Resource allocation in a Cloud partially powered by renewable energy sources

Yunbo Li

UNIVERSITE BRETAGNE LOIRE

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Thèse de Doctorat

Yunbo LI

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'École nationale supérieure Mines-Télécom Atlantique Bretagne Pays de la Loire
sous le sceau de l'Université Bretagne Loire

École doctorale : Sciences et technologies de l'information, et mathématiques (STIM 503)

Discipline : Informatique, section CNU 27
Unité de recherche : Laboratoire des Sciences du Numérique de Nantes (LS2N) & Institut de Recherche en Informatique et Systèmes Aléatoires (IRISA)

Soutenue le 12 juin 2017
Thèse numéro : 2017IMTA0019

Resource allocation in a Cloud partially powered by renewable energy sources

JURY

Président :	**M. Romain ROUVOY**, Professeur, Université de Lille
Rapporteurs :	**M. Pascal BOUVRY**, Professeur, Université du Luxembourg
	M^me^ Patricia STOLF, Maître de conférences (HDR), Université Toulouse Jean-Jaurès
Examinateurs :	**M. Laurent LEFÈVRE**, Chargé de recherche (HDR), Inria
	M. Romain ROUVOY, Professeur, Université de Lille
Directeur de thèse :	**M. Jean-Marc MENAUD**, Professeur, IMT Atlantique
Co-directrice de thèse :	**M^me^ Anne-Cécile ORGERIE**, Chargée de recherche, CNRS

Remerciements

Je voudrais tout d'abord remercier tout particulièrement mes encadrants de thèse, Dr. Anne-Cécile Orgerie (CNRS) et Pr. Jean-Marc Menaud (IMT-Atlantique), qui m'ont dirigé tout au long de ces trois années de thèse. Ils ont toujours été disponibles, à l'écoute de mes nombreuses questions, et se sont toujours intéressés à l'avancée de mes travaux.

Je tiens à exprimer ma reconnaissance à Dr. Christine Morin, je suis ravi d'avoir travaillé dans son équipe Myriads. Je remercie également Pr. Jean-Louis Pazat qui m'a beaucoup apporté.

Dr. Patricia Stolf et Pr. Pascal Bouvry m'ont fait l'honneur d'être rapporteurs de ma thèse, pour le temps consacré à la lecture de cette thèse, et pour les suggestions et les remarques intéressantes qu'ils m'ont faites.

Je tiens à remercier Dr. Laurent Lefèvre et Pr. Romain Rouvoy pour avoir accepté de participer à mon jury de thèse.

Ayant effectué mon stage à l'Université Rutgers, je remercie Pr. Manish Parashar et Dr. Ivan Rodero avec qui j'ai eu la chance de pouvoir travailler. Leur rigueur, leurs capacités d'analyse des problèmes et leurs connaissances ont été très utiles pour me permettre de progresser ainsi quel leurs réponses à mes questions.

Je remercie également tous les thésards David Guyon, Ismael Cuadrado Cordero, Anna Giannakou, Bogdan F. Cornea, Amir Teshome Wonjiga, Anca Iordache, Genc Tato et Gilles Madi-Wamba qui m'ont entouré et m'ont conseillé.

Je remercie toutes les personnes dans le projet EPOC qui j'ai partagé mes travaux et notamment ces années de thèse : Nicolas Beldiceanu, Bárbara Dumas Feris, Philippe Gravey, Sabbir Hasan, Claude Jard, Thomas Ledoux, Didier Lime, Gilles Madi-Wamba, Pascal Morel, Michel Morvan, Marie-Laure Moulinard, Jean-Louis Pazat, Olivier H. Roux, Ammar Sharaiha.

Mes derniers remerciements vont à mes parents et ma copine Zifan, je n'aurais rien fait de tout cela sans votre amour.

Contents

List of Tables

List of Figures

1

Introduction

1.1 Context

With the rapidly increasing demand in Cloud computing services, the usage of data centers (DCs), increases dramatically [OAaL14]. Consequently, the global electricity part dedicated to DCs' consumption has reached unprecedented levels. In 2012, the number of data centers worldwide was estimated at 509,147 consuming roughly the output of 30 nuclear power plants [Gla12]. In 2016, another study estimates that worldwide the data centers use 91 billion kilowatt-hours of electricity – enough to power New York City twice over – and their consumption is still growing rapidly [Res16]. This ever-growing electricity consumption constitutes a major concern for DC operators.

Besides the ecological impact, the energy consumption is a predominant criteria for cloud providers since it determines the daily cost of their infrastructure. As a consequence, power management becomes one of the main challenges for DC infrastructures and more generally for large-scale distributed systems [OAaL14].

In parallel to the expansion of cloud computing, since several years, a new model emerges: decentralized cloud infrastructures [Ber+14]. To improve the performance of their cloud and to leverage their available infrastructure, telecommunication operators, like Orange, try to deploy micro data center (20 to 50 servers by micro-DC) at the network border, closer to customers. In this new model, by deploying data centers closer to the user, cloud operators aim at improving the response time and throughput of applications.

One way to save energy at a data center level consists in locating it close to where the electricity is generated, hence minimizing transmission losses. For example, Western North Carolina, USA, attracts data centers with its low electricity prices due to abundant capacity of coal and nuclear power following the departure of the region's textile and furniture manufacturing [Gre]. In 2011, this region had three super-size data centers from Google, Apple and Facebook with respective power demands of 60 to 100 MW, 100 MW and 40 MW [Gre] and these DCs are still in-use nowadays. However, such huge facilities represent only a small fraction of the global consumption of data centers. Indeed, small- and medium-sized server rooms continue to account for nearly half the electricity consumption of the market [WD14].

Other companies opt for greener sources of energy. For example, Quincy (Washington, USA) supplies electricity to data facilities from Yahoo, Microsoft, Dell and Amazon with its

low-cost hydro-electrics left behind following the shutting down of the region's aluminum industry [Gre]. Several renewable energy sources like wind power, solar energy, hydropower, bio-energy, geothermal power and marine power can be considered to power up super-sized facilities. The production variability of most renewable sources leads data center facilities to only partially rely on them and to depend also on the the regular electrical grid as a backup.

While using renewable sources bring new opportunities to reduce energy costs, reduce peak power costs, or both [Goi+13], they are mostly intermittent and fluctuating over time (sun, wind, etc.). These variations may lead to electricity losses if the computing workload does not match the renewable production. Cloud infrastructures, on the other hand, can take advantage of multiple locations to increase their green consumption with approaches such as follow-the-sun and follow-the-wind [Fig+09]. As sun and wind provide renewable sources of energy whose capacity fluctuates over time, the rationale is to place computing jobs on resources using renewable energy, and migrate jobs as renewable energy becomes available on resources in other locations.

From an energy point of view, these micro-data centers allow the study of new power supply solutions based on renewable energy, like wind or sun. Using these renewable energy sources can reduce the operating cost but, unfortunately, this kind of energy stays intermittent by nature.

1.2 Problem Statement and Research Challenges

For the last decade, there has been substantial improvements in data center efficiency. Much of the progress on efficiency has been made in the domain of facility and equipment. For instance, Google and Facebook developed their own ultra-efficient server farms with free cooling solutions. While the modern server has become more power-efficient, despite massive state-of-the-art work to improve the power management of possessors, servers are still far from pure power-proportional behavior. Indeed, an idle server can consume up to 50% of its maximal power [LWW07; FWB07; MGW09].

