}\lambda_j}{1-\sum_{j=1}^{N}P(N,j)}+\lambda_i 1_{i<N}\right]\right)^{-1}\left(\sum_{i=1}^{N}\lambda_i\right)^{-1}. \end{aligned} \tag{8.60}$$

For the resolution of this optimization problem, we still perform the heuristic algorithm (Algorithm 8) and the Gradient descent algorithm (Algorithm 9). However, we have to use Lemma 8.4.5 instead of Lemma 8.4.3 in the step 1) of both algorithms. Other steps remains the same.

8.4.2.2 A star sensor network of N = 10 cells

Next, we consider a star EPN network to model a LoRa network. The system contains $N=10$ cells (Fig. 8.9). The service is still an Erlang process with $K=4$ phases. Cells that receive data packets are 1,2,3,4,5,6,7,8,9. Arrival rate of DPs, respectively in each cell, are $\{1, 1, 1, 2, 2, 2, 3, 3, 3, 0\}$. We suppose that service rate and battery leakage rate of EPs are $\forall\ i \in \{1,\ldots,9\}, \mu_i = 40$ and $\gamma_i = 4$. It is assumed that the service rate in the gateway cell (cell 10) is 1.5 times faster than that in end-devices, also battery leakage is less frequent (i.e. $\mu_{10}=100$ and $\gamma_{10}=2$). EPs rate generated by a single solar panel is $\alpha=2$. Notice that using these parameters the total number of panels to use in this topology should be in $\{26,57\}$. Hence, we

suppose that $\Delta = 40$ panels. The non-zero transition probabilities in the routing matrix P are $\forall i \in \{1,\ldots,9\}$ $P(i,N) = 1$ and $P(N,i) = 0.01$.

The trend in the results for the mean waiting time of a DP (Fig. 8.10) and the total loss rate of EPs (Fig. 8.11) remains the same as in the previous examples. This means that the mean waiting time decreases and the total loss rate of EPs increases, while increasing the amount of energy affected to the batteries. Furthermore, the heuristic and Gradient descent algorithm shows nearly the same distribution for the panels. Three different distributions ($\Delta = \{27,28,29\}$) out of 15 (see Table 8.3). And the difference in waiting time in these cases does not exceed $10^{-}4$. Note, however, that the Gradient descent algorithm provides the optimal panel distribution. (i.e. we have numerically verified the solution of the Gradient descent with an exhaustive algorithm). The verifications go up to $\Delta = 50$ since the exhaustive algorithm is exponential.

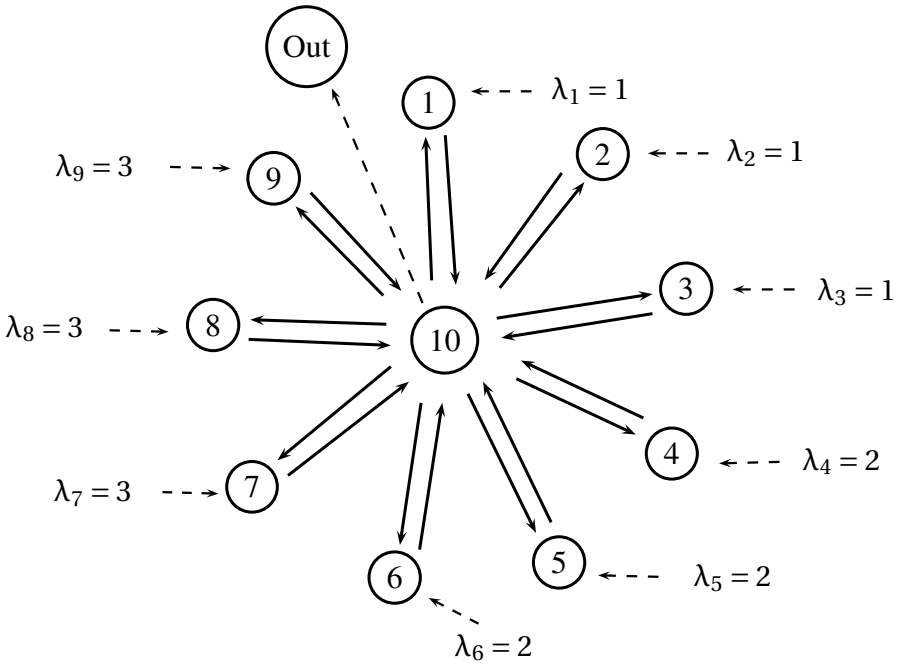


Figure 8.9: A star EPN topology of $N = 10$ cells.

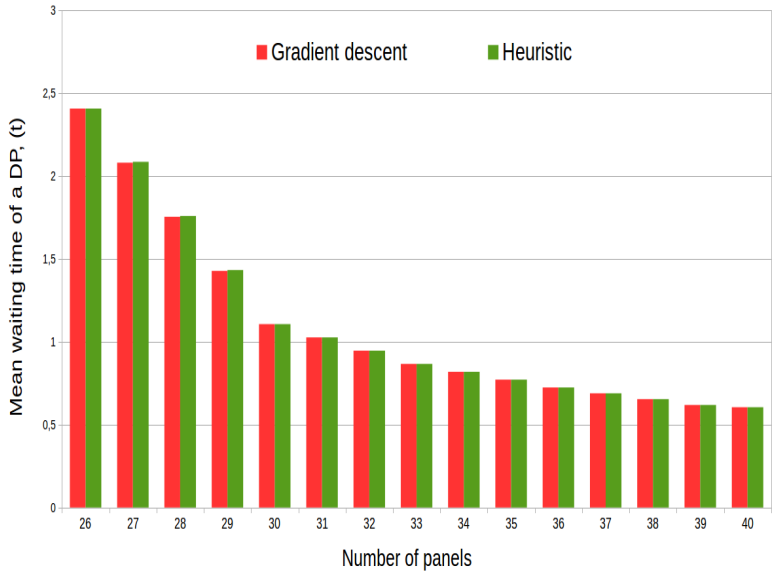


Figure 8.10: Mean waiting time of a DP in a star EPN of N = 10 cells: Heuristic solution Vs Gradient descent solution .

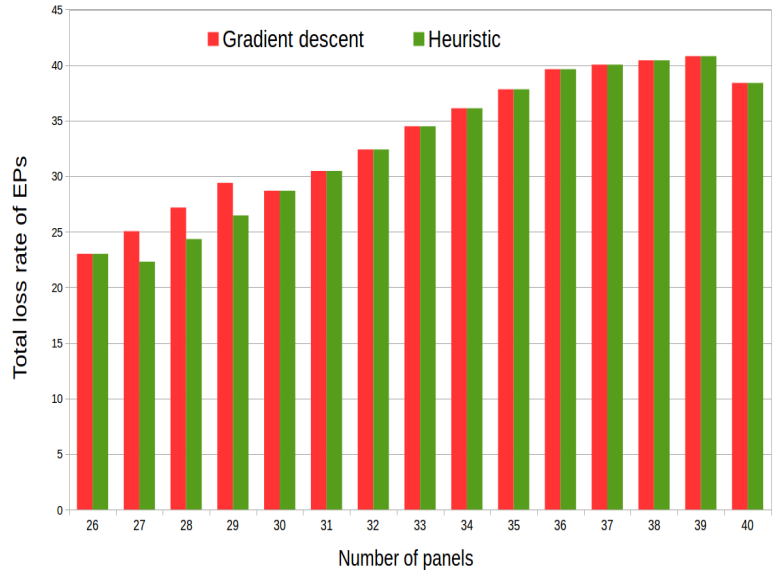


Figure 8.11: Loss rate of EPs in a star EPN of N = 10 cells: Heuristic solution Vs Gradient descent solution .

Δ	Φ_1	Φ_2	Φ_3	Φ_4	Φ_5	Φ_6	Φ_7	Φ_8	Φ_9	Φ_{10}
26	1	1	1	2	2	2	2	2	2	11
27	1	1	1	2	2	2	2 & 3	2	2	12 & 11
28	1	1	1	2	2	2	3	2 & 3	2	12 & 11
29	1	1	1	2	2	2	3	3	2 & 3	12 & 11
30	1	1	1	2	2	2	3	3	3	12
31	2	1	1	2	2	2	3	3	3	12
32	2	2	1	2	2	2	3	3	3	12
33	2	2	2	2	2	2	3	3	3	12
34	2	2	2	3	2	2	3	3	3	12
35	2	2	2	3	3	2	3	3	3	12
36	2	2	2	3	3	3	3	3	3	12
37	2	2	2	3	3	3	4	3	3	12
38	2	2	2	3	3	3	4	4	3	12
39	2	2	2	3	3	3	4	4	4	12
40	2	2	2	3	3	3	5	4	4	12

Table 8.3: Distribution of Δ panels using Heuristic and Gradient descent algorithm, $N = 10$ cells.

Chapter 9

Processor Sharing G-queues with inert customers and catastrophes: a model for server aging and rejuvenation

Contents

9.1 Introduction

In this study, we consider a G-network with Processor Sharing (PS) queues with inert customers and signals. Inert customers were introduced by Dao Thi and her co-authors in [DFT13]. Inert customers are customers which do not use the service capacity but they stay in the queue until they interact with the signal. More precisely, in a PS queue, the service is shared among all the customers whatever they are usual customers or inert customers. But the inert customers do not use the server and this part of the service capacity is wasted. The signal is a catastrophe or a disaster: it removes all the customers (both inert or usual). Note that the arrival rate of signal must be positive to obtain a stationary system. Indeed, the signal is the only possibility to let the inert customers leave the queue. We depict in Fig. 9.1 a typical sample-path for a queue with both usual and inert customers obtained by the simulation tool in XBorne [Fou+16]. We will see that the queue is less and less efficient with aging until it is refreshed by the signal.

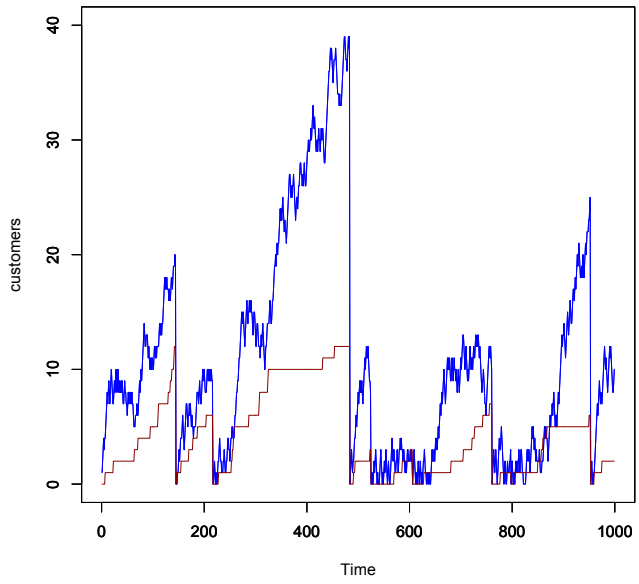


Figure 9.1: Sample-paths for a queue with catastrophes and customers (usual in blue, inert in red). Parameters: $\lambda^I = 0.1$, $\lambda^S = 0.01$, $\lambda^U = 1.0$, $\mu = 1.0$.

This work is organized as follows. In Section 9.2, we introduce networks with inert customers and catastrophes signals. We state that the steady-state distribution has a product form if the chain is ergodic (the proof based on the analysis of the Kolmogorov equation at steady-state in postpone in Appendix A for the sake of readability). In Section 9.3, we prove that under some technical assumptions, the flow equations have a solution. We present in Section 9.4 , a more complex catastrophe signal which only deletes the inert customers (some of them).

9.2 Model and product-form steady-state distribution

We consider an open network with N processor sharing queues and two types of customers: usual customer and inert customers. Furthermore, the queues can receive signals from the outside or sent by another queue at the completion of service for an usual customer (see Fig. 9.2).

Both types of customers arrive from the outside at queue i according to Poisson processes with rate λ_i^U for usual customers and λ_i^I for inert ones. Usual customers receive service at queue i with a exponentially distributed duration with rate μ_i. Inert customers have a service rate equal to 0: they do not receive service. However a part of the service capacity of the server is given to inert customers according to the PS discipline. Therefore it is wasted.

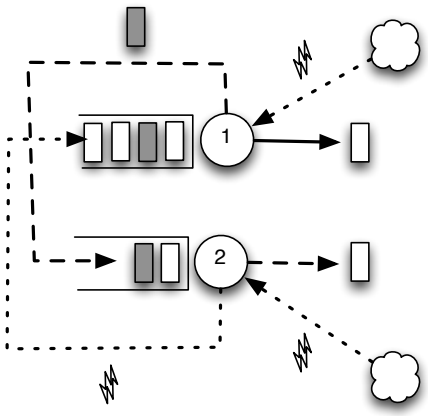


Figure 9.2: Two PS-queue with usual customers (white boxes), inert customers (grey boxes) and catastrophe signals.

At the completion of its service at queue i, an usual customer may join queue j either as an usual customer (routing matrix $P(i,j)$), or a signal (routing matrix $C(i,j)$) or an inert customer (routing matrix $E(i,j)$), or it can leave the network with probability d_i. Of course we have for all i,

$$d_i + \sum_{j=1}^{N} P(i,j) + \sum_{j=1}^{N} E(i,j) + \sum_{j=1}^{N} C(i,j) = 1.$$

We also assume that there is no loop in the routing matrices: for all i

$$P(i,i) = 0, \quad E(i,i) = 0, \quad C(i,i) = 0.$$

Signals may also arrive from the outside following Poisson processes of rate λ_i^S for queue i. A signal entering queue i deletes all the customers present in the queue irrespective of their types. A signal is never queued. It disappear immediately after its arrival. Such a signal has been previously studied in the literature [FKQ95;

KA00; DK01]. It is a particular case of the batch deletion of customers proposed in [Gel93b].

Note that the open topology is mandatory because in a closed queuing network with catastrophes, the customers disappear and all the queues are empty at steady-state with probability one.