Little progress has been done in the field of server operation efficiency with regard to server utilization. The server utilization represents the ratio between the physical resources (e.g., CPU, RAM) consumed by processing load and the maximum server capacity [BH07]. Several studies show that the average server utilization – particularly, in term of average CPU utilization – remains static around 12 to 18 percent [WK12; Sny10]. These idle or underutilized servers are still consuming energy while most of the time doing little work. Virtualization has been rapidly and widely implemented in modern data centers. This technology enables a single physical machine to run multiple isolated operating systems. This also means that using fewer physical machines, one is able to handle the same quantity of computational tasks. Yet, even with virtualization, that is broadly deployed in data centers, the server's average utilization is typically below 40 percent [WD14].

Integrating renewable energy into DCs is expected to be an important factor in the design of next generation of DCs. It can offset a part of energy consumed from traditional supply (e.g. fossil fuel), thus reducing the carbon emissions. However, a major challenge for employing renewable energy sources for DCs, such as solar and wind, is their variable and intermittent nature. Unlike traditional energy sources that enable to provide a controllable and steady power, the renewable energy is difficult to meet the workload power requirements. Another possible method for improving the effective utilization of intermittent and fluctuating renewable consists in using energy storage devices (i.e. batteries) to store green production surplus, and to use it during low production periods [Goi+13]. Typically for

solar sources, energy can be stored during the day – if not fully consumed – and be utilized during nights when there is no production. However, batteries have an inherent energy efficiency (their yield) that leads to energy losses. A framework for managing Infrastructure-as-a-Service (IaaS) cloud resources of a single data center is needed to improve not only the energy-efficiency, but also to give means for optimizing the utilization of renewable energy.

Our objectives are the following:

— investigating renewable energy integration into DCs;
— providing resource management algorithms to increase energy-efficiency and to optimize renewable energy consumption for a single Cloud DC;
— providing a framework making use of energy storage devices and resource management algorithms for maximizing renewable energy consumption in a Cloud DC;
— validating this framework under realistic conditions.

1.3 Contributions

The main goal of this thesis consists in keeping a low fossil energy consumption level in the data center, thus reducing the CO_2 emissions. It starts by observing real traces of solar power production that verifies the intermittent and variable nature of renewable energy. Another data analysis on trace of real-world server utilization from a small data center demonstrates that the server average utilization stays in very low levels of use with regard to CPU utilization. This analysis also shows that the server average utilization trend is less variable than renewable energy that is intermittent by nature. Meanwhile, we find that part of computational tasks present slack periods of time that enables to shift the computations in time. According to these observations and analysis, we present our contributions in this thesis as follows:

1. We propose a novel framework: oPportunistic schedulIng broKer infrAstructure (PIKA) [LOM15] to save energy in small mono-site data centers. PIKA aims at reducing the brown energy consumption (ie. from non-renewable energy sources), and improves the usage of renewable energy without energy storage for mono-site data center. It exploits jobs with slack periods, and executes or suspends them depending on the renewable energy availability. By consolidating the virtual machines (VMs) on the physical servers, PIKA adjusts the number of powered-on servers in order for the overall energy consumption to match with the renewable energy supply.
2. Another approach for improving the effective utilization of intermittent and fluctuating renewable energy consists in using batteries to store green production surplus, and to use it during low production periods. Typically for solar sources, energy can be stored during the day – if not fully consumed – and be utilized during nights when there is no production. However, Energy Storage Devices (ESDs) have an inherent energy efficiency due to different battery technologies that leads to energy losses. In the second contribution [LOM17], we discuss both the opportunistic scheduling and ESDs-based approaches for maximizing the usage of renewable energy in small and medium data centers, and we propose a solution mixing both approaches.
3. Finally, inspired by previous works, we propose to leverage on-site renewable energy production in the different edge cloud nodes to make Internet of Things (IoT) greener [Li+17]. Our aim is to evaluate, on a concrete use-case, the benefits of edge computing regarding renewable energy consumption. We propose an analytic model for deciding whether to offload computation from the objects to the edge or to the core Cloud, depending on the renewable energy availability and the desired application

Quality of Service (QoS), in particular trading-off between performance (response time) and reliability (service accuracy).

This thesis has been done in the context of the EPOC project (Energy Proportional and Opportunistic Computing systems, Labex CominLabs, http://www.epoc.cominlabs.ueb.eu, 2013-2017).

1.3.1 Publications

Peer-reviewed journal papers:

— ***Towards energy-proportional Clouds partially powered by renewable energy***
Nicolas Beldiceanu, Bárbara Dumas Feris, Philippe Gravey, Sabbir Hasan, Claude Jard, Thomas Ledoux, Yunbo Li, Didier Lime, Gilles Madi-Wamba, Jean-Marc Menaud, Pascal Morel, Michel Morvan, Marie-Laure Moulinard, Anne-Cécile Orgerie, Jean-Louis Pazat, Olivier H. Roux, Ammar Sharaiha
Computing, Springer Verlag, volume 99, issue 1, pages 3-22, January 2017.

Peer-reviewed international conference articles:

— ***The EPOC project: Energy Proportional and Opportunistic Computing system***
Nicolas Beldiceanu, Bárbara Dumas Feris, Philippe Gravey, Sabbir Hasan, Claude Jard, Thomas Ledoux, Yunbo Li, Didier Lime, Gilles Madi-Wamba, Jean-Marc Menaud, Pascal Morel, Michel Morvan, Marie-Laure Moulinard, Anne-Cécile Orgerie, Jean-Louis Pazat, Olivier Roux, Ammar Sharaiha
International Conference on Smart Cities and Green ICT Systems (SMARTGREENS), pages 1-7, May 2015, Lisbon, Portugal.

— ***Opportunistic Scheduling in Clouds Partially Powered by Green Energy***
Yunbo Li, Anne-Cécile Orgerie, Jean-Marc Menaud
IEEE International Conference on Green Computing and Communications (GreenCom), pages 448-455, December 2015, Sydney, Australia.

— ***Balancing the use of batteries and opportunistic scheduling policies for maximizing renewable energy consumption in a Cloud data center***
Yunbo Li, Anne-Cécile Orgerie, Jean-Marc Menaud
Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP), pages 1-8, March 2017, St Petersburg, Russia.

— ***Leveraging Renewable Energy in Edge Clouds for Data Stream Analysis in IoT***
Yunbo Li, Anne-Cécile Orgerie, Ivan Rodero, Manish Parashar, Jean-Marc Menaud
IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), May 2017, Madrid, Spain.

— ***Towards energy-proportional Clouds partially powered by renewable energy***
Nicolas Beldiceanu, Bárbara Dumas Feris, Philippe Gravey, Sabbir Hasan, Claude Jard, Thomas Ledoux, Yunbo Li, Didier Lime, Gilles Madi-Wamba, Jean-Marc Menaud, Pascal Morel, Michel Morvan, Marie-Laure Moulinard, Anne-Cécile Orgerie, Jean-Louis Pazat, Olivier H. Roux, Ammar Sharaiha
Computing, Springer Verlag, volume 99, issue 1, pages 3-22, January 2017.

1.3.2 Dissertation organization

The rest of this manuscript is organized as follows.