We consider an open network with N queues. The state of queue i is $x_i = (x_i^{\mathrm{U}}, x_i^{\mathrm{I}})$. Under the classical assumptions we have presented, $(x)_t = (x_1, ..., x_i, ...x_{\mathrm{N}})_t$ is a Markov chain.

Let us first introduce some notation. Let:

- $||x_i|| = x_i^{\mathrm{U}} + x_i^{\mathrm{I}}$.
- e_i^{U} is a vector whose all entries are equal to 0 except entry (i, U) which is equal to 1.
- Similarly, e_i^{I} is a vector whose all entries are equal to 0 except entry (i, I) which is equal to 1.

Theorem 9.2.1 *Assume that the Markov chain $(x)_t = (x_1, ..., x_i, ...x_{\mathrm{N}})_t$ is ergodic. If the following flow equations have a solution such that $\rho_i^{\mathrm{U}} + \rho_i^{\mathrm{I}} < 1$ for all i,*

$$\rho_i^{\mathrm{U}} = \frac{\lambda_i^{\mathrm{U}} + \sum_j \mu_j \rho_j^{\mathrm{U}} \mathrm{P}(j,i)}{\mu_i + (\lambda_i^{\mathrm{S}} + \sum_j \mu_j \rho_j^{\mathrm{U}} \mathrm{C}(j,i))\mathrm{R}_i}, \tag{9.1}$$

and

$$\rho_i^{\mathrm{I}} = \frac{\lambda_i^{\mathrm{I}} + \sum_j \mu_j \rho_j^{\mathrm{U}} \mathrm{E}(j,i)}{(\lambda_i^{\mathrm{S}} + \sum_j \mu_j \rho_j^{\mathrm{U}} \mathrm{C}(j,i))\mathrm{R}_i}, \tag{9.2}$$

where

$$\mathrm{R}_i = \sum_{k=0}^{\infty} (\rho_i^{\mathrm{U}} + \rho_i^{\mathrm{I}})^k = (1 - \rho_i^{\mathrm{U}} + \rho_i^{\mathrm{I}})^{-1}, \tag{9.3}$$

then the steady state distribution has a product form solution:

$$\pi(x) = \prod_{i=1}^{\mathrm{N}} (1 - \rho_i^{\mathrm{U}} - \rho_i^{\mathrm{I}}) \frac{||x_i||!}{x_i^{\mathrm{U}}!\ x_i^{\mathrm{I}}!} (\rho_i^{\mathrm{U}})^{x_i^{\mathrm{U}}} (\rho_i^{\mathrm{I}})^{x_i^{\mathrm{I}}} \tag{9.4}$$

The proof is based on the analysis of the global balance equation. The Kolmogorov equation at steady-state is:

$$\begin{aligned}
\pi(x) \left(\sum_i (\lambda_i^{\mathrm{U}} + \lambda_i^{\mathrm{I}} + \lambda_i^{\mathrm{S}}) + \sum_i \frac{x_i^{\mathrm{U}} \mu_i}{||x_i||} 1_{||x_i||>0} \right) = & \sum_i \pi(x - e_i^{\mathrm{U}}) \lambda_i^{\mathrm{U}} 1_{x_i^{\mathrm{U}}>0} \\
+ & \sum_i \pi(x - e_i^{\mathrm{I}}) \lambda_i^{\mathrm{I}} 1_{x_i^{\mathrm{I}}>0} \\
+ & \sum_i \pi(x + e_i^{\mathrm{U}}) \frac{(x_i^{\mathrm{U}}+1)\mu_i}{||x_i + e_i^{\mathrm{U}}||} d_i \\
+ & \sum_i \sum_j \pi(x + e_i^{\mathrm{U}} - e_j^{\mathrm{U}}) \frac{(x_i^{\mathrm{U}}+1)\mu_i}{||x_i + e_i^{\mathrm{U}}||} \mathrm{P}(i,j) 1_{x_j^{\mathrm{U}}>0} \\
+ & \sum_i \sum_j \pi(x + e_i^{\mathrm{U}} - e_j^{\mathrm{I}}) \frac{(x_i^{\mathrm{U}}+1)\mu_i}{||x_i + e_i^{\mathrm{U}}||} \mathrm{E}(i,j) 1_{x_j^{\mathrm{I}}>0} \\
+ & \sum_i \lambda_i^{\mathrm{S}} \sum_{a \geq 0} \sum_{b \geq 0} \pi(x + e_i^{\mathrm{U}} a + e_i^{\mathrm{I}} b) 1_{||x_i||=0} \\
+ & \sum_i \mu_i \sum_j \mathrm{C}(i,j) \sum_{a \geq 0} \sum_{b \geq 0} \pi(x + e_i^{\mathrm{U}} + e_j^{\mathrm{U}} a + e_j^{\mathrm{I}} b) 1_{||x_j||=0}
\end{aligned}$$

Remark that this equation includes null transitions, on both sides of the equation, when the queue size is 0. For the sake of readability the proof is postponed to Appendix A.

These queues exhibit a very interesting behavior which is depicted in Fig. 9.3. As mentioned earlier, the part of the capacity given by the servers to the inert customers is lost, thus one can observe a waste of the server power. At state $(x^{\mathrm{U}}, x^{\mathrm{I}})$, the lost part is $\frac{x^{\mathrm{I}}}{x^{\mathrm{U}}+x^{\mathrm{I}}}$. In Fig. 9.3, we have depicted a sample path of the remaining part of the service capacity. When $x^{\mathrm{U}}+x^{\mathrm{I}}=0$, we have set this lost part to 0 (or equivalently, the remaining part is equal to 1). The service capacity evolves with time with a decreasing trend which is due to the increasing number of inert customers. Clearly, the remaining service which is equal to $\frac{x^{\mathrm{U}}}{x^{\mathrm{U}}+x^{\mathrm{I}}}$ decreases with the number of inert customers (i.e. x^{I}). As the number of inert customers increases with time until the next catastrophe, we obtain a queue where the service capacity decreases with age until a rejuvenation (i.e. the catastrophe) refreshes the server and its capacity. The small fluctuations are due to the number of usual customers which increases or decreases as a result of arrivals and departures. Finally at time 144, a signal occurs and all the customers are deleted. Another signal arrives at time 217 and also clears the queue. However these two signals do not have exactly the same effect on the future of the sample-path.

The signal arriving at time 144 empties the queue. But the next event is an arrival of an inert customer and the remaining capacity of service jumps to 0 as the queue only contains one inert customer. When an usual customer arrives, the capacity jumps at 0.5 as the queue population is now one inert and one usual customer sharing the capacity.

At time 217, the signal is followed by the arrival of several usual customers (i.e. exactly 3). Thus the capacity stays at 1 for a short period of time before decreasing when the first inert customer arrives.

Note that even if these queues exhibit a very unusual behavior, they are still PS queues with a well-known steady-state distribution. The only difference is described by the flow equation, not by the distribution. Therefore the usual formulas for PS queues are still valid and we obtain the average number of usual customers at queue i by:

$$\mathrm{E}[\mathrm{N}_i] = \frac{\rho_i^{\mathrm{U}} + \rho_i^{\mathrm{I}}}{1 - \rho_i^{\mathrm{U}} - \rho_i^{\mathrm{I}}}.$$

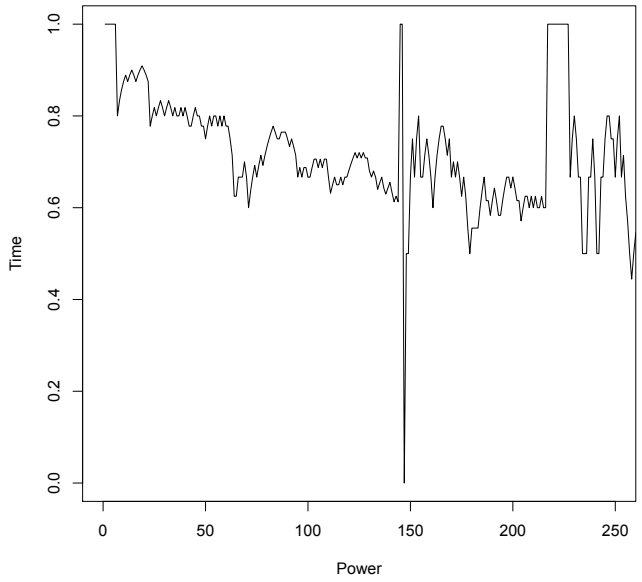


Figure 9.3: Sample-paths for the effective capacity for a queue with catastrophes and both types of customers. Same parameters as in Fig. 9.1.

9.3 Stability

Clearly Equations 9.1 to 9.3 define a non linear fixed point system. Due to the non linearity, existence of a fixed point solution is not a trivial question and must be addressed. Furthermore one may expect that the arrival of catastrophes make the queueing process stationary.

Theorem 9.3.1 *Assume that the chain is ergodic. If* $\lambda_i^S > 0$ *and* $\lambda_i^U > 0$ *for all queue* i*, then the solution of the fixed point exists and* $\rho_i^U + \rho_i^I < 1$ *for all* i*.*

Proof. *We first prove the existence with Brouwer's theorem. Let us define operator* F *on* $(R^+)^{2N}$ *by its components* F_i^U *and* F_i^I*:*

- *if* $F_i^U + F_i^I < 1$

$$F_i^U(F) = \frac{\lambda_i^U + \sum_j \mu_j F_j^U P(j,i)}{\mu_i + (\lambda_i^S + \sum_j \mu_j F_j^U C(j,i))R_i},$$

$$F_i^I(F) = \frac{\lambda_i^I + \sum_j \mu_j F_j^U E(j,i)}{(\lambda_i^S + \sum_j \mu_j F_j^U C(j,i))R_i}$$

  and

$$R_i = (1 - F_i^U - F_i^I)^{-1}$$

- *and* $F_i^U = F_i^I = 0$ *otherwise*

We investigate the fixed points of F. *Remark that the system of flow equations and* F *define the same system when* $\rho_i^U + \rho_i^I < 1$. *We now define a new operator, say* G, *on* $(R^+)^{2N}$ *by its components:*

$$G_i^U(F) = \frac{\lambda_i^U + \sum_j \mu_j F_j^U P(j,i)}{\mu_i},$$

and

$$G_i^I(F) = \frac{\lambda_i^I + \sum_j \mu_j F_j^U E(j,i)}{\lambda_i^S}$$

Clearly, $F \preceq G$. *It is sufficient to take into account that* $\mu_j > 0$, $F_j^U \geq 0$, $C(j,i) \geq 0$, *and* $R_i \geq 0$ *or* $R_i \geq 1$ *in Equations 9.1 and 9.2.*

Furthermore, operator G *is non negative, contracting, continuous and* (G_i^U) *is associated with a classical Jackson network. Thus,* G *has a fixed point* $\hat{f}$.

Now, we define $\mathcal{S}$ *as a subset of* $(R^+)^{2N}$ *as follows:*

$$\mathcal{S} = \{q \in (R^+)^{2N} : 0 \preceq q \preceq \hat{f}\}.$$

Clearly, $\mathcal{S}$ *is compact and convex. Since for all* j, $\lambda_j^U > 0$, *we have* $\hat{f} \succ 0$ *and then interior of* $\mathcal{S}$ *is not empty.*

As mentioned earlier, $F(q) \preceq G(q)$. *Furthermore* G *is non-decreasing in* $\mathcal{S}$, *so for all* q *in* $\mathcal{S}$ *we have* $G(q) \preceq G(\hat{f})$. *Combining these inequalities and the fixed point, we get:*

$$F(q) \preceq G(q) \preceq G(\hat{f}) = \hat{f}$$

and then $F(\mathcal{S}) \subseteq \mathcal{S}$.

$\mathcal{S}$ *is compact convex and has a non empty interior,* F *is continuous and* $F(\mathcal{S}) \subseteq \mathcal{S}$. F *satisfies assumptions of Brouwer's theorem [GZ81]. Thus,* F *has a fixed point.*

This sufficient condition of existence of fixed point of F *is also a sufficient condition for system of Equations 9.1, 9.2 and 9.3 to have a fixed point.*

Finally we prove that $\rho_i^U + \rho_i^I < 1$. *Clearly, if a fixed point exist with* $\rho_i^U + \rho_i^I > 1$ *for some* i, *then* $F_i^U = 0$. *But* F_i^U *cannot be equal to* 0 *for a fixed point. Indeed* $F_i^U > 0$ *as* $\lambda_i^U > 0$ *for all queue* i.

Theorem 9.3.2 *If* $\lambda_i^S > 0$ *and* $\lambda_i^U > 0$ *for all queue* i, *then the chain is ergodic.*

Proof. *As the rates are bounded, the chain is uniformizable and we consider the embedded Markov chain. Remember that* N *is the number of queues. At any time, there is a positive probability that the next* N *events are signals sent to the* N *queues and which empty all the queues. Therefore the chain is ergodic.*

9.4 Partial rejuvenation

We now assume that the effect of the signal is to delete some of the inert customers present in the queue. However, it does not delete all of them with probability 1. The probability of deletion is state dependent. It depends on the number of

inert customers and the number of usual customers. Let the state be $(x^{\mathrm{U}}, x^{\mathrm{I}})$, the probability of a deletion of m customer is given by the following probability:

$$Pr(m\ deletions\ given\ state(x^{\mathrm{U}}, x^{\mathrm{I}})) = \beta(x^{\mathrm{U}}, x^{\mathrm{I}}, m) \tag{9.5}$$

Of course this is only defined for $m \leq x^{\mathrm{I}}$ and we have: $\sum_{m=0}^{x^{\mathrm{I}}} \beta(x^{\mathrm{U}}, x^{\mathrm{I}}, m) = 1$. We prove, in the following theorem, that for a well defined distribution β, the steady-state distribution has a multiplicative form. For the sake of readability, we assume that matrix $\mathbf{C}$ is zero and the signals only arrive from the outside.