Chapter 2 surveys recent green computing efforts to save energy at infrastructure level in data centers. We highlight the different mechanisms at server level to save brown energy consumption, particularly with regard to power proportionality and energy efficiency. We identify opportunities for further reducing data center energy consumption that relates

to renewable energy integration. We then survey the technical landscape to increase the renewable energy usage.

Chapter 3 presents the EpoCloud architecture from hardware to middleware layers: this data center architecture has been designed within the context of the EPOC project. This prototype aims at optimizing the energy consumption of mono-site Cloud data centers connected to the regular electrical grid and to renewable-energy sources. Later in this chapter, we describe a self-developed trace-driven simulator implementing this prototype and that is used for parts of the experimentation done in this thesis.

Chapter 4 presents the proposed framework PIKA that consists of energy-aware opportunistic scheduling algorithms based on the distinction of two kinds of tasks (web tasks and batch tasks) and that leverages renewable energy availability to perform opportunistic tasks without hampering performance.

Chapter 5 presents an ESD-based approach for increasing the renewable energy utilization. We distinguish the performance between opportunistic scheduling and ESD-only solution. A hybrid solution is proposed later in this chapter that intends to find a good trade-off between these two approaches.

Chapter 6 advocates for leveraging on-site renewable energy production in the different edge cloud nodes for greening IoT systems while offering improved QoS compared to core cloud solution. We propose an analytic model to decide whether to offload computation from the objects to the edge or to the core Cloud, depending on the renewable energy availability and the desired application QoS. This model is validated on our application use-case that deals with video stream analysis from vehicle cameras.

Chapter 7 concludes and presents research perspectives.

2

State of the Art

2.1 Introduction

Cloud computing represents currently the most emphasized paradigm for providing and managing the Information and Communications Technology (ICT) resources to the online user. The basic concept of Cloud computing consists in making the distribution of computing resources in a large number of distributed computers, rather than the local computer or remote server. Since its appearance, the demand for computing and storage resources in data centers has rapidly grown, leading to a consequent increase of their energy consumption. As an example, for 2010, Google used 900,000 servers and consumed 260 million watts of electricity [Koo11]. Electricity becomes a key issue for deploying data center equipment.

Since the servers are among the primary energy consumers of data centers [SBD07], many green proposals have addressed the problem of the server's energy-efficiency. Dynamic voltage and frequency scaling that exploits server performance knobs is an example of such proposals. Meanwhile, virtualization technology brings new opportunity for saving energy. It enables processing multiple requests on the same server, thus making it possible to run the workload on fewer servers by consolidation. In addition, using clean energy and integrating renewable energy into data centers can result in further brown energy saving and reduces dependency on traditional energy sources (e.g. fossil fuel).

The rest of this chapter is organized as follows. Section 2.2 provides an overview of the energy consumption of different components in cloud data centers. The research efforts of energy-efficient technologies are presented in Section 2.3.1 and energy proportionality are presented in Section 2.3.2. Section 2.3.3 presents virtualization technologies for energy-saving and formalizes corresponding optimization problems. Section 2.3.4 presents the opportunity of integrating renewable energy for further reducing energy consumption. Section 2.3.5 describes a novel cloud architecture that leverages renewable energy in edge cloud data centers. Lastly, we summarizes this chapter in Section 2.4.

2.2 Data center: server, storage and network energy use

Cloud resources are gathered in data centers whose size depends on the activity of the Cloud provider. A data center (DC) is a facility used to house tens to thousands of computers and their associated components. These servers are used to host applications avail-

able in the Internet, from simple web servers to multi-tier applications, but also some batch jobs [MSJ14]. Besides the ecological impact, the energy consumption is a predominant criteria for DC providers since it determines the daily cost of their infrastructure. As a consequence, power management becomes one of the main challenges for DC infrastructures and more generally for large-scale distributed systems [OAaL14]. In this section, we provide an overview of DC energy consumption worldwide.

2.2.1 Data center types and components

In Figure 2.1, each segment represents the estimated energy consumption of data center market based on the number of installed servers and infrastructure electricity consumption [WD14]. Small- and medium-sized data centers account for nearly half the energy consumption of the market; they are typically composed of less than 100 servers with a light cooling system. Enterprise/Corporate data centers occupies 27 percent and multi-tenant data centers for 19 percent of global data center electricity consumption; multi-tenant data centers provide services to individual enterprises on a lease basis. The customers place and manage their own equipment while the data center provider manages the infrastructure and cooling facility.

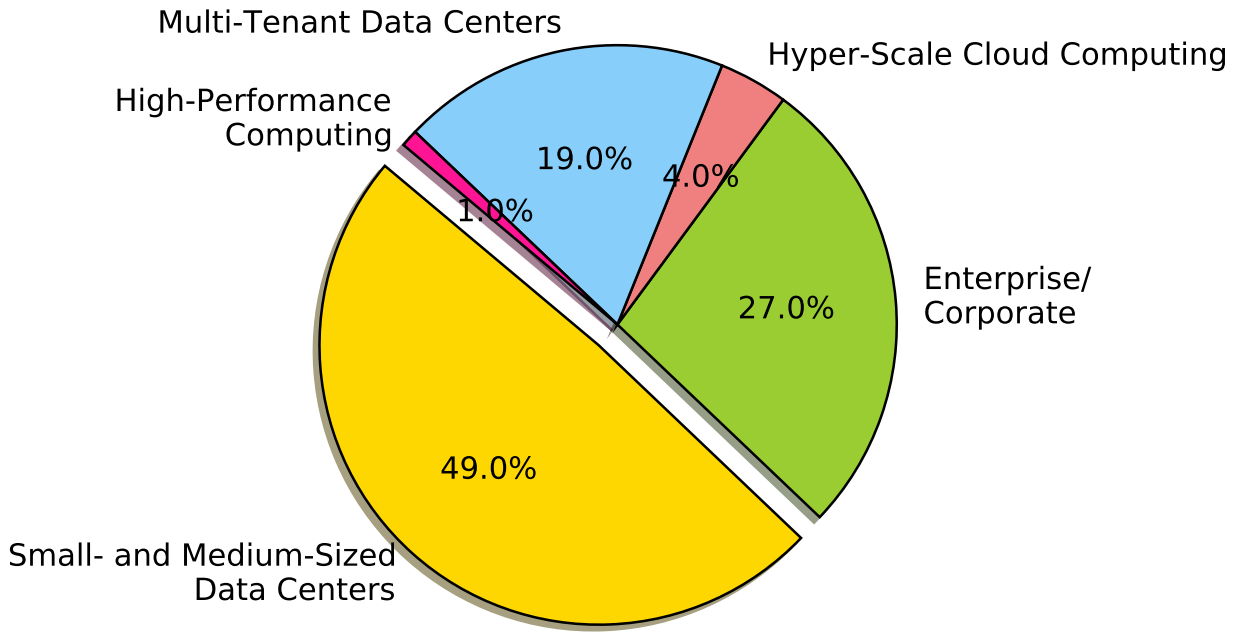


Figure 2.1 – Estimated U.S. data center electricity consumption by market segment (data from [WD14])

The energy consumption of hyper-scale cloud providers such as Google, Amazon and Facebook, only occupies 4 percent of the global data center energy consumption due to their aggressive deployment of energy efficiency mechanisms that lower their power bill. Finally, the smallest segment within data center market is high-performance computing (HPC); it is usually operated by universities and national research laboratories.

Inside the DC, the major energy consumer comprises IT (Information Technology) systems and cooling facilities. As shown in Figure 2.2, the servers can consume twice more energy than cooling systems. The server utilization represents processing data on the server relative to its maximum capacity. It directly affects the data center efficiency because the

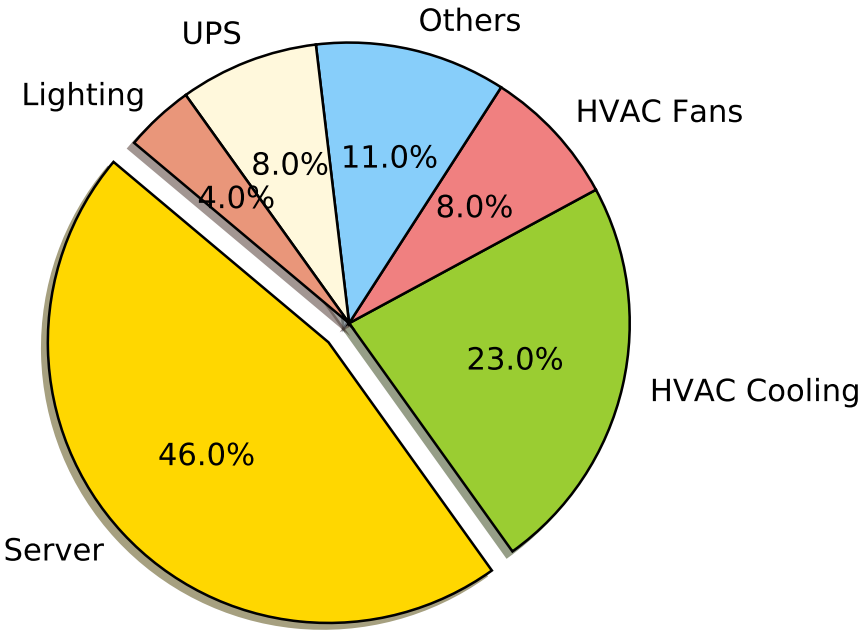


Figure 2.2 – Typical breakdown of the data center energy consumption (data from [SBD07])

server power efficiency drops dramatically when its utilization decreases [OAaL14]. The server's power consumption at average utilization is also relevant to its power-proportional level. For instance, a server with a perfect power-proportionality means that it consumes 50% of server maximum power when the utilization is at 50%.

2.2.2 Energy consumption of computing, storage and network devices

This section relies on the data gathered for a 2016 study on United States data center energy usage [She+16]. It explores the energy consumption of the different data center components: the computing (servers), storage (disks) and network devices.

Figure 2.3 demonstrates the number of installed servers and the future trend from the 2016 study [She+16] classified according to the number of processors (1S and 2S+) and the type of vendor (branded and unbranded, unbranded usually refers to the self-assembled or original design manufacturer (ODM) servers). This study relies on three previous studies covering the 2000-2020 period and which are used to determine the approximate lifetime of servers: for 2006-2020, the server lifetime is about 4.4 years. Then the estimated annual servers' electricity consumption is shown in Figure 2.4 under the assumption that for each server, its power consumption is a linear function of its utilization.

The installed base of data storage equipment's estimate in terabyte (TB) is shown in Figure 2.5. This installed base is divided into solid-state drive (SSD) and hard disk drive (HDD) storage categories [She+16]. It uses past installed data (2010-2014) while SSD accounted for 8% in 2012, and forecast shipment data (2015-2019) revealing that SSD would grow to 22% by 2017.

The power consumption of traditional HDDs is usually higher according to per-disk level and is rarely dependent on the capacity of the disk. Instead, the power consumption of SSD units is more related to its capacity. Thus, it is more reasonable to convert both HDD and SSD storage from capacity to number of drive units to estimate the future trend. Based on industry feedback, HDDs should be able to provide a capacity of nearly 10TB per

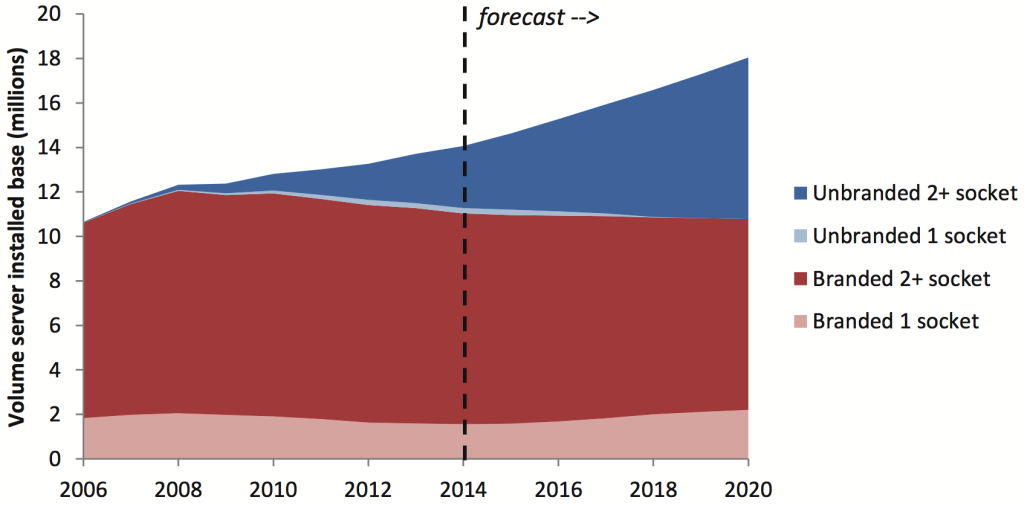


Figure 2.3 – Number of installed servers (data from [She+16])

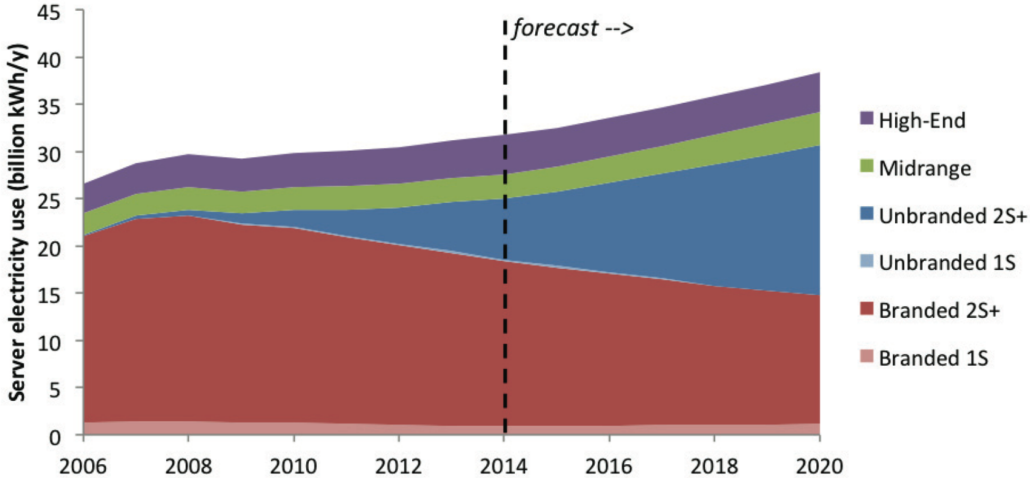


Figure 2.4 – Total U.S. Annual Direct Server Electricity Consumption by Server Class (data from [She+16])