Theorem 9.4.1 *Assume that for all queue, the effect of the signal on inert customers is given by probability:*

$$\beta(x^{\mathrm{U}}, x^{\mathrm{I}} + m, m) = \frac{x^{\mathrm{U}}}{x^{\mathrm{U}} + x^{\mathrm{I}}} \left(\prod_{k=1}^{m} \frac{x^{\mathrm{I}} + k}{x^{\mathrm{U}} + x^{\mathrm{I}} + k} \right) = \frac{x^{\mathrm{U}}}{x^{\mathrm{U}} + x^{\mathrm{I}}} \frac{(x^{\mathrm{I}} + m)!(x^{\mathrm{U}} + x^{\mathrm{I}})!}{(x^{\mathrm{U}} + x^{\mathrm{I}} + m)! x^{\mathrm{I}}!} 1_{||x||>0}, \tag{9.6}$$

and $\beta(0,0,0) = 1$. *Assume that the Markov chain* $(x)_t = (x_1, ..., x_i, ...x_{\mathrm{N}})_t$ *is ergodic. If the following flow equations have a solution such that* $\rho_i^{\mathrm{U}} + \rho_i^{\mathrm{I}} < 1$ *for all* i,

$$\rho_i^{\mathrm{U}} = \frac{\lambda_i^{\mathrm{U}} + \sum_j \mu_j \rho_j^{\mathrm{U}} \mathrm{P}(j,i)}{\mu_i + \lambda_i^{\mathrm{S}}}, \tag{9.7}$$

and

$$\rho_i^{\mathrm{I}} = \frac{\lambda_i^{\mathrm{I}} + \sum_j \mu_j \rho_j^{\mathrm{U}} \mathrm{E}(j,i)}{\frac{\lambda_i^{\mathrm{S}}}{1-\rho_i^{\mathrm{I}}}}, \tag{9.8}$$

then the steady state distribution has a product form solution:

$$\pi(x) = \prod_{i=1}^{\mathrm{N}} (1 - \rho_i^{\mathrm{U}} - \rho_i^{\mathrm{I}}) \frac{||x_i||!}{x_i^{\mathrm{U}}!\ x_i^{\mathrm{I}}!} (\rho_i^{\mathrm{U}})^{x_i^{\mathrm{U}}} (\rho_i^{\mathrm{I}})^{x_i^{\mathrm{I}}} \tag{9.9}$$

Proof. *Once again, the proof is based on the analysis of the Kolmogorov equation at steady-state:*

$$\begin{aligned}
\pi(x) \left(\sum_i (\lambda_i^{\mathrm{U}} + \lambda_i^{\mathrm{I}} + \lambda_i^{\mathrm{S}}) + \sum_i \frac{x_i^{\mathrm{U}} \mu_i}{||x_i||} 1_{||x_i||>0} \right) = &\ \sum_i \pi(x - e_i^{\mathrm{U}}) \lambda_i^{\mathrm{U}} 1_{x_i^{\mathrm{U}}>0} \\
+ &\ \sum_i \pi(x - e_i^{\mathrm{I}}) \lambda_i^{\mathrm{I}} 1_{x_i^{\mathrm{I}}>0} \\
+ &\ \sum_i \pi(x + e_i^{\mathrm{U}}) \frac{(x_i^{\mathrm{U}}+1)\mu_i}{||x_i + e_i^{\mathrm{U}}||} d_i \\
+ &\ \sum_i \sum_j \pi(x + e_i^{\mathrm{U}} - e_j^{\mathrm{U}}) \frac{(x_i^{\mathrm{U}}+1)\mu_i}{||x_i + e_i^{\mathrm{U}}||} \mathrm{P}(i,j) 1_{x_j^{\mathrm{U}}>0} \\
+ &\ \sum_i \sum_j \pi(x + e_i^{\mathrm{U}} - e_j^{\mathrm{I}}) \frac{(x_i^{\mathrm{U}}+1)\mu_i}{||x_i + e_i^{\mathrm{U}}||} \mathrm{E}(i,j) 1_{x_j^{\mathrm{I}}>0} \\
+ &\ \sum_i \lambda_i^{\mathrm{S}} \sum_{m \geq 0} \pi(x + e_i^{\mathrm{I}} m) \beta(x_i^{\mathrm{U}}, x_i^{\mathrm{I}} + m, m)
\end{aligned}$$

Once again this equation includes null transitions, on both sides of the equation, when the queue size is 0. *We use the same arguments as in the previous proof of*

product form.

$$\begin{aligned}\sum_i(\lambda_i^{\mathrm{U}}+\lambda_i^{\mathrm{I}}+\lambda_i^{\mathrm{S}})+\sum_i\frac{x_i^{\mathrm{U}}\mu_i}{||x_i||}1_{||x_i||>0} = &\quad \sum_i\lambda_i^{\mathrm{U}}\frac{x_i^{\mathrm{U}}}{\rho_i^{\mathrm{U}}||x_i||}1_{||x_i||>0}\\ &+\quad \sum_i\lambda_i^{\mathrm{I}}\frac{x_i^{\mathrm{I}}}{\rho_i^{\mathrm{I}}||x_i||}1_{||x_i||>0}\\ &+\quad \sum_i\rho_i^{\mathrm{U}}\mu_i d_i\\ &+\quad \sum_i\sum_j\mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{U}}}{\rho_j^{\mathrm{U}}||x_j||}\mathrm{P}(i,j)1_{||x_j||>0}\\ &+\quad \sum_i\sum_j\mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{I}}}{\rho_j^{\mathrm{I}}||x_j||}\mathrm{E}(i,j)1_{||x_j||>0}\\ &+\quad \sum_i\lambda_i^{\mathrm{S}}\sum_{m\geq 0}(\rho_i^{\mathrm{I}})^m\frac{(||x_i||+m)!x_i^{\mathrm{I}}!}{||x_i||!\ \ (x_i^{\mathrm{I}}+m)!}\beta(x_i^{\mathrm{U}},x_i^{\mathrm{I}}+m,m)\end{aligned}$$

We use the definition of the distribution β*. After substitution, we obtain:*

$$\begin{aligned}\sum_i(\lambda_i^{\mathrm{U}}+\lambda_i^{\mathrm{I}}+\lambda_i^{\mathrm{S}})+\sum_i\frac{x_i^{\mathrm{U}}\mu_i}{||x_i||}1_{||x_i||>0} = &\quad \sum_i\lambda_i^{\mathrm{U}}\frac{x_i^{\mathrm{U}}}{\rho_i^{\mathrm{U}}||x_i||}1_{||x_i||>0}\\ &+\quad \sum_i\lambda_i^{\mathrm{I}}\frac{x_i^{\mathrm{I}}}{\rho_i^{\mathrm{I}}||x_i||}1_{||x_i||>0}\\ &+\quad \sum_i\rho_i^{\mathrm{U}}\mu_i d_i\\ &+\quad \sum_i\sum_j\mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{U}}}{\rho_j^{\mathrm{U}}||x_j||}\mathrm{P}(i,j)1_{||x_j||>0}\\ &+\quad \sum_i\sum_j\mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{I}}}{\rho_j^{\mathrm{I}}||x_j||}\mathrm{E}(i,j)1_{||x_j||>0}\\ &+\quad \sum_i\lambda_i^{\mathrm{S}}\frac{x_i^{\mathrm{U}}}{||x_i||}\sum_{m\geq 0}(\rho_i^{\mathrm{I}})^m1_{||x_i||>0}\\ &+\quad \sum_i\lambda_i^{\mathrm{S}}\sum_{m\geq 0}(\rho_i^{\mathrm{I}})^m1_{||x_i||=0}\end{aligned}$$

As $\rho_i^{\mathrm{I}}<1$*,* $\sum_{m\geq 0}(\rho_i^{\mathrm{I}})^m=(1-\rho_i^{\mathrm{I}})^{-1}$*. We make the same decomposition of* λ_i^{S} *into* $\lambda_i^{\mathrm{S}}\frac{x_i^{\mathrm{U}}}{||x_i||}+\lambda_i^{\mathrm{S}}\frac{x_i^{\mathrm{I}}}{||x_i||}$*. Furthermore,* $\frac{x_i^{\mathrm{U}}}{||x_i||}=1-\frac{x_i^{\mathrm{I}}}{||x_i||}$ *and we move the negative part to the left hand side.*

$$\begin{aligned}\sum_i(\lambda_i^{\mathrm{U}}+\lambda_i^{\mathrm{I}}+\lambda_i^{\mathrm{S}})+\sum_i\frac{\lambda_i^{\mathrm{S}}}{1-\rho_i^{\mathrm{I}}}\frac{x_i^{\mathrm{I}}}{||x_i||}1_{||x_i||>0}+\sum_i\frac{x_i^{\mathrm{U}}(\mu_i+\lambda_i^{\mathrm{S}})}{||x_i||}1_{||x_i||>0} = &\quad \sum_i\lambda_i^{\mathrm{U}}\frac{x_i^{\mathrm{U}}}{\rho_i^{\mathrm{U}}||x_i||}1_{||x_i||>0}\\ &+\quad \sum_i\lambda_i^{\mathrm{I}}\frac{x_i^{\mathrm{I}}}{\rho_i^{\mathrm{I}}||x_i||}1_{||x_i||>0}\\ &+\quad \sum_i\rho_i^{\mathrm{U}}\mu_i d_i\\ &+\quad \sum_i\sum_j\mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{U}}}{\rho_j^{\mathrm{U}}||x_j||}\mathrm{P}(i,j)1_{||x_j||>0}\\ &+\quad \sum_i\sum_j\mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{I}}}{\rho_j^{\mathrm{I}}||x_j||}\mathrm{E}(i,j)1_{||x_j||>0}\\ &+\quad \sum_i\lambda_i^{\mathrm{S}}\frac{1}{1-\rho_i^{\mathrm{I}}}1_{||x_i||>0}\\ &+\quad \sum_i\lambda_i^{\mathrm{S}}\frac{1}{1-\rho_i^{\mathrm{I}}}1_{||x_i||=0}\end{aligned}$$

The last two terms of the r.h.s are gathered and we cancel the term λ_i^{S} *which is*

present on both sides on the equation.

$$\begin{aligned}\sum_i(\lambda_i^{\mathrm{U}}+\lambda_i^{\mathrm{I}})+\sum_i\frac{\lambda_i^{\mathrm{S}}}{1-\rho_i^{\mathrm{I}}}\frac{x_i^{\mathrm{I}}}{||x_i||}1_{||x_i||>0}+\sum_i\frac{x_i^{\mathrm{U}}(\mu_i+\lambda_i^{\mathrm{S}})}{||x_i||}1_{||x_i||>0} = &\sum_i\lambda_i^{\mathrm{U}}\frac{x_i^{\mathrm{U}}}{\rho_i^{\mathrm{U}}||x_i||}1_{||x_i||>0}\\ &+\sum_i\lambda_i^{\mathrm{I}}\frac{x_i^{\mathrm{I}}}{\rho_i^{\mathrm{I}}||x_i||}1_{||x_i||>0}\\ &+\sum_i\rho_i^{\mathrm{U}}\mu_i d_i\\ &+\sum_i\sum_j\mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{U}}}{\rho_j^{\mathrm{U}}||x_j||}\mathrm{P}(i,j)1_{||x_j||>0}\\ &+\sum_i\sum_j\mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{I}}}{\rho_j^{\mathrm{I}}||x_j||}\mathrm{E}(i,j)1_{||x_j||>0}\\ &+\sum_i\lambda_i^{\mathrm{S}}\frac{\rho_i^{\mathrm{I}}}{1-\rho_i^{\mathrm{I}}}\end{aligned}$$

We decompose into three equations:

$$\sum_i(\lambda_i^{\mathrm{U}}+\lambda_i^{\mathrm{I}})=\sum_i\rho_i^{\mathrm{U}}\mu_i d_i+\sum_i\lambda_i^{\mathrm{S}}\frac{\rho_i^{\mathrm{I}}}{1-\rho_i^{\mathrm{I}}} \tag{9.10}$$

$$\sum_i\frac{\lambda_i^{\mathrm{S}}}{1-\rho_i^{\mathrm{I}}}\frac{x_i^{\mathrm{I}}}{||x_i||}1_{||x_i||>0}=\sum_i\lambda_i^{\mathrm{I}}\frac{x_i^{\mathrm{I}}}{\rho_i^{\mathrm{I}}||x_i||}1_{||x_i||>0}+\sum_i\sum_j\mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{I}}}{\rho_j^{\mathrm{I}}||x_j||}\mathrm{E}(i,j)1_{||x_j||>0} \tag{9.11}$$

$$\sum_i\frac{x_i^{\mathrm{U}}(\mu_i+\lambda_i^{\mathrm{S}})}{||x_i||}1_{||x_i||>0}=\sum_i\lambda_i^{\mathrm{U}}\frac{x_i^{\mathrm{U}}}{\rho_i^{\mathrm{U}}||x_i||}1_{||x_i||>0}+\sum_i\sum_j\mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{U}}}{\rho_j^{\mathrm{U}}||x_j||}\mathrm{P}(i,j)1_{||x_j||>0} \tag{9.12}$$

The second and third equations are satisfied due to the definition of ρ_i^{I} *(Equation (9.8)) and* ρ_i^{U} *(Equation (9.7)). It remains to prove that the first equation is a global flow equation between the network and the outside. We consider Equation (9.8) and we multiply by the denominator:*

$$\frac{\lambda_i^{\mathrm{S}}}{1-\rho_i^{\mathrm{I}}}\rho_i^{\mathrm{I}}=\lambda_i^{\mathrm{I}}+\sum_j\mu_j\rho_j^{\mathrm{U}}\mathrm{E}(j,i).$$

We proceed the same way for Equation (9.7):

$$\mu_i\rho_i^{\mathrm{U}}=\lambda_i^{\mathrm{U}}+\sum_j\mu_j\rho_j^{\mathrm{U}}\mathrm{P}(j,i).$$

We sum for all queue index i *and we add the two equalities:*

$$\sum_i\mu_i\rho_i^{\mathrm{U}}+\sum_i\frac{\lambda_i^{\mathrm{S}}}{1-\rho_i^{\mathrm{I}}}\rho_i^{\mathrm{I}}=\sum_i\lambda_i^{\mathrm{U}}+\sum_i\sum_j\mu_j\rho_j^{\mathrm{U}}\mathrm{P}(j,i)+\sum_i\lambda_i^{\mathrm{I}}+\sum_i\sum_j\mu_j\rho_j^{\mathrm{U}}\mathrm{E}(j,i).$$

Taking into account that for all i, $\sum_j\mathrm{P}(i,j)+\sum_j\mathrm{E}(i,j)=1-d_i$, *as matrix* C *is zero, we get:*

$$\sum_i\mu_i\rho_i^{\mathrm{U}}d_i+\sum_i\frac{\lambda_i^{\mathrm{S}}}{1-\rho_i^{\mathrm{I}}}\rho_i^{\mathrm{I}}=\sum_i\lambda_i^{\mathrm{U}}+\sum_i\lambda_i^{\mathrm{I}}.$$

And we find the first flow equation and that concludes the proof.