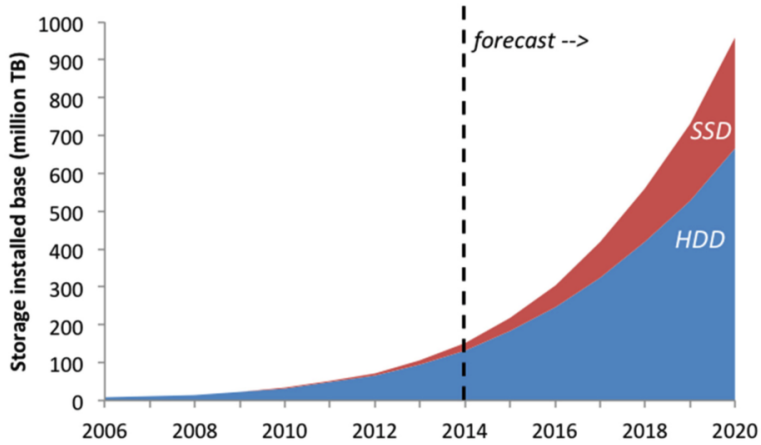


Figure 2.5 – Total U.S. Data Center Storage Installed Base in Capacity (data from [She+16])

drive unit in the future and SSD is assumed to reach an average capacity of 5 TB per drive unit [She+16]. The estimated energy consumption of storage is shown in Figure 2.6. It shows that the energy consumption of installed HDDs will continue to grow until 2018 and after, it will begin to decrease, while the consumption of installed SSD will keep growing.

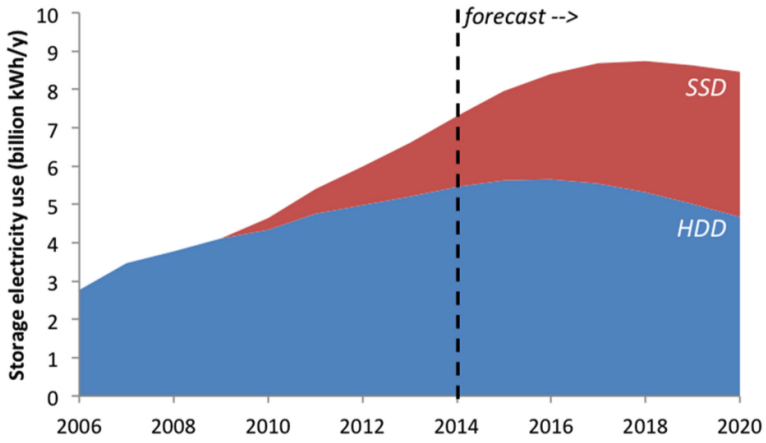


Figure 2.6 – Total U.S. Data Center Storage Electricity Consumption (data from [She+16])

The power consumption of network equipment is estimated by network port with different port speed [She+16]. The categories of network ports can be divided into four: 100 Mb, 1000 Mb, 10 Gb, and 40 Gb. The estimate per-port wattage are 1.4 W, 2.3 W, 3.6 W and 6.12 W for 100 MB, 1000 Mb, 10 Gb and 40 Gb ports respectively. The energy consumption of network is estimated as the number of network ports and per-port power drawn for each port speed, based on the historical and forested data: the trend is displayed on Figure 2.7. Basically, lower speeds (100 and 1000 Mb), which are less efficient, should almost disappear by 2020.

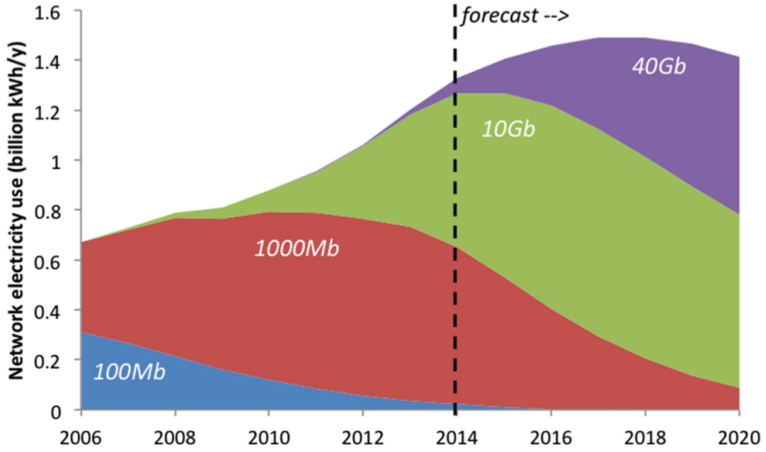


Figure 2.7 – Total U.S. Data Center Network Equipment Electricity Consumption (data from [She+16])

As shown in Figure 2.8, the growth rate of data center energy consumption has slowed down in recent years [She+16]. Server shipments experienced a five-year rapid growth period with 15% annual growth rate from 2000 to 2005. From 2005-2010, the annual growth rate fell to 5% probably due to economic recession and also because the energy efficiency mechanisms started to be implemented in server, storage, network and infrastructure along with virtualization techniques. After 2010, the growth rate drops to 3% and is expected to stay stable by 2020.

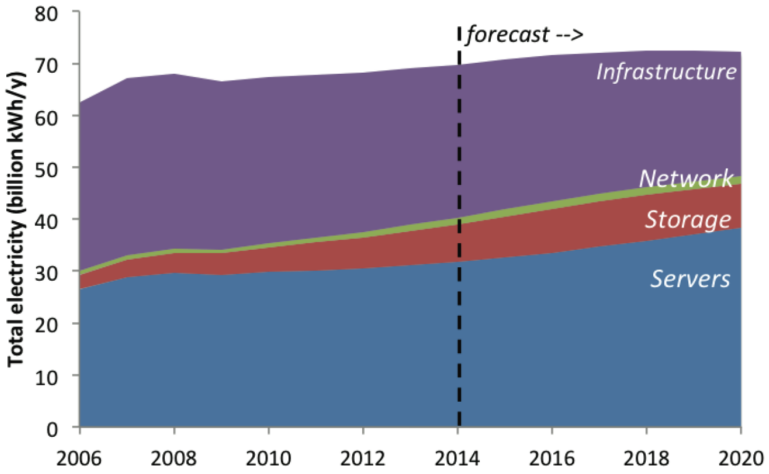


Figure 2.8 – Total Data Center Electricity Consumption (data from [She+16])

Through the examination of the power consumption share in DCs and the extrapolation for future trends, this study show that despite energy-efficiency improvements, the overall energy consumption of DCs will keep growing, and the server part will stay dominant.

2.2.3 Green metrics

In order to assess the energy-efficiency of data centers, several metrics have been proposed in literature. Table 2.1 details several existing green metrics for data centers.

Metric	Description	Formulation
PUE	Power Usage Effectiveness	$\text{PUE} = \frac{\text{Total facility energy}}{\text{IT equipment energy}}$
CUE	Carbon Usage Effectiveness	$\text{CUE} = \frac{\text{Total CO2 emission}}{\text{IT equipment energy}}$
WUE	Water Usage Effectiveness	$\text{WUE} = \frac{\text{Annual site water usage}}{\text{IT equipment energy}}$
ERF	Energy Reuse Factor	$\text{ERF} = \frac{\text{Reuse energy}}{\text{IT equipment energy}}$
ERE	Energy Reuse Effectiveness	$\text{ERE} = \frac{\text{Total energy - Reused energy}}{\text{IT equipment energy}}$
DCiE	Data Center Infrastructure Efficiency	$\text{DCiE} = \frac{1}{\text{PUE}}$
DCP	Data Center Productivity	$\text{DCP} = \frac{\text{Useful work}}{\text{IT total facility power}}$
ERP	Energy-Response time Product	$\text{ERP}^{\pi} = \text{E}[p^{\pi}] \cdot \text{E}[T^{\pi}]$ [1]
GEC	Green Energy Coefficient	$\text{GEC} = \frac{\text{Green power}}{\text{Total facility power}}$

Table 2.1 – Green metrics

The most widely used and industry-acknowledged metric for current data centers is the PUE [Bel+08; Gre14]: Power Usage Effectiveness. Usually, a ideal PUE value is equal to 1.0: this indicates that the energy consumed by IT equipment is same as the total facility energy.

The carbon dioxide emissions can be measured by their electrical equivalents (depending on the energy source) and multiplied their local carbon factor. The Carbon Usage Effectiveness (CUE) [Aze+10] is calculated through dividing the total carbon emission equivalents of total facility energy by the total IT energy consumption. The CUE is described in kilograms of carbon per kilowatt-hour. The lower CUE value indicates that lower carbon emission is associated with the data center operations.

1. where $E[P^{\pi}]$=average power consumed under policy π, $E[T^{\pi}]$=mean customer response time under policy π

The water usage effectiveness (WUE) [ABP11] expresses water used for cooling and regulating humidity. Combined, the PUE, CUE and WUE allow the data center operators to quickly evaluate the energy, carbon emission, and water aspects. While PUE has been broadly adopted in the industry, the CUE and WUE constitutes less used extensions of the xUE family.

The Energy Reuse Factor (ERF) [Pat+10] represents the ratio of the data center energy that is reused in the facility (e.g., the form of warm water or warm airflow is considered as a kind of reused energy) and the total energy consumed (including IT, cooling system, UPS, lighting, etc.). ERF will range from 0.0 to 1.0. Energy Reuse Effectiveness (ERE) is based on ERF, it is used for measuring the reusing heat generated by data center for other useful purposes.

Data Center Infrastructure Efficiency (DCiE) [Ver+07; Bel+08] is a metric used to describe the energy efficiency of a data center. DCiE is calculated by dividing the total power delivered to the server racks in the data center by total facility power. In addition, DCiE can be showed as the reciprocal of PUE, where DCiE $= \frac{1}{\text{PUE}}$.

Data Center Productivity (DCP) [Bel+08] refers to quantify the useful work in relation to the energy it consumed. The data center is considered as a black box – the power goes into the box and the heat out. Similarly, the data goes into the box and the result out of the box – a net quantity of useful work has been done in this box. Energy-Response time Product (ERP) [GAL16; Cer+16] is used to capture the trade-off between the performance and energy consumption: a delay is occurred when the server switches from a sleep state to a functional state, resulting in an energy cost.

Green Energy Coefficient (GEC) is a metric providing the ratio of the energy generated by renewable sources over the energy production from all the energy sources. For example, France, on November 2015 [GOM17], has a GEC of 0.14 as 14% of the energy use is sourced from green providers [2] like solar, wind and hydro.

2.3 Energy saving approaches

In this section, we present recent efforts on reducing data center energy consumption. These efforts generally address the problems of: energy efficiency, energy proportionality and integrating renewable energy in DCs. The energy efficiency and energy proportionality problems in particular concentrate on optimizing the energy consumption at the server level (e.g., optimize processor used frequency, turn on/off the underutilized server). The solutions of integrating renewable energy are mainly divided into without-battery approaches (i.e., this relies on supply-following model with a classification of job types, their characteristics enabling the workload to match the renewable energy supply) and with-battery approaches (i.e., this simply stores the surplus energy from renewable energy supply into battery for future use).

2.3.1 Energy efficiency

The Dynamic Voltage Frequency Scaling (DVFS) technology is widely implemented in DC servers. It is an efficient mechanism that enables servers and other devices to increase/decrease dynamically the operating frequency and voltage. Mathematically, the dynamic power consumption of processor [Bel+11] can be expressed as:

$$P = \frac{1}{4}\alpha C V^2 f \tag{2.1}$$

2. http://www.rte-france.com/fr/eco2mix/eco2mix-mix-energetique

where α is the value of switching activity and C is the capacitance, they are both physical parameters that are determined by the system design. V represents the processor voltage, and f corresponds to the clock frequency. Formula 2.1 illustrates that the power consumption of processor is approximately proportional to its frequency and voltage[RTZ11; Riz+10]. However, there is no free lunch. Scaling down the processor voltage and frequency in order to reduce the power consumption of the processor will lead to a performance degradation [WCC14].

Yao *et al.* [YDS95] propose an off-line job scheduling algorithm in which each job has to be finished before its deadline by a single processor with variable speeds. In [IY98], *T. Ishihara et al.* propose a model of dynamically variable voltage processor to optimize the power-delay trade off. By determining the voltage scheduling for a given application, this model is able to minimize energy consumption without missing its deadline.

In early researches, the proposed DVFS algorithms focused on minimizing energy consumption while still meeting the deadlines. Quan *et al.* [QH01] present two off-line DVFS algorithms to save energy. The first algorithm seeks the minimum constant speed needed to satisfy each job deadline and shuts down the processor when it is idle, since the minimum constant speed usually has a lower power consumption. The second algorithm builds on the first one and integrates a global voltage schedule. The authors prove that the global voltage schedule outperforms approaches that only uses the minimum constant voltage.

Hua *et al.* [HQ03] propose an analytic optimal solution and a linear search algorithm for the dual-voltage system. The voltage set-up algorithm determines the optimal voltage that minimizes the total energy consumption without missing the deadline of applications. The authors also point out the hardware overhead cannot be ignored (e.g., the area and power on the voltage regulators). When there are more than two voltages available, a linear approximation algorithm is proposed for determining the optimal number of voltage levels taking into account the hardware overhead. The experimental results demonstrate that when the voltages are set up properly, the DVFS technique can reduce energy consumption significantly.

Several studies have shown that a high temperature in the operation of data center leads to a lower reliability of the servers and increase the cooling energy overhead. Cupertino *et al.*[Cup+15] addressed the problem of energy-thermal efficiency of data centers. The authors proposed a cooling model of data center to present cooling devices (chiller, fans) power consumption, in terms of ambient temperature, inlet air room operation temperature and partial load. More recently, Sun *et al.* [SSP17] studied the spatio-temporal thermal-aware scheduling for homogeneous data centers. In their proposed scheduling algorithm, DVFS is not only used to regulate the temperature of servers during their executions, but also to maximize the throughput (i.e., minimize the makespan for high performance computing applications), so as to optimize the energy consumption.

Although DVFS provides a way to control power consumption of the CPU, it still has limited effect on power consumption in comparison with the overall power drawn by servers. Since an idle server consumes up to half of the maximal server power, the power proportionality needs to be considered as an important issue for energy consumption.