Appendix A

Proof of product form solution for the steady state distribution

We assume that the solution as a product form solution and each queue has distribution at steady-state given by Equation 9.4. After simplification, and exchanging indices i and j in the last term of the r.h.s., we get:

$$\begin{aligned}
\sum_i(\lambda_i^{\mathrm{U}}+\lambda_i^{\mathrm{I}}+\lambda_i^{\mathrm{S}})+\sum_i \frac{x_i^{\mathrm{U}}\mu_i}{||x_i||}1_{||x_i||>0} = &\quad \sum_i \lambda_i^{\mathrm{U}}\frac{x_i^{\mathrm{U}}}{\rho_i^{\mathrm{U}}||x_i||}1_{||x_i||>0}\\
&+ \sum_i \lambda_i^{\mathrm{I}}\frac{x_i^{\mathrm{I}}}{\rho_i^{\mathrm{I}}||x_i||}1_{||x_i||>0}\\
&+ \sum_i \rho_i^{\mathrm{U}}\mu_i d_i\\
&+ \sum_i\sum_j \mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{U}}}{\rho_j^{\mathrm{U}}||x_j||}\mathrm{P}(i,j)1_{||x_j||>0}\\
&+ \sum_i\sum_j \mu_i\rho_i^{\mathrm{U}}\frac{x_j^{\mathrm{I}}}{\rho_j^{\mathrm{I}}||x_j||}\mathrm{E}(i,j)1_{||x_j||>0}\\
&+ \sum_i \lambda_i^{\mathrm{S}}\sum_{a\geq 0}\sum_{b\geq 0}(\rho_i^{\mathrm{U}})^a(\rho_i^{\mathrm{I}})^b 1_{||x_i||=0}\\
&+ \sum_j \mu_j\rho_j^{\mathrm{U}}\sum_i \mathrm{C}(j,i)\sum_{a\geq 0}\sum_{b\geq 0}\frac{(a+b)!(\rho_i^{\mathrm{U}})^a(\rho_i^{\mathrm{I}})^b}{a!b!}1_{||x_i||=0}
\end{aligned}$$

First let us consider the double summation in the last two terms of the r.h.s: $\frac{(a+b)!(\rho_i^{\mathrm{U}})^a(\rho_i^{\mathrm{I}})^b}{a!b!}$. It is well-known that, if $\rho_i^{\mathrm{U}}+\rho_i^{\mathrm{I}}<1$, then:

$$\sum_{a\geq 0}\sum_{b\geq 0}\frac{(a+b)!(\rho_i^{\mathrm{U}})^a(\rho_i^{\mathrm{I}})^b}{a!b!} = (1-\rho_i^{\mathrm{U}}-\rho_i^{\mathrm{I}})^{-1}.$$

Thus, this double summation is equal to R_i. On the r.h.s, we now write $1_{||x_i||=0} = 1 - 1_{||x_i||>0}$ and the move the negative terms on the l.h.s. to factorize.

$$\begin{aligned}
\sum_i(\lambda_i^U + \lambda_i^I + \lambda_i^S) + \sum_i \frac{x_i^U \mu_i}{||x_i||} 1_{||x_i||>0} \quad & + \sum_i(\lambda_i^S + \sum_j \mu_j \rho_j^U C(j,i)) R_i 1_{||x_i||>0} \\
= \quad & \sum_i \lambda_i^U \frac{x_i^U}{\rho_i^U ||x_i||} 1_{||x_i||>0} \\
+ \quad & \sum_i \lambda_i^I \frac{x_i^I}{\rho_i^I ||x_i||} 1_{||x_i||>0} \\
+ \quad & \sum_i \rho_i^U \mu_i d_i \\
+ \quad & \sum_i \sum_j \mu_i \rho_i^U \frac{x_j^U}{\rho_j^U ||x_j||} P(i,j) 1_{||x_j||>0} \\
+ \quad & \sum_i \sum_j \mu_i \rho_i^U \frac{x_j^I}{\rho_j^I ||x_j||} E(i,j) 1_{||x_j||>0} \\
+ \quad & \sum_i(\lambda_i^S + \sum_j \mu_j \rho_j^U C(j,i)) R_i
\end{aligned}$$

Now we decompose R_i into $R_i \frac{x_i^U}{||x_i||} + R_i \frac{x_i^I}{||x_i||}$ and we substitute in the l.h.s. After substitution, factorization and exchanging indices i and j in the fifth and sixth terms of the r.h.s., we get:

$$\begin{aligned}
&\sum_i(\lambda_i^U + \lambda_i^I + \lambda_i^S) + \sum_i(\mu_i + (\lambda_i^S + \sum_j \mu_j \rho_j^U C(j,i)) R_i) \frac{x_i^U}{||x_i||} 1_{||x_i||>0} + \\
&\sum_i(\lambda_i^S + \sum_j \mu_j \rho_j^U C(j,i)) R_i \frac{x_i^I}{||x_i||} 1_{||x_i||>0} =
\end{aligned}$$

$$\begin{aligned}
&\sum_i \lambda_i^U \frac{x_i^U}{\rho_i^U ||x_i||} 1_{||x_i||>0} \\
&+ \sum_i \lambda_i^I \frac{x_i^I}{\rho_i^I ||x_i||} 1_{||x_i||>0} \\
&+ \sum_i \rho_i^U \mu_i d_i \\
&+ \sum_i \sum_j \mu_j \rho_j^U \frac{x_i^U}{\rho_i^U ||x_i||} P(j,i) 1_{||x_i||>0} \\
&+ \sum_i \sum_j \mu_j \rho_j^U \frac{x_i^I}{\rho_i^I ||x_i||} E(j,i) 1_{||x_i||>0} \\
&+ \sum_i(\lambda_i^S + \sum_j \mu_j \rho_j^U C(j,i)) R_i
\end{aligned}$$

which can be decomposed into three parts:

$$\sum_i(\lambda_i^U + \lambda_i^I + \lambda_i^S) = \sum_i \rho_i^U \mu_i d_i + \sum_i(\lambda_i^S + \sum_j \mu_j \rho_j^U C(j,i)) R_i$$

$$\sum_i(\mu_i + (\lambda_i^S + \sum_j \mu_j \rho_j^U C(j,i)) R_i) \frac{x_i^U}{||x_i||} 1_{||x_i||>0} = \sum_i(\lambda_i^U + \sum_j \mu_j \rho_j^U P(j,i)) \frac{x_i^U}{\rho_i^U ||x_i||} 1_{||x_i||>0}$$

and

$$\sum_i(\lambda_i^S + \sum_j \mu_j \rho_j^U C(j,i)) R_i \frac{x_i^I}{||x_i||} 1_{||x_i||>0} = \sum_i(\lambda_i^I + \sum_j \mu_j \rho_j^U E(j,i)) \frac{x_i^I}{\rho_i^I ||x_i||} 1_{||x_i||>0}$$

And the last two equations hold because of the flow equations (i.e. Equations 9.1 and 9.2). It remains to prove that the first equation is consistent with Equations

9.1 and 9.2 and describes the flow between the network and the outside. From these equations, we obtain:

$$\rho_i^{\mathrm{U}}\mu_i + \rho_i^{\mathrm{U}}\mathrm{R}_i(\lambda_i^{\mathrm{S}} + \sum_j \mu_j\rho_j^{\mathrm{U}}\mathrm{C}(j,i)) = \lambda_i^{\mathrm{U}} + \sum_j \mu_j\rho_j^{\mathrm{U}}\mathrm{P}(j,i)$$

and

$$(\lambda_i^{\mathrm{S}} + \sum_j \mu_j\rho_j^{\mathrm{U}}\mathrm{C}(j,i))\mathrm{R}_i\rho_i^{\mathrm{I}} = \lambda_i^{\mathrm{I}} + \sum_j \mu_j\rho_j^{\mathrm{U}}\mathrm{E}(j,i)$$

Thus, adding the two equalities:

$$\rho_i^{\mathrm{U}}\mu_i + (\rho_i^{\mathrm{U}} + \rho_i^{\mathrm{I}})\mathrm{R}_i(\lambda_i^{\mathrm{S}} + \sum_j \mu_j\rho_j^{\mathrm{U}}\mathrm{C}(j,i)) = \lambda_i^{\mathrm{U}} + \lambda_i^{\mathrm{I}} + \sum_j \mu_j\rho_j^{\mathrm{U}}(\mathrm{P}(j,i) + \mathrm{E}(j,i))$$

But $\mathrm{R}_i(\rho_i^{\mathrm{U}} + \rho_i^{\mathrm{I}}) = \mathrm{R}_i - 1$. After substitution and summation on i, we get:

$$\sum_i \rho_i^{\mathrm{U}}\mu_i + \sum_i(\lambda_i^{\mathrm{S}} + \sum_j \mu_j\rho_j^{\mathrm{U}}\mathrm{C}(j,i))(\mathrm{R}_i - 1) = \sum_i(\lambda_i^{\mathrm{U}} + \lambda_i^{\mathrm{I}}) + \sum_j \mu_j\rho_j^{\mathrm{U}}\sum_i(\mathrm{P}(j,i) + \mathrm{E}(j,i))$$

Moving the negative terms on the r.h.s., we get:

$$\sum_i \rho_i^{\mathrm{U}}\mu_i + \sum_i(\lambda_i^{\mathrm{S}} + \sum_j \mu_j\rho_j^{\mathrm{U}}\mathrm{C}(j,i))\mathrm{R}_i = \sum_i(\lambda_i^{\mathrm{U}} + \lambda_i^{\mathrm{I}} + \lambda_i^{\mathrm{S}}) + \sum_j \mu_j\rho_j^{\mathrm{U}}\sum_i(\mathrm{P}(j,i) + \mathrm{E}(j,i) + \mathrm{C}(j,i))$$

Remember that due to the normalization, we have for all i

$$d_i + \sum_j \mathrm{P}(i,j) + \sum_j \mathrm{E}(i,j) + \sum_j \mathrm{C}(i,j) = 1.$$

Therefore, after cancellation of terms, we get:

$$\sum_i(\lambda_i^{\mathrm{U}} + \lambda_i^{\mathrm{I}} + \lambda_i^{\mathrm{S}}) = \sum_i \rho_i^{\mathrm{U}}\mu_i d_i + \sum_i(\lambda_i^{\mathrm{S}} + \sum_j \mu_j\rho_j^{\mathrm{U}}\mathrm{C}(j,i))\mathrm{R}_i$$

This concludes the proof.

Part V

Conclusion and perspectives

A. Synthesis

We have considered in this document the problem of the trade-off between power consumption and performance in IT networks. Motivated by the "Green by IT" concept, we have proposed analytical and numerical analysis for several subjects.

- For numerical studies we used the XBorne tool (Chapter 2). The numerical analysis of Markov chains always deals with a trade-off between complexity and accuracy. After many years of development of exact or bounding algorithms for stochastic matrices, we have gathered the most efficient into XBorne. Typically using XBorne, one can easily build models with tens of millions of states. Note that solving any questions with this size of models is a challenging issue.

- We first proposed, in Chapter 3, a numerical study for a multi-core DVFS processor. The processor adapts its speed (i.e. its frequency) to the workload. We used birth-death processes to generate closed forms for steady-state distribution, performance and energy measures (power and energy per job). We also used stochastic order to compare the performance of different processor configurations (one Pstate, two Pstates and all Pstates). For the energy measures in one Pstate configuration, we proposed sufficient conditions for the systems comparison. In a two Pstates configuration, the stochastic order for the energy measures does not fit. There is no monotony when changing the system's thresholds (monotony remains valid for performance measures with assumptions on the considered Pstates). Therefore we have proposed an algorithm that optimizes a cost function derived from both measures. The comparison of energy measures in all Pstates configuration is more complex and we intend to use an MDP algorithm in future works.

- In the following work (Chapter 4), we treated the problematic of performance and energy in networking level. With the deployment of new services related to the Internet of Things (IoT), a large quantities of data will be generated and processed in real time. Optical networks will represent the most efficient solution for high speed transmission in communication networks. NGREEN solution is based on a ring topology and new advances on optical technology which allows to design OSS. Thus the routing problem does not exist anymore and we only have to deal with the access to the ring, by the filling of the optical container and the insertion mode (opportunistic or slot reservation). In this work, we developed mathematical models in order to analyze aggregation efficiency and end to end delays taking into account some constraints. The goal is to propose a trade-off between energy efficiency and delays. We proposed a discrete-time Markov chain to model the packet aggregation in the optical container. This model is efficiently solved by a numerical algorithm based on the structure of the chain. This new numerical technique we developed and proved combines two previously proposed algorithms. Note that the KMS-BGS algorithm would have been much faster if we have been able to decompose the problem into many smaller problems. Nevertheless, the complexity analysis and the numerical results showed that our approach is the best one (among the ones we check) for the problem we consider and more generally

for DTMC with a block structure such that the diagonal blocks exhibit some structural properties for the directed cycles. Note that this property is not numerically based, it only takes into account the graph. From the resolution of this Markov chain, we compute the inter-arrival PDUs which are used for insertion delay analysis in both opportunistic and reservation slot modes. Another important contribution of this study is to propose efficient mathematical methods to decide which values of parameters (thresholds, parameters) and insertion modes to choose in order to guarantee both energy efficiency and delays.