2.3.2 Energy proportionality

When certain electronic components enter an idle state, Dynamic Power Management (DPM) exploits this period and turns off these components to an inactive state in order to reduce energy consumption. ACPI (Advanced Configuration and Power Interface [Ind]) specification provides an open standard which can be used to perform power management of hardware components. It enables the server to switch its unused components such as

processor physical cores and Ethernet cards to sleep-mode.

If the server is able to be switched-on/off dynamically, the effect on reducing power consumption will be more obvious. Vary-on/vary-off (VOVO) policy reduces the aggregate-power consumption of a server cluster during periods of reduced workload. The VOVO policy switches-off servers so that only the minimum number of servers that can support the workload are kept alive. Yet, much of the applications, such as web applications, running in a data center must be online constantly.

Lin *et al.* adopt DPM schema in [Lin+13]. They consider that the servers are homogeneous in data center and a discrete-time model. The operating costs and switching costs are both taken into account for optimizing the data center energy consumption. The operation costs model the cost incurred by an active server and the switching costs represent the cost to toggle a server into/out sleep mode. The authors propose an online algorithm called Lazy Capacity Provisioning" (LCP) to dynamically adjust the number of active servers to meet the current system requirements. The results show that LCP guarantees that the cost is not larger than 3 times more than the optimal solution and LCP algorithm is proved as 3-competitive (i.e., an algorithm is n-competitive means that the cost of algorithm is less than n times of the cost of optimal offline solution).

Gandhi *et al.* [Gan+09] investigate the problem of finding the optimal power allocation for a server farm so as to get maximum performance with a fixed power budget, by using the optimal frequencies of servers. The authors employed DFS (Dynamic Frequency Scaling), DVFS (Dynamic Voltage and Frequency Scaling) and DVFS+DFS for various workloads to measure the effects of CPU frequency scaling on power consumption of a single server. The experimental results show that the power-to-frequency relationship is linear for DFS and DVFS when the workload is CPU bound and a memory bound workload is usually cubic and the relationship for DVFS+DFS is always cubic. Meanwhile, the arrival rate, the maximum speed of a server, the total power budget also affect the the mean response time for a server farm. They propose a queuing theoretic model to predict the mean response time that is used to determine the optimal power allocation. In their later work, they propose a class of Distributed and Robust Auto-Scaling (DRAS) policies to optimize power management for computationally intensive server farms. DRAS policies aim to lower the power consumption and to maintain a near-optimal response time. The key idea of DRAS policies consists in dynamically adjusting the server farm capacity to meet the incoming demand. The experimentation uses Intel's LINPACK [Cor] workload and a 20-servers test-bed. The results show that waiting for some time (with the best settings in DRAS) before turning-on and off servers can reduce power consumption by as much as 30% while increasing by 4% the response time.

Niyato *et al.* [NCS09] propose an optimal power management (OPM) of a batch scheduler for an individual server farm. The OPM aims to minimize the power consumption of the server while meeting the performance requirement. It observes the state of a server and dynamically switches the operation mode of the server (i.e., active or sleep) in order to reduce power consumption. The optimization problem of OPM is formulated as a constrained Markov decision process (CMDP) and solved by transforming it into a Linear Programming (LP) problem. The authors also study the problem of job assignment to multiple server farms, ensuring the power consumption and network costs are minimized. The numerical results show that with OPM, the performance in terms of job waiting time and job blocking probability are met while the power consumption is minimized. However, although dynamic switching on and off unused server can further reduce the energy consumption, such operations may bring several inconveniences.

Besides, Orgerie *et al.* [OLG08] point out that if the interval between two operations (i.e., switch on and off) is too short, this kind of operations becomes unnecessary and should be

avoided. They propose the Energy Aware Reservation Infrastructure (EARI) framework that alternatively switches on and off servers in a clever way. Similar approaches can be found in the mobile computing community [BBDM99; Che+07; KR07] where in order to reduce energy consumption a server can be left purposely idle for a bit of time before turning it off, thus avoiding some unnecessary operations (i.e. re-turn on or off).

Finally, Villebonnet *et al.* [Vil+14; Vil+15] propose a platform including two different physical architectures (i.e. ARM and x86) in order to archive better energy proportionality. Their purpose is to select the most suitable architecture to execute the application taking into account the hardware's energy consumption. Particularly, the x86 architecture is more powerful than ARM, it is usually used to execute the applications which have higher requirements on performance. In contrast, the ARM processor Cortex-A15 is much less powerful compared with x86 processor, but its energy consumption is pretty low. Thus, once the performance demand of application is reduced, the applications can be migrated to the ARM architecture to save energy.

2.3.3 Virtualized Infrastructure for Cloud Computing

The technique of virtualization has fast become a fundamental technology in cloud data centers. Relying on virtualization technology, a physical machine (PM) can be disaggregated in many virtual machines (VMs). In other words, it enables to run multiple operating systems on a single server. Each operating system is running inside a virtual machine which contains part of the server resources. Virtualization can reduce the energy consumption of data center - less servers to host, less cooling to pay for, and higher energy efficiency. In an Infrastructure as a Service (IaaS) cloud, the server allocates amounts of resources to a VM at creation time and these amounts can be expressed with relative values (i.e. a fraction of the capacity of CPU/RAM and etc.)

Packing heuristics

To save energy in a single data center, a common goal is to reduce the number of powered-on PMs (ON PMs). In cloud computing, each job has a size that is represented by virtual machine's resource requirements. A job is assigned to run on a server. Each PM has a fixed resource capacity and the sum of actual resource requirements of all VMs on a server cannot exceed its capacity. A reasonable VM placement algorithm can effectively use server resources and reduce the number of used servers. e.g., More VMs can be placed on a single server than before. The VM placement problem is typically modeled as a n-dimensional bin-packing problem. The n represents the number of needed constraints.

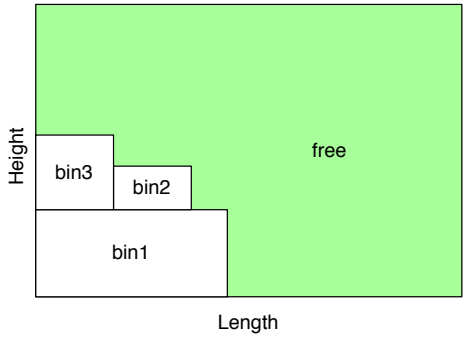


Figure 2.9 – Traditional 2-dimensional bin packing

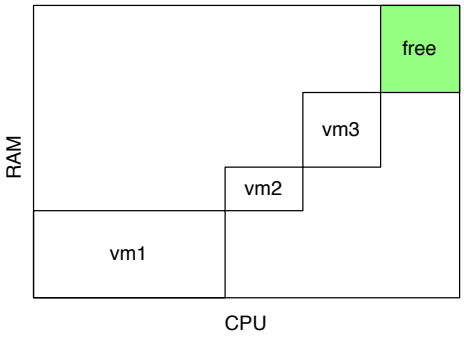


Figure 2.10 – VM 2-dimensional (CPU/RAM constraints) Parking Problem

As shown in Figure 2.9, a classic 2-dimensional bin packing problem can be seen as: how to place more bins with different sizes into a box with a fix size. Figure 2.10 demonstrates the difference between the classic 2-dimensional bin packing problem and VM packing problem. First, the global objective of both is to place as more bins as possible into the box. Then, in 2-dimensional bin-packing problem, the size of each bin is represented by the area of a rectangle. The problem becomes to minimize the remaining area of box (the green part). In VM packing problem, it takes into account for the side length of each rectangle (VM) instead of area, the sum of sides of all the rectangles can not exceed the length of box.