- In Chapter 6, we have modelled a data center with a multi-server Jackson network in order to represent the resources (physical servers, VMs), and task activities (services and migrations) between the servers. We studied task migrations between overloaded servers to unloaded servers in order to see the effect on power consumption. Using a closed form solution for the steady-state probability, we derived analytic formulas for the power consumption. So we can compute bounds on migration power in order to reduce overall power consumption, also we gave the migration rate minimizing power consumption in the case of two servers. We have also extended the work to larger scale data centers by proposing and comparing two heuristics that significantly reduce power consumption.

- In Chapter 8, we analyzed mathematically an Energy Packet Network (EPN) using Markov chains and balance equations. We proved the product form solution for the stationary probability from which we compute the end to end delays and energy packet loss rates. We also proved the existence of a solution for the fixed point problem for any open and connected EPN's topology. In order to optimize end to end delays , we proposed and compare an heuristic and a Gradient descent algorithm for different sensor network architectures. The algorithms we proposed are designed to optimize the assignment of Δ solar panels over the sensor network.

- In the last chapter, we presented a new types of G-networks with a new type of customers. Typically, we represented an age-dependent server, with a service capacity which decreases with time until a rejuvenation takes place. We hope that such a theoretical result will help to develop new models for G-networks in the performability domain. It is possible to extend this result for a more general partial rejuvenation with a more complex state dependent distribution of destruction.

B. Perspectives

In this section, we propose further perspectives for some of our works. In Chapter 3 we proposed a comparison study for different configurations of a DVFS processor with one Pstate. We derived an optimization algorithm with the aim of choosing the best configuration for a two Pstates model. We can also extend the optimization study to a model with several Pstates (> 2), by proposing an heuristic that gives the

best distribution of thresholds that optimize the cost function according to the input parameters. Another approach is to integrate the reinforcement learning [Wan+18]. This provides a representation of the states in an environment by taking actions and receiving rewards. One of the features of reinforcement learning is learning the optimal strategy by exploring unknown environments. From this perspective, three components are required with careful treatment, namely agents, rewards, and policies. We can derive a Markovian decision process (MDP) algorithm that optimizes the cost function and generate the thresholds directly from traces. The advantage is that this approach will be based on real input parameters, therefore the system "learns" an optimal policy. However, convergence time may be rather significant.

In Chapter 4 the mathematical model proposed for the filling of an optical container can be used for a variety of other applications. Any system that receives data and subject to a global clock can be considered. For instance the loading of sensor's batteries with energy packets in a sensor network.

In Chapter 6 we manage to derive an MDP algorithm for the load balancing problem between physical servers in order to minimize power consumption/performance. The state environment would be the number of tasks in each physical server, and the actions involved are "migrate" or "do not migrate" from one physical server to another. We can also extend the problem by considering that logical servers of physical servers support the DVFS mechanism which will further complicate the task migration decision.

We also seek to obtain more accurate and realistic energy models as in [Out+15] where authors compare some well known cloud simulation tools as CloudSim, GreenCloud, SimGrid, and iCanCloud. We can use theses simulators to get realistic power and energy consumption for our numerical results. In particular, the power consumption related to servers (logical and physical) in several states "IDLE", "AWAKE", "SWITCHING ON", "SWITCHING OFF", "HIBERNATE" ... , also the power consumption when migrating a single task from a physical server to another. These records are often difficult to acquire from other sources than simulation.

The development of the fifth-generation 5G cellular network technologies creates the possibility to deploy enormous sensors in the framework of the IoT and to process massive data. In Chapter 8 we proposed an analytical study of an energy packet network (EPN), this model is later used to the analysis of the assignment of solar panels over a sensor network where each sensor carries a rechargeable battery. We have supposed that the inter-arrival rate of energy packet that are generated by solar panels is Poissonian. However, we can extend this work to a more general and realistic distribution as the amount of energy generated by solar panels depends on the time of day and the weather. For instance, in [AJ20] authors propose a new model for stochastic sources of solar energy. This can be the next step of our EPN analysis study.

Bibliography

[Mun72] R.R. Muntz. "Poisson departure processes and queueing networks". In: *Technical Report. IBM Research Report RC4145* (1972).

[Bar76] A. D. Barbour. "Networks of Queues and the Method of Stages". In: *Advances in Applied Probability* 8.3 (1976), pp. 584–591.

[GZ81] C. D. Garcia and W. I. Zangwill. *Pathways to solutions, fixed points, and equilibria*. Prentice-Hall, Englewood Cliffs, N.J., 1981.

[Sto83] D. Stoyan. *Comparaison Methods for Queues and Other Stochastic Models*. Berlin: John Wiley and Sons, 1983.

[AD85] Tantawi A. and Towsley D. "Optimal static load balancing in distributed computer systems". In: *ACM* 32 (1985), pp. 445–465.

[CS85] P.-J. Courtois and P. Semal. "On polyhedra of Perron-Frobenius eigenvectors". In: *Linear Algebra and Applications* 65 (1985), pp. 157–170.

[DEJ86] Eager D.L., Lazowska E.D., and Zahorjan J. "A comparison of receiver-initiated and sender-initiated adaptive load sharing". In: *Perform. Eval* 6.1 (1986), pp. 53–68.

[Kel87] F. P. Kelly. *Reversibility and Stochastic Networks.* J. Wiley, 1987.

[TEK88] R. S. Tucker, G. Eisenstein, and S. K. Korotky. "Optical time-division multiplexing for very high bit-rate transmission". In: *Journal of Lightwave Technology* 6.11 (1988), pp. 1737–1749.

[Gel89] E. Gelenbe. "Random neural networks with negative and positive signals and product form solution". In: *Neural Computation* 4.1 (1989), pp. 502–510.

[Rob89] T.G. Robertazzi. "Recursive solution of a class of non-product form protocol models". In: *INFOCOM*. 1989.

[RDJ90] Mirchandaney R., Towsley D., and Stankovic J.A. "Adaptive load sharing in heterogeneous distributed systems". In: *Parallel Distrib. Comput.* 9.4 (1990), pp. 331–346.

[Fou91] J-M. Fourneau. "Computing the steady-state distribution of networks with positive and negative customers". In: *13^th IMACS World Congress on Computation and Applied Mathematics, Dublin*. 1991.

[Gel91] E. Gelenbe. "Product-form queuing networks with negative and positive customers". In: *Journal of Applied Probability* 28 (1991), pp. 656–663.

[HTS91] O. Hashida, Y. Takahaski, and S. Shimogawa. “Switched Batch Bernoulli Process (SBBP) and the Discrete-Time SBBP/G/1 Queue with Application to Statistical Multiplexer Performance”. In: *IEEE JSAC* 9.3 (1991), pp. 394–401.

[MN91] Squillante M.S. and R.D Nelson. “Analysis of task migration in shared-memory multiprocessor scheduling”. In: *ACM SIGMETRICS Performance Evaluation Review* 19.1 (1991), pp. 143–155. DOI: 107972.107987.

[SM91] E. de Souza e Silvia and M.Gerla. “Queueing network models for load balancing in distributed systems”. In: *Journal of Parallel and Distributed Computing* (May 1991).

[GS92] E. Gelenbe and R. Schassberger. “Stability of G-Networks”. In: *Probability in the Engineering and Informational Sciences* 6 (1992), pp. 271–276.

[Jai92] R. Jain. *THE ART OF COMPUTER SYSTEMS PERFORMANCE ANALYSIS, Techniques for Experimental Design Measurement, Simulation and Modeling*. Wiley professional computing, 1992.

[Gel93a] E. Gelenbe. “G-networks with instantaneous customer movement”. In: *Journal of Applied Probability* 30.3 (1993), pp. 742–748.

[Gel93b] E. Gelenbe. “G-Networks with signals and batch removal”. In: *Probability in the Engineering and Informatonal Sciences* 7 (1993), pp. 335–342.

[Gel93c] E. Gelenbe. “G-networks with triggered customer movement”. In: *Journal of Applied Probability* 30 (1993), pp. 742–748.

[Gel93d] E. Gelenbe. “Learning in the recurrent random network”. In: *Neural Computation* 5 (1993), pp. 154–164.

[Gel94] E. Gelenbe. “G-networks: An unifying model for queuing networks and neural networks”. In: *Annals of Operations Research* 48.1–4 (1994), pp. 433–461.

[Hil94] J. Hillston. “A compositional approach to Performance Modeling.” PhD thesis. University of Edinburgh, 1994.

[Wil94] William J. Stewart. *Introduction to the numerical solution of Markov Chains*. 1994.

[FKQ95] J-M. Fourneau, L. Kloul, and F. Quessette. “Multiple Class G-Networks with Jumps back to Zero”. In: *MASCOTS ’95: Proceedings of the 3rd International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*. Washington, DC, USA: IEEE Computer Society, 1995, pp. 28–32.

[FV95] J-M. Fourneau and D. Verchere. “G-networks with triggered batch state-dependent movement”. In: *MASCOTS ’95: Proc. of the Third Int. Workshop on Modeling, Analysis and Simulation on Computer and Telecommunication Systems* (1995), pp. 33–37.

[JLF95] Fourneau J-M., Kloul L., and Quessette F. "G-networks with jumps back to zero". In: *In Proceedings of ACM/IEEE MASCOTS* (1995), pp. 28–32.

[She95] Theordore Sheskin. "Computing the fundamental matrix for a reducible Markov chain". In: *Mathematics magazine* 68.5 (1995), pp. 391–398.

[Ste95] W.J. Stewart. *Introduction to the numerical Solution of Markov Chains*. New Jersey: Princeton University Press, 1995.

[FGS96] J-M. Fourneau, E. Gelenbe, and R. Suros. "G-networks with multiple classes of positive and negative customers". In: *Theoretical Computer Science* 155 (1996), pp. 141–156.

[Duf97] M. Duflo. *Random iterative models*. Berlin, Germany: Springer Verlag., 1997.

[FPS98] P. Fernandes, B. Plateau, and W. J. Stewart. "Efficient Descriptor-Vector Multiplications in Stochastic Automata Networks". In: *J. Acm* 45.3 (1998), pp. 381–414.

[GL98] Erol Gelenbe and Ali Labed. "G-networks with multiple classes of signals and positive customers". In: *European Journal of Operations Research* 108 (1998), pp. 293–305.

[CMP99] X. Chao, M. Miyazawa, and M. Pinedo. *Queueing Networks Customers, Signals and Product Form solutions*. John Wiley and Sons, 1999.

[GF99] E. Gelenbe and J-M. Fourneau. "Random Neural Networks with Multiple Classes of Signals". In: *Neural Computation* 11.4 (1999), pp. 953–963.

[LG99] John C. S. Lui and L. Golubchik. "Stochastic Complement Analysis of Multi-Server Threshold Queues with Histeresis". In: *Perform. Eval.* 35.1-2 (1999), pp. 19–48.

[XMM99] Chao X., Miyazawa M., and Pinedo M. "Queueing Networks: Customers, Signals and Product Form Solutions". In: *Wiley* (1999).

[JLD00] Fourneau J-M., Kloul L., and Verchere D. "Multiple class G-networks with list-oriented deletions". In: *European Journal of Operational Research* 126.2 (2000), pp. 250–272.

[JLF00] Fourneau J-M., Kloul L., and Quessette F. "Multiple class G-networks with iterated deletions". In: *Performance Evaluation* 42.1 (2000), pp. 1–20.

[KA00] B. Krishna Kumar and D. Arivudainambi. "Transient solution of an M/M/1 queue with catastrophes". In: *Computers and Mathematics With Applications* 40.10-11 (2000), pp. 1233–1240.

[CY01] H. Chen and D. Yao. "Jackson Networks". In: *Fundamentals of Queueing Networks. Stochastic Modelling and Applied Probability* 46 (2001), pp. 15–35.

[DK01] Alexander N. Dudin and A. V. Karolik. “BMAP/SM/1 queue with Markovian input of disasters and non-instantaneous recovery”. In: *Perform. Eval.* 45.1 (2001), pp. 19–32.

[GLX01] Erol Gelenbe, Ricardo Lent, and Zhiguang Xu. “Design and performance of cognitive packet networks”. In: *Perform. Eval.* 46.2-3 (2001), pp. 155–176.

[GDF01] O. Gusak, T. Dayar, and J. Fourneau. “Stochastic Automata Networks and Near Complete Decomposability”. In: *SIAM Journal on Matrix Analysis and Applications* 23.2 (2001), pp. 581–599.

[EJ02] Gelenbe E. and Fourneau J-M. “G-networks with resets”. In: *Performance Evaluation* 49 (2002), pp. 179–191.

[FP02] J-M. Fourneau and N. Pekergin. “An Algorithmic Approach to Stochastic Bounds”. In: *Performance Evaluation of Complex Systems: Techniques and Tools, Performance 2002, Tutorial Lectures*. Vol. 2459. LNCS. Springer, 2002, pp. 64–88.

[GF02] E. Gelenbe and J-M. Fourneau. “G-Networks with resets”. In: *Perform. Eval.* 49.1-4 (2002), pp. 179–191.

[Fou+03] J-M. Fourneau et al. “An open tool to compute stochastic bounds on steady-state distributions and rewards”. In: *11th Int. Conf. on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, Orlando*. IEEE Computer Society, 2003.

[Har03] P.G. Harrison. “Turning back time in Markovian process algebra”. In: *Theoretical Computer Science* 290.3 (2003), pp. 1947–1986.

[Ben+04] A. Benoit et al. “On the benefits of using functional transitions and Kronecker algebra”. In: *Perform. Eval.* 58.4 (2004), pp. 367–390.

[FLQ04] J-M. Fourneau, M. Le Coz, and F. Quessette. “Algorithms for an irreducible and lumpable strong stochastic bound”. In: *Linear Algebra and Applications* 386 (2004), pp. 167–185.

[Har04] Peter G. Harrison. “Compositional reversed Markov processes, with applications to G-networks”. In: *Perform. Eval.* 57.3 (2004), pp. 379–408.

[MRV04] Samir Mohamed, Gerardo Rubino, and Martín Varela. “Performance evaluation of real-time speech through a packet network: a random neural networks-based approach”. In: *Perform. Eval.* 57.2 (2004), pp. 141–161.

[PG04] Harrison P.G. “Compositional reversed Markov processes, with applications to G-networks”. In: *Performance Evaluation* 57.3 (2004), pp. 379–408.

[W04] Whitt W. “A Diffusion Approximation for the G/GI/n/m Queue”. In: *Operations Research* 52.6 (2004), pp. 922–941. DOI: 10.1287/opre.1040.0136.

[BF05] A. Busic and J-M. Fourneau. “A Matrix Pattern Compliant Strong Stochastic Bound”. In: *2005 IEEE/IPSJ International Symposium on Applications and the Internet Workshops (SAINT Workshops), Italy*. IEEE Computer Society, 2005, pp. 260–263.

[Inc05] Advanced Micro Devices (AMD) Inc. *POWER AND COOLING IN THE DATA CENTER - Addressing today's and tomorrow's challenges with the AMD Opteron TM processor and AMD PowerNow TM technology with Optimized Power Management (OPM)*. Tech. rep. Mar. 2005, pp. 1–8.

[VF05] V. Venkatachalam and M. Franz. “Power reduction techniques for microprocessor systems”. In: *ACM Comput* 37.3 (Sept. 2005), pp. 195–237.

[DPY06] T. Dayar, N. Pekergin, and S. Younès. “Conditional steady-state bounds for a subset of states in Markov chains”. In: *Structured Markov Chain (SMCTools) workshop in VALUETOOLS*. ACM, 2006.

[FQ06] J-M. Fourneau and F. Quessette. “Computing the Steady-State Distribution of G-networks with Synchronized Partial Flushing”. In: *Computer and Information Sciences - ISCIS 2006, 21th International Symposium, Istanbul, Turkey, November 1-3, 2006, Proceedings*. Ed. by Albert Levi et al. Vol. 4263. Lecture Notes in Computer Science. Springer, 2006, pp. 887–896.

[JF06] Fourneau J-M and Quessette F. “Computing the steady-state distribution of G-networks with synchronized partial flushing”. In: *Proceedings of ISCIS, 21th International Symposium* (2006), pp. 887–896.

[JJ06] Leino J. and Virtamo J. “Insensitive load balancing in data networks”. In: *Computer Networks* 50.8 (2006), pp. 1059–1068. DOI: 10.1016/j.comnet.2005.09.009.

[Ram06] R. Ramaswami. “Optical networking technologies: what worked and what didn't”. In: *IEEE Communications Magazine* 44.9 (2006), pp. 132–139.

[BKB07] Norman Bobroff, Andrzej Kochut, and Kirk Beaty. “Dynamic placement of virtual machines for managing SLA violations”. In: *International symposium on integrated network management* (2007), pp. 119–128.

[DeC07] DeCandia G, Hastorun D, Jampani M, kakulapati G, Lakshman A, Pilchin A, Sivasubramanian S, Vosshall P, Vogels W. “Amazon's highly available key-value store”. In: *Proceedings of twenty-first ACM SIGOPS Symposium on Operating Systems Principles* (2007), 205–220.

[Fou07] J-M. Fourneau. “Closed G-networks with resets: product form solution”. In: *In Proceedings of QEST* (2007), pp. 287–296.

[VAN08] Akshat Verma, Puneet Ahuja, and Anindya Neogi. “pmapper: power and migration cost aware application placement in virtualized systems”. In: *Springer Middleware 2008* (2008), pp. 243–264.

[B+09] Yang B. et al. “Performance Evaluation of Cloud Service Considering Fault Recovery”. In: *First International Conference, Cloud Computing* 5931 (2009), pp. 571–576. DOI: 10.1007/s11227-011-0551-2.

[SZC09] IRENEUSZ SZCZESNIAK. “Overview of optical packet switching”. In: *heoretical and Applied Informatics* 21.3-4 (2009), pp. 167–180.

[T+09] Czachórski T. et al. “Diffusion Approximation Model of Multi-server Stations with Losses”. In: *Electr. Notes Theor. Comput. Sci* 232 (2009), pp. 125–143. DOI: 10.1016/j.entcs.2009.02.054.

[BHM10] Simonetta Balsamo, Peter G. Harrison, and Andrea Marin. “A unifying approach to product-forms in networks with finite capacity constraints”. In: *SIGMETRICS 2010, Proceedings of the 2010 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, New York*. Ed. by Vishal Misra, Paul Barford, and Mark S. Squillante. ACM, 2010, pp. 25–36.

[Bus+10] A. Busic et al. “PSI2: Envelope Perfect Sampling of Non Monotone Systems”. In: *QEST 2010, Seventh Int. Conf. on the Quantitative Evaluation of Systems, Virginia, USA*. IEEE Computer Society, 2010, pp. 83–84.

[BBA10] R. Buyya, A. Beloglazov, and J. H. Abawajy. “Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges”. In: *CoRR* abs/1006.0308 (2010).

[GM10] Erol Gelenbe and Isi Mitrani. *Analysis and Synthesis of Computer Systems*. Imperial College Press, 2010.

[Kri10] Krioukov A, Mohan P,Alspaugh S, Keys L, Culler D, Katz R. “NapSAC: design and Implementation of a power-proportional web cluster”. In: *Green Networking: Proceendings of the first ACM SIGCOMM Workshop on Green Networking* (2010), 15–22.

[LA10] Lefevre L. and Orgerie A.C. “Designing and evaluating an energy efficient cloud”. In: *Journal of Supercomputing* 51.3 (2010), pp. 352–373.

[VF10] A. Valmari and G. Franceschinis. “Simple $O(m \log n)$ Time Markov Chain Lumping”. In: *16th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems, in Joint European Conferences on Theory and Practice of Software, ETAPS, Cyprus*. Ed. by J. Esparza and R. Majumdar. Vol. 6015. Lecture Notes in Computer Science. Springer, 2010, pp. 38–52.

[WTB10] Z. Wang, N. Tolia, and C. Bash. “Opportunities and challenges to unify workload, power, and cooling management in data centers”. In: *SIGOPS Oper. Syst. Rev.* 44.3 (2010), pp. 41–46.

[Bas+11] R. Basmadjian et al. “A methodology to predict the power consumption of servers in data centres”. In: *Proc. 2nd Int. Conf. e-Energy-Efficient Comput. Netw.* (2011), pp. 1–10.

[Bel+11] A. Beloglazov et al. “A taxonomy and survey of energy-efficient data centers and cloud computing systems”. In: *Adv. in Comput.* 82.11 (2011), pp. 47–111.

[BF11] A. Busic and J-M. Fourneau. “Iterative component-wise bounds for the steady-state distribution of a Markov chain”. In: *Numerical Linear Algebra with Applications* 18.6 (2011), pp. 1031–1049.

[Gel11] E. Gelenbe. “Energy Packet Networks: ICT based energy allocation and storage (invited paper)”. In: *GreenNets*. 2011, pp. 186–195.

[GNS11] V. Gupta, R. Nathuji, and K. Schwan. “An analysis of power reduction in datacenters using heterogeneous chip multiprocessors”. In: *SIGMETRICS Perform. Eval. Rev.* 39.3 (2011), pp. 87–91.

[11a] *https://www.techopedia.com/definition/25332/optical-burst-switching-obs*. 2011.

[11b] *https://www.techopedia.com/definition/27135/optical-circuit-switching-ocs*. 2011.

[Lim+11] S.-H. Lim et al. “A dynamic energy management scheme for multi-tier data centers”. In: *Proc. IEEE ISPASS* (2011), pp. 257–266.

[Mit11] I. Mitrani. “Service center trade-offs between customer impatience and power consumption”. In: *Perform. Eval.* 68.11 (2011), pp. 1222–1231.

[TNR11] Marco A.S.Netto Tiago C.Ferreto, Rodrigo N.Calheiros, and César A.F.De Rose. “Server consolidation with migration control for virtualized data centers”. In: *Future Generation Computer Systems* 27.8 (Oct. 2011), pp. 1027–1034.

[Wan+11] J. Wang et al. “A survey on energy efficient data management”. In: *SIGMOD Rec* 40.2 (Sept. 2011), pp. 17–23.

[AB12] Marin A. and Harrison P.G. Balsamo S. “Analysis of stochastic Petri nets with signals”. In: *Perform. Eval.* 69.11 (2012), pp. 551–572.

[BAB12] A. Beloglazov, J. Abawajy, and R. Buyya. “Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing”. In: *Future Gener. Comput. Syst.* 28.5 (2012), pp. 755–768.

[BDF12] A. Busic, H. Djafri, and J-M. Fourneau. “Bounded state space truncation and censored Markov chains”. In: *51st IEEE Conf. on Decision and Control (CDC 2012)*. 2012.

[Don12] Xiaowen Dong. “Green Optical Networks”. PhD thesis. The University of Leeds School of Electronic and Electrical Engineering, Nov. 2012.

[Gel12] Erol Gelenbe. “Energy packet networks: smart electricity storage to meet surges in demand”. In: *International ICST Conference on Simulation Tools and Techniques, SIMUTOOLS ’12, Sirmione-Desenzano, Italy, March 19-23, 2012*. ICST/ACM, 2012, pp. 1–7.

[LA12] Osama Masfary Lu Liu and Nick Antonopoulos. “Energy Performance Assessment of Virtualization Technologies Using Small Environmental Monitoring Sensors”. In: *Sensors* 12.5 (May 2012), pp. 6610–6628.

[RN12] S. Reda and A. N. Nowroz. “Power modeling and characterization of computing devices: A survey”. In: *Found. Trends Electron. Des. Autom.* 6.2 (Feb. 2012), pp. 121–216.

[Rei+12] Charles Reiss et al. “Towards understanding heterogeneous clouds at scale: Google trace analysis”. In: *ISTC-CC-TR-12-101* (Apr. 2012).

[SJD12] Jyothi Sekhar, Getzi Jeba, and S. Durga. “A SURVEY ON ENERGY EFFICIENT SERVER CONSOLIDATION THROUGH VM LIVE MIGRATION”. In: *International Journal of Advances in Engineering and Technology* 5.1 (Nov. 2012), pp. 515–525.

[Sou+12] P Beaulah Soundarabai et al. “COMPARATIVE STUDY ON LOAD BALANCING TECHNIQUES IN DISTRIBUTED SYSTEMS”. In: *International Journal of Information Technology and Knowledge Management* 6.1 (Dec. 2012), pp. 53–60.

[A13] Beloglazov A. “Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing”. PhD thesis. Department of Computing and Information Systems, the university of Melbourne, 2013.

[BKZ13] K. Bilal, S. Khan, and A. Zomaya. “Green data center networks: Challenges and opportunities”. In: *Proc. 11th Int. Conf. FIT.* (Dec. 2013), pp. 229–234.

[BMB13] T. Bostoen, S. Mullender, and Y. Berbers. “Power-reduction techniques for data-center storage systems”. In: *ACM Comput.* 45.3 (July 2013), 33:1–33:38.

[DFT13] Thu-Ha Dao-Thi, J-M. Fourneau, and Minh-Anh Tran. “Network of queues with inert customers and signals”. In: *7th International Conference on Performance Evaluation Methodologies and Tools, ValueTools ’13, Italy*. ICST/ACM, 2013, pp. 155–164.

[Gao+13] Y. Gao et al. “Quality of service aware power management for virtualized data centers”. In: *J. Syst. Architect.* 59.4/5 (2013), pp. 245–259.

[GSW13] C. Ge, Z. Sun, and N. Wang. “A survey of power-saving techniques on data centers and content delivery networks”. In: *IEEE Commun.* 15.3 (2013), pp. 1334–1354.

[KR13] Maurya Khushbu and Sinha Richa. “Energy conscious dynamic provisioning of virtual machines using adaptive migration thresholds in cloud data center”. In: *International Journal of Computer Science and Mobile Computing* 3.2 (2013), pp. 74–82.

[R+13] Ghosh R. et al. “Modeling and performance analysis of large scale IaaS Clouds”. In: *Future Generation Computer Systems* 29.5 (2013), pp. 1216–1234. DOI: 10.1016/j.future.2012.06.005.

[SA13] Balsamo S. and Marin A. “Separable solutions for Markov processes in random environments”. In: *European J. Oper. Res.* 229.2 (2013), pp. 391–403.

[SJH13] Haddad S., Mairesse J., and Nguyen H.-T. “Synthesis and analysis of product-form petri nets”. In: *Fund. Inform.* 1-2.122 (2013), pp. 147–172.

[Zha+13] Xiao Zhang et al. “A high-level energy consumption model for heterogeneous data centers”. In: *Simulation Modelling Practice and Theory* 39 (2013), pp. 41–55.

[Abd+14] Amany Abdelsamea et al. “Virtual Machine Consolidation Challenges: A Review”. In: *International Journal of Innovation and Applied Studies ISSN* 8.4 (Oct. 2014), pp. 1504–1516.

[al14] K. Bilal et al. “A taxonomy and survey on green data center networks”. In: *Future Gener. Comput. Syst.* 36 (July 2014), pp. 189–208.

[BFB14] A. Busic, J-M. Fourneau, and M. Ben Mamoun. “Stochastic Bounds with a Low Rank Decomposition”. In: *Stochastic Models, Special Issue with selected papers from the Eighth Int. Conf. on Matrix-Analytic Methods in Stochastic Models* 30.4 (2014), pp. 494–520.

[Cou14] Natural Resources Defense Council. “America’s data centers are wasting huge amounts of energy: critical action needed to save billions of dollars and kilowatts”. In: *NRDC Issue Brief, IB:14-08-A* (2014), pp. 1–6.

[DMM14] C. Dellacherie, S. Martinez, and J. san Martin. *Inverse M-matrices and Ultrametric Matrices*. Springer International Publishing Switzerland, 2014.

[EJF14] K. Ebrahimi, G. F. Jones, and A. S. Fleischer. “A review of data center cooling technology and operating conditions and the corresponding low grade waste heat recovery opportunities”. In: *Renew. Sustain. Energy Rev.* 31 (Mar. 2014), pp. 622–638.

[G14] Le Louët G. “Maîtrise énergétique des centres de données virtualisés: D’un scénario de charge à l’optimisation du placement des calculs”. PhD thesis. École nationale supérieure des mines de Nantes, 2014.

[Gel14a] E. Gelenbe. “A sensor node with energy harvesting”. In: *SIGMETRICS Performance Evaluation Review* 42.2 (2014), pp. 37–39.

[Gel14b] E. Gelenbe. “Adaptive Management of Energy Packets”. In: *IEEE 38th Annual Computer Software and Applications Conference, COMPSAC Workshops,* IEEE Computer Society, 2014, pp. 1–6.

[GHJ14] C. Gu, H. Huang, and X. Jia. “Power metering for virtual machine in cloud computing-challenges and opportunities”. In: *IEEE Access* 2 (Sept. 2014), pp. 1106–1116.

[HM14a] A. Hammadi and L. Mhamdi. “A survey on architectures and energy efficiency in data center networks”. In: *Comput. Commun.* 40 (Mar. 2014), pp. 1–21.

[HM14b] P.G. Harrison and A. Marin. “Product-forms in multi-way synchronizations”. In: *Comp. J.* 57.11 (2014), pp. 1693–1710.

[KL14] F. Kong and X. Liu. “A survey on green-energy-aware power management for datacenters”. In: *ACM Comput.* 47.2 (Nov. 2014), 30:1–30:38.

[Mit14] S. Mittal. “Power management techniques for data centers: A survey”. In: *CoRR*. abs/1404.6681 (2014).

[OAL14] A.-C. Orgerie, M. D. de Assuncao, and L. Lefevre. “A survey on techniques for improving the energy efficiency of large-scale distributed systems”. In: *ACM Comput.* 46.4 (Mar. 2014), 47:1–47:31.

[RLK14] A. Rahman, X. Liu, and F. Kong. “A survey on geographic load balancing based data center power management in the smart grid environment”. In: *IEEE Commun.* 16.1 (2014), pp. 214–233.

[SMi14] S.Mittal. “A survey of techniques for improving energy efficiency in embedded computing systems”. In: *arXiv preprint arXiv:1401.0765* (2014).

[SV14] S.Mittal and J. S. Vetter. “A survey of methods for analyzing and improving GPU energy efficiency”. In: *ACM Comput.* 47.2 (Aug. 2014), 19:1–19:23.

[A+15] Benoit A. et al. “Opportunistic Scheduling in Clouds Partially Powered by Green energy”. In: *IEEE International Conference on Green Computing and Communications (GreenCom)* (2015). DOI: 10.1109/DSDIS.2015.80.

[EE15] Gelenbe E. and Ceran E.T. “Central or distributed energy storage for processors with energy harvesting”. In: *Sustainable Internet and ICT for Sustainability, SustainIT* (2015), pp. 1–3. DOI: 10.1109/SustainIT.2015.7101380.

[GC15] E. Gelenbe and E. Tugce Ceran. “Central or distributed energy storage for processors with energy harvesting”. In: *2015 Sustainable Internet and ICT for Sustainability, SustainIT*. IEEE, 2015, pp. 1–3.

[GM15] E. Gelenbe and A. Marin. “Interconnected Wireless Sensors with Energy Harvesting”. In: *Analytical and Stochastic Modelling Techniques and Applications - 22nd International Conference, ASMTA*. Ed. by M. Gribaudo, D. Manini, and A. Remke. Vol. 9081. Lecture Notes in Computer Science. Springer, 2015, pp. 87–99.

[GK15] A Gupta and R Khumar Jha. “A survey of 5G network: architecture and emerging technologies”. In: *IEEE Access* (2015), pp. 1206–1232.

[MD15] V.J. Maccio and D.G. Down. “On optimal policies for energy-aware servers”. In: *Performance Evaluation* 90 (Apr. 2015), pp. 36–52.

[Mas+15] Toni Mastelic et al. “Cloud computing: survey on energy efficiency”. In: *ACM Computing Surveys, Association for Computing Machinery* 47.2 (2015), pp. 1–36.

[Out+15] E. Outin et al. “Enhancing Cloud Energy Models for Optimizing Datacenters Efficiency”. In: *2015 International Conference on Cloud and Autonomic Computing*. 2015, pp. 93–100. DOI: 10.1109/ICCAC.2015.10.

[RLK15] Aghajani R., Xingjie L., and Ramanan K. “Mean-field Dynamics of Load-Balancing Networks with General Service Distributions”. In: (2015).

[Rei+15] Charles Reiss et al. “Energy Efficiency Techniques in Cloud Computing: A Survey and Taxonomy”. In: *ACM Computing Surveys* (Oct. 2015).

[Sho+15] S.Y. Shorgin et al. “Threshold-based queuing system for performance analysis of cloud computing system with dynamic scaling”. In: *AIP Conference Proceedings* (2015).

[Van+15] Vangelista et al. “Long-Range IoT Technologies: The Dawn of LoRa™”. In: Sept. 2015, pp. 51–58. ISBN: 978-3-319-27071-5. DOI: 10.1007/978-3-319-27072-2_7.

[Ver+15] Abhishek Verma et al. “Large-scale cluster management at Google with Borg”. In: *Proceedings of the tenth European conference on computer systems - EuroSys* 15 (2015), pp. 1–17.

[al16] J. Shuja et al. “Survey of techniques and architectures for designing energy-efficient data centers”. In: *IEEE Syst. Journal* 10.2 (June 2016), pp. 507–519.

[Ase+16] A Asensio et al. “Study of the Centralization Level of Optical Network-Supported Cloud RAN”. In: *20th International Conference on Optical Network Design and Modeling (ONDM 2016), Cartagena, Spain*. IEEE. 2016, pp. 1–6.

[DWF16] Miyuru Dayarathna, Yonggang Wen, and Rui Fan. “Data Center Energy Consumption Modeling: A Survey”. In: *IEEE COMMUNICATIONS SURVEYS and TUTORIALS* 18.1 (2016), pp. 74–82.

[EE16] Gelenbe E. and Ceran E.T. “Energy Packet Networks With Energy Harvesting”. In: *IEEE Access* 4 (2016). DOI: 10.1109/ACCESS.2016.2545340, pp. 1321–1331.

[EC16] IEEE) EROL GELENBE (Fellow and ELIF TUGÇE CERAN. “Energy Packet Networks With Energy Harvesting”. In: *IEE Access, SPECIAL SECTION ON SMART GRIDS: A HUB OF INTERDISCIPLINARY RESEARCH* (Mar. 2016).

[FMB16] J-M. Fourneau, A. Marin, and S. Balsamo. “Modeling Energy Packets Networks in the Presence of Failures”. In: *24th IEEE International Symposium on Modeling and Analysis and Simulation of Computer and Telecommunication Systems and MASCOTS*. 2016, pp. 144–153.

[Fou+16] J-M. Fourneau et al. “XBorne 2016: A Brief Introduction”. In: *Computer and Information Sciences - 31st International Symposium, ISCIS 2016, Krakow, Poland*. Ed. by T. Czachórski et al. Vol. 659. Communications in Computer and Information Science. Springer, 2016.

[G+16] Huang G. et al. “Auto Scaling Virtual Machines for Web Applications with Queueing Theory”. In: *ICSAI The 3rd International Conference on Systems and Informatics* (2016).

[Gar+16] K. Gardner et al. “The power of d choices for redundancy”. In: *ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science* (2016), pp. 409–410.

[GC16] E. Gelenbe and E. T. Ceran. "Energy Packet Networks With Energy Harvesting". In: *IEEE Access* 4 (2016), pp. 1321–1331.

[HE16] Omer H.A. and Gelenbe E. "A Diffusion Model for Energy Harvesting Sensor Nodes". In: *24th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS* (2016), pp. 154–158.

[MAA16] Kurpiez Ma., Orgerie A.C., and Sobe A. "How Much Does a VM Cost? Energy-Proportional Accounting in VM-Based Environments". In: *PDP Euromicro International Conference on Parallel, Distributed, and Network-Based Processing* (2016), p. 8. DOI: 10.1109/PDP.2016.70.

[MAR16] ANDREA MARIN. "PRODUCT-FORM IN G-NETWORKS". In: *Probability in the Engineering and Informational Sciences* 30 (2016), pp. 345–360.

[X+16] Chang X. et al. "Modeling Active Virtual Machines on IaaS Clouds Unsing an M/G/m/m+k Queue". In: *IEEE Transactions on services computing* 9.3 (2016), pp. 408–420. DOI: 10.1109/TSC.2014.2376563.

[A+17] Benoit A. et al. "Reducing the energy consumption of large scale computing systems through combined shutdown policies with multiple constraints". In: *International Journal of high performance computing applications* 32.1 (2017). DOI: 10.1177/1094342017714530.

[ASJ17] Marin A., Balsamo S., and Fourneau J.M. "LB-networks: A model for dynamic load balancing in queueing networks". In: *Performance Evaluation* 115 (2017). DOI: 10.1016/j.peva.2017.06.004.

[AC+17] Orgerie A.C. et al. "Simulation Toolbox for Studying energy Consumption in Wired Networks". In: *CNSM International Conference on Network and Service Management* (2017). hal-01630226.

[Chi17a] D. Chiaroni. "ANR N-GREEN project: Eco-designed solutions for a greener ICT in high sensitive network segments". In: *Workshop Green Days@Sophia*. Sophia-Antipolis, 2017.

[Chi17b] D. Chiaroni. "Network Energy: Problematic and solutions towards sustainable ICT". In: *ICO 24*. Tokyo, Japan, 2017.

[CIG17] CIGREF. *"Du Green IT au Green by IT, Exemples d'applications dans les Grandes Entreprises"*. Tech. rep. 2017. URL: https://www.cigref.fr/wp/wp-content/uploads/2017/01/CIGREF-Du-Green-IT-au-Green-by-IT-2017.pdf.

[D C17] B. Uscumlic D. Chiaroni. "Potential of WDM packets". In: 2017. DOI: 10.23919/ONDM.2017.7958521.

[FG17] J-M. Fourneau and E. Gelenbe. "G-Networks with Adders". In: *Future Internet* 9.3 (2017), p. 34.

[Son+17] Yonghua Song et al. "An Internet of Energy Things Based on Wireless LPWAN". In: *Engineering* 3.4 (2017), pp. 460–466. ISSN: 2095-8099. DOI: https://doi.org/10.1016/J.ENG.2017.04.011.

[WC17] C. Ware and D. Chiaroni. “Towards WDM Slot switching for aggregation access and metropolitan applications: the ANR N-GREEN project”. In: *ICTON*. 2017.

[GA18] E. Gelenbe and O. H. Abdelrahman. “An Energy Packet Network model for mobile networks with energy harvesting”. In: *Nonlinear Theory and Its Applications and IEICE* 9.3 (2018), pp. 322–336.

[Gra+18] Annie Gravey et al. “Modelling packet insertion on a WSADM ring”. In: *2018 International Conference on Optical Network Design and Modeling, ONDM 2018, Dublin, Ireland, May 14-17, 2018*. Ed. by Marco Ruffini et al. IEEE, 2018, pp. 82–87.

[Wan+18] D. Wang et al. “From IoT to 5G I-IoT: The Next Generation IoT-Based Intelligent Algorithms and 5G Technologies”. In: *IEEE Communications Magazine* 56.10 (2018), pp. 114–120. DOI: 10.1109/MCOM.2018.1701310.

[Zak18] Muhammad Zakarya. “Energy, performance and cost efficient datacenters: A survey”. In: *Renewable and Sustainable Energy Reviews* 94 (Oct. 2018), pp. 363–385.

[DF19] J. Doncel and J-M. Fourneau. “Balancing Energy Consumption and Losses with Energy Packet Network Models”. In: *IEEE International Conference on Fog Computing and ICFC*. 2019, pp. 59–68.

[GZ19] E. Gelenbe and Y. Zhang. “Performance optimization with energy packets”. In: *IEEE Systems Journal* 13.4 (2019), pp. 3770–3780.

[Ray19] Partha Pratim Ray. “Energy Packet Networks: an Annotated Bibliography”. In: *SN Computer Science* 1.6 (2019). DOI: 10.1007/s42979-019-0008-x.

[AJ20] Sara Alouf and Alain Jean-Marie. “Short-Scale Stochastic Solar Energy Models: A Datacenter Use Case”. In: *MDPI Mathematics* 8.12 (2020). DOI: doi.org/10.3390/math8122127.

[Fou20] J-M. Fourneau. “Modeling Green Data-Centers and Jobs Balancing with Energy Packet Networks and Interrupted Poisson Energy Arrivals”. In: *SN Comput. Sci.* 1.1 (2020), 28:1–28:6.

[dec] decrypterlenergie.org. *La révolution numérique fera-t-elle exploser nos consommations d’énergie ?* Published on December 7, 2017 and modified on February 10, 2020.

[Tou+] T. Tournaire et al. “Génération de seuils optimaux dans une file hystérésis : application à un modèle de cloud”. In: (). URL: https://hal.archives-ouvertes.fr/hal-02126493/document.

Résumé substantiel

Contexte et problématiques

Le secteur de l'informatique et des télécommunications (IT) contribue très fortement à l'accroissement de la consommation globale d'énergie. Par conséquent, les émissions de CO_2 augmentent. Nombre de méthodes de réduction de la consommation, dans d'autres industries ou services, se traduisent par une augmentation de la consommation dans le secteur IT (approche "Green by IT" [CIG17]). Le concept "Green by IT" vise à réduire les impacts économiques et écologiques des activités de production des entreprises (produits ou services), grâce aux technologies numériques. Divers thèmes peuvent être explorés: les réseaux intelligents (smart grids), la mobilité et les transports intelligents , la surveillance environnementale et urbaine (urban monitoring), la dématérialisation, le travail à distance et la vidéo-conférence, les bâtiments intelligents et les logiciels d'éco-conception.

Dans le domaine du traitement (processing) et de la mise en réseau, l'optimisation énergétique repose principalement sur une adaptation de l'architecture et des moyens mis en œuvre en fonction des flux de trafic à transporter ou à traiter, ainsi que sur la qualité de service promise (QoS). On cherche donc à adapter les ressources à la demande, ce qui se traduit par un dimensionnement dynamique approprié à la charge. Ce dernier est, par nature, différent du dimensionnement "dans le pire des cas" le plus couramment utilisé. Sur le plan technologique, cette démarche implique que les équipements de réseau disposent de modes "sommeil", "sommeil profond" ou "hibernation" (la terminologie varie selon les fournisseurs); tous ces modes sont associés au même concept: mettre l'équipement en mode "sommeil", afin de réduire sa consommation d'énergie. La décision de passer d'un mode à l'autre n'est pas anodine: par exemple, mettre un appareil en mode "sommeil" pendant une très courte période pourrait ne pas être efficace, en raison de la consommation de redémarrage ou de réactivation. Pour un compromis performance/énergie pertinent, il est important de faire appel à des formules de consommation d'énergie obtenues à partir de l'utilisation des ressources du réseau et de ses équipements.

Travaux conduits

Les méthodes utilisées dans cette thèse sont fondées sur la théorie des réseaux de files d'attente, l'analyse numérique et analytique de chaînes de Markov, et sur la théorie de la comparaison stochastique. Face aux problématiques traitées, nous déterminons d'abord le réseau de files d'attentes (ou processus markovien) le plus adéquat

pour modéliser l'interconnectivité des systèmes. Nous proposons ensuite une solution analytique de la distribution stationnaire (si elle existe), sous forme produit ou forme "close". En exploitant certaines propriétés des graphes markoviens, nous avons également proposé de nouveaux algorithmes numériques rapides, notamment en améliorant la complexité temporelle.

Ce document se divise en cinq parties. La première (resp. deuxième) est consacrée à l'introduction générale (resp. conclusion et perspectives). Chacune des autres parties s'ouvre sur un état des connaissances afférentes aux études qui y sont présentées. On trouvera ci-dessous un récapitulatif des différents travaux:

- En premier lieu (Chapitre 2), nous avons proposé la dernière version de l'outil XBorne. L'analyse numérique des chaînes de Markov traite systématiquement d'un compromis entre complexité et précision. Après de nombreuses années de développement d'algorithmes exacts ou de bornes pour les matrices stochastiques, nous avons rassemblé les plus efficaces dans XBorne. À l'aide de ce logiciel, on peut facilement construire des modèles contenant des dizaines de millions d'états. On notera que la résolution numérique de toute problématique, avec des modèles de cette taille, représente un vrai défi.

- Au Chapitre 3, nous avons proposé une étude numérique d'un processeur multicœur (DVFS). Ce dernier adapte sa vitesse (i.e. sa fréquence) à la charge de travail. Nous avons utilisé des processus de naissance et de mort pour générer des formes "closes" pour la distribution stationnaire, ainsi que des mesures de performance et d'énergie (puissance et énergie par tâche). Nous avons également employé l'ordre stochastique pour comparer les performances de différentes configurations de processeurs (un Pstate, deux Pstates, tous les Pstates). Un Pstate correspond à un palier de vitesse et de consommation. Les plus hauts Pstates génèrent des vitesses supérieures, mais consomment davantage. Pour les mesures d'énergie dans une configuration à un Pstate, nous avons proposé des conditions suffisantes sur les paramètres, afin de comparer les systèmes. Dans une configuration à deux Pstates, l'ordre stochastique pour les mesures énergétiques ne convient pas. On ne constate pas de monotonie lorsque l'on modifie les seuils du système; en revanche, la monotonie reste valable pour les mesures de performance avec des hypothèses sur les Pstates considérés. Nous avons donc suggéré un algorithme qui optimise une fonction de coût dérivée des deux mesures. Dans un modèle multi-Pstates (>2), la comparaison des mesures énergétiques est plus complexe: nous avons l'intention d'utiliser un algorithme MDP pour nos travaux futurs.

- Dans l'étude suivante (Chapitre 4), nous avons abordé la problématique de la performance et de l'énergie dans un réseau optique. Avec le déploiement de nouveaux services liés à l'internet des objets (IoT), de grandes quantités de données seront générées et traitées en temps réel. Les réseaux optiques représenteront la solution la plus efficace pour la transmission à haut débit dans les réseaux de communication. La solution NGREEN est basée sur une topologie en anneau et sur de nouvelles avancées en technologie optique permettant de concevoir des OSS (Optical Slots Switching). Ainsi, le problème

du routage n'existe plus: il suffit de régler l'accès à l'anneau par le remplissage du conteneur optique et le mode d'insertion (opportuniste ou réservation de créneau). Dans ce travail, nous avons développé des modèles mathématiques, afin d'analyser l'agrégation efficace et les délais de bout en bout, en tenant compte de certaines contraintes. L'objectif est de proposer un compromis entre l'efficacité énergétique et les délais. Nous avons proposé une chaîne de Markov à temps discret, afin de modéliser l'agrégation des paquets dans le conteneur optique. Ce modèle est résolu efficacement par un algorithme numérique basé sur la structure de la chaîne. Cette nouvelle technique numérique, que nous avons développée et prouvée, combine deux algorithmes existants. On notera que l'algorithme KMS-BGS eût été beaucoup plus rapide si nous avions été capable de décomposer le graphe du système en plusieurs sous-matrices de petite taille. Néanmoins, l'analyse de la complexité et les résultats numériques ont montré que notre approche est la meilleure (parmi celles que nous avons vérifiées) pour le problème que nous considérons et, plus généralement, pour les DTMC comportant une structure en blocs tels que les blocs diagonaux présentent certaines propriétés structurelles pour les cycles dirigés. Vous remarquerez que cette propriété n'est pas numérique et qu'elle ne prend en compte que le graphe de la chaîne. À partir de la résolution de cette chaîne de Markov, nous calculons la distribution des inter-arrivées des PDUs et l'utilisons ensuite pour l'analyse du délai d'insertion dans les créneaux opportunistes ou avec réservation. Une autre contribution importante de cette étude est de proposer des méthodes mathématiques pour décider des valeurs des paramètres (notamment les seuils) et des modes d'insertion à choisir, afin de garantir à la fois efficacité énergétique et délais.

- Au Chapitre 6, nous avons modélisé un centre de données avec un réseau de Jackson multi-serveurs, afin de représenter les ressources (serveurs physiques, VM) et les activités des tâches (services et migrations) entre les serveurs. Nous avons étudié les migrations de tâches entre des serveurs surchargés et des serveurs délestés, afin d'analyser l'effet sur la consommation d'énergie. En utilisant une solution à forme produit de la distribution stationnaire, nous avons obtenu des formules analytiques pour la consommation d'énergie. Nous sommes ainsi parvenu à calculer une borne supérieure de la puissance de migration, afin de réduire la consommation énergétique globale. Nous avons également déterminé le taux de migration optimal minimisant la consommation énergétique, dans le cas de deux serveurs. Nous avons, par ailleurs, étendu l'étude à des centres de données à plus grande échelle, en proposant et en comparant deux heuristiques qui réduisent considérablement la consommation d'énergie.

- Au Chapitre 8, nous avons analysé mathématiquement un réseau de paquets énergétiques (EPN) en utilisant les chaînes de Markov et les équations de balance. Nous avons proposé et prouvé la solution sous forme produit de la probabilité stationnaire, à partir de laquelle nous calculons le délai de bout en bout et les taux de perte de paquets d'énergie. Nous avons également démontré l'existence d'une solution pour le problème du point fixe comprenant

toute topologie ouverte et connectée de l'EPN. Afin d'optimiser les délais de bout en bout, nous avons proposé et comparé un algorithme heuristique et un algorithme de descente de gradient, pour différentes architectures de réseaux de capteurs. Les algorithmes établis sont conçus pour optimiser l'affectation d'un certain nombre de panneaux solaires dans un réseau de capteurs.

- Dans le dernier chapitre, nous avons présenté un nouveau modèle de G-réseaux avec un nouveau type de clients. Plus spécialement, nous avons représenté un serveur dont le taux d'utilisation dépend de l'âge. La capacité de service diminue avec le temps, jusqu'à ce qu'un rajeunissement ait lieu. Nous espérons que ce résultat théorique aidera à développer de nouveaux modèles de G-réseaux dans le domaine des performances. Avec une distribution plus complexe dépendant de l'état de destruction, il serait possible d'étendre ce résultat à un rajeunissement partiel plus général.

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Évaluation des performances pour des réseaux IT économes en énergie

Mots clés: Consommation énergétique et performances. Réseaux de files d'attente. Chaînes de Markov. Forme produit pour la distribution stationnaire. Centre de données.

Résumé: L'économie d'énergie dans les réseaux de télécommunication poursuit un objectif majeur: réduire la consommation globale. La part du domaine IT est déjà très élevée et tend à augmenter. En effet, de nombreuses techniques destinée à réduire la consommation dans d'autres industries ou services se traduisent par davantage de traitements informatiques et de télécommunications (l'approche "Green by IT"); en bref, par une augmentation de la consommation dans les domaines IT. Il est donc important, d'un point de vue économique, de parvenir à réduire la consommation énergétique par bit transmis ou calculé (l'approche "Green by IT", au coeur de nos travaux). Dans le domaine des réseaux, l'optimisation énergétique repose essentiellement sur une adaptation de l'architecture et des ressources employées, en fonction des flux à transporter et de la qualité de service promise. En conséquence, on cherche donc à adapter les ressources à la demande, ce qui se traduira par un dimensionnement dynamique et adapté à la charge. Ce procédé est, par nature, différent d'un dimensionnement "au pire cas" que l'on utilise généralement. Sur le plan technologique, il sera nécessaire que les équipements de réseaux disposent de modes "sommeil", "sommeil profond" ou "hibernation" (terminologie variable selon les fournisseurs) ; tous ces modes sont associés au même concept: mettre en sommeil l'équipement afin de réduire sa consommation d'énergie. Pour que le compromis performance/énergie soit pertinent, il parait important d'employer des formules de consommation énergétiques obtenues à partir de l'utilisation des ressources du réseau. Les méthodes que nous proposons relèvent de la théorie des réseaux de files d'attente, de l'analyse des chaînes de Markov (analytiquement, en proposant de nouvelles formes produit , numériquement, en suggérant de nouveaux algorithmes de résolution), et de la théorie de la comparaison stochastique. Au niveau des applications, nous avons abordé diverses problématiques: les mécanismes de DVFS avec un changement de vitesse des processeurs; la migration de tâche entre serveurs physiques dans un centre de données (équilibre de charge, consolidation); les réseaux optiques avec un remplissage efficace des conteneurs (optiques); la distribution d'énergie intermittente dans un réseau de capteurs (et réseau LORA). À cette fin, nous proposons un nouveau modèle des réseaux à paquets d'énergie (EPNs).

Maison du doctorat de l'Université Paris-Saclay
2ème étage aile ouest, Ecole normale supérieure Paris-Saclay
4 avenue des Sciences,
91190 Gif sur Yvette, France

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Title: Performance evaluation of green IT networks

Keywords: Energy/Power consumption. Performance evaluation. Queueing networks. Markov chains analysis. Product Form for equilibrium distribution. Data center.

Abstract: Energy saving in telecommunication networks is a major objective to reduce overall consumption. The IT sector already has a very high contribution to this increase. Indeed, many techniques to reduce consumption in other industries or services results in more IT and telecommunications (the "Green by IT" approach) and therefore in an increase of consumption in IT domains. It is therefore important from an economical point of view to reduce the energy consumption per transmitted or calculated bit (the "Green by IT" concept). In the networks domain, energy optimization is mainly based on an adaptation of the architecture and the resources employed according to the traffic flows to be transported and the promised quality of service. We therefore seek to adapt resources to demand, which results in a dynamic dimensioning adapted to the load. This is by nature different from the worst-case dimensioning commonly used. In terms of technology, this requires network equipment to have "sleep", "deep sleep" or "hibernate" modes (terminology varies among suppliers); all these modes are associated with the same concept: putting the equipment to sleep to reduce its energy consumption. For the performance/energy tradeoff to be relevant, it seems important to use energy consumption formulas obtained from the network resource utilization. The approaches we propose are based on the theory of queuing networks, Markov chain analysis (analytically by proposing new product forms and numerically by suggesting new resolution algorithms) and the theory of stochastic comparison. At the application level we have addressed various issues: DVFS mechanisms with a change of processors speed; task migration between physical servers in a data center (load balancing, consolidation); optical networks with efficient filling of optical containers; intermittent power distribution in a sensor network (and LoRa network), including a new model of Energy Packet Networks (EPNs).

Maison du doctorat de l'Université Paris-Saclay
2ème étage aile ouest, Ecole normale supérieure Paris-Saclay
4 avenue des Sciences,
91190 Gif sur Yvette, France