Hereafter, we list three classical heuristic algorithms to deal with the VM packing problem which consider a sorted list of PMs:

— First-Fit (FF): the FF approach is allocating to the *first* server portion which can accommodate the job resource requirements. FF finishes after finding the first suitable free portion. For allocating each job to a server, it begins to search from the first element of list.

— Next-Fit (NF): NF is a modified version of FF. It begins as FF to seek a free portion. When NF is called the next time, it starts searching from a server in the list from the previous allocation position, instead of searching from the beginning of list as FF.

— Best-Fit (BF): the BF deals with allocating the smallest free portion which meets the requirement of the job. This algorithm first searches the entire list of free portions and considers the smallest portion that is adequate. It then tries to find a portion which is close to actual job resources requirements.

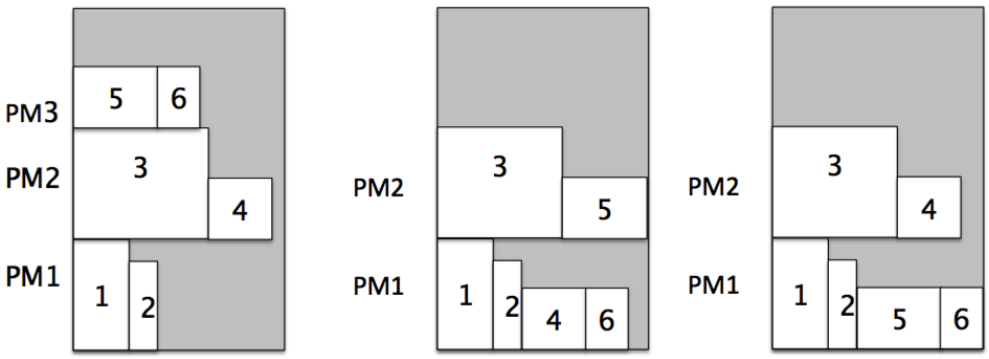


Figure 2.11 – Next-fit, First-fit, Best-fit

These three heuristics are illustrated on Figure 2.11. The classic bin packing problem aims to place a fix amount of items into a minimum number of bins. As mentioned in [CK99], the classic bin packing (BP) problem is proved to be an NP-hard problem. An online BP is different compared to an offline BP problem, it does not have any knowledge of subsequent items when it places an item. Usually, The *competitive ratio* [BEY05] is used for measuring the performance of online algorithms. It represents the ratio between the result of worst-case and optimal solutions.

Dynamic bin packing (DBP) is a branch of the classical bin packing problem. It assumes the items may arrive and leave at any time. An item is assigned upon arrival and it cannot move from one bin to another after it has been assigned. Li *et al.* [LTC14; LTC16] analyze the competitive ratios of Best fit and First fit scheduling algorithms. The authors prove that the competitive ratio of Best fit packing is not bounded for any μ where μ is the ratio of the maximum duration of an item to the minimum duration of an item. For first fit packing, it has a competitive ratio bounded above by $2\mu + 7$. In particular, their works indicate that First fit packing can achieve a better upper bound of $\frac{\beta}{\beta-1} \cdot \mu + \frac{3\beta}{\beta-1} + 1$ on competitive ratio while all the item sizes are strictly smaller than $\frac{1}{\beta}$ of the bin capacity ($\beta > 1$ is a constant).

Ren *et al.* [Ren+16] prove Next fit packing has an upper bound $2\mu + 1$ on the competitive ratio and a new upper bound $\mu + 3$ of First fit packing. Obviously, the First fit packing is better than Next fit packing when $\mu > 2$. The authors also indicate First Fit packing is near optimal for the DBP problem.

It is difficult to find an online solution which can always ensure that the result is optimal or near optimal. An alternative choice consists in using the existing heuristics (described above) to solve this kind of problems. Although it cannot always offer a perfect and stable performance, these greedy heuristics are simple and efficient features are often applied to online algorithms.

Meng *et al.* [Men+10] leverage VM multiplexing to improve resource utilization in cloud. The authors introduce an SLA model and a joint-VM sizing algorithm to calculate the capacity needs for consolidating VMs. A VM selection algorithm is proposed to find the VMs with the most compatible demand patterns. The selection algorithm first builds a correlation matrix [DHS12] for a set of VMs based on the historical or demand behavior of each VM. Then, the correlations are used as indicators for distinguishing the compatibility among VMs. The principle of VM multiplexing is used to consolidate the VMs to increase the resources utilization when the utilization of VMs are temporally unaligned. The results show that these solutions improve more than 45% resource utilization.

Khosravi *et al* [KGB13] investigate the problem of VM placement to reduce computing energy consumption and carbon footprint. The authors propose energy and carbon-efficient (ECE) architecture which is based on distributed data centers. The different locations of distributed data centers have different carbon footprint rates and PUE values. The proposed centralized-based global broker places each VM request in the most suitable data center with taking into account energy efficiency and carbon footprint parameters. Simulation results show that ECE saves between 10% and 45% of carbon footprint compared to three other heuristics.

As multiple-dimensional bin-packing problem is NP-hard problem, genetic algorithms (GA) based on evolutionary theory can been adopted as a way for addressing the VM placement optimization problem. Tang *et al* [Wu+12; TP15] propose a hybrid genetic algorithm (HGA) for the energy-efficient VM placement problem. The HGA extends the GA, it uses a repairing procedure to migrate the VMs from the current server to other servers until all the constraint violations are resolved. Meanwhile, a local optimization procedure is designed in the HGA to accelerate the convergence speed. Compared with the original GA, the results show that HGA has better exploitation capacity and convergence speed. Similarly, Wang *et al* [WWZ12] use a modified genetic algorithm to solve the energy-efficient multi-job scheduling problem.

Making advantage of live migration

Placing several VMs on a single PM can provide better use of PM resources. Consolidating VMs to fewer PMs enables to switch off some underutilized servers. Clark *et al.* propose to employ *live migration* of VM [Cla+05] as it allows to migrate a running VM from one PM to another and guarantees that VM performance does not degrade excessively during the migration. The principle of live migration is to transfer the VM's memory state from the source PM to the destination PM without stopping the execution of VM in pre-copy way [TLC85]. The memory pages are iteratively copied to the destination PM, a set of pages which is modified frequently is called *dirty pages*. Thereby, the VM has to be paused for a short time while copying all the dirty pages to the destination. Then the VM is resumed on the destination side.

In general, the VM migration can be divided into three phases:

1. Push phase: the VM on the source PM keeps running and the memory pages are repeatedly copied to the destination over the network. The dirty pages must be re-transferred iteratively during this phase.
2. Stop-and-copy phase: the source VM is stopped and the CPU state and any remaining memory pages are transferred to the destination, then the destination VM is executed.
3. Pull phase: if the destination VM found that it misses any memory pages, the source VM then transfers these missing pages.

As shown in Figure 2.12, Deshpande *et al* proposed a system for parallel live migration of multiple VMs across racks. The system exploits deduplication of VMs' memory images during multiple VM migrations in order to reduce the amount of data transferred over the network, thus saving energy. The results show that this system can reduce by up to 44% the network traffic load and by 27% the total migration time in comparison with the default live migration technique in QEMU/KVM.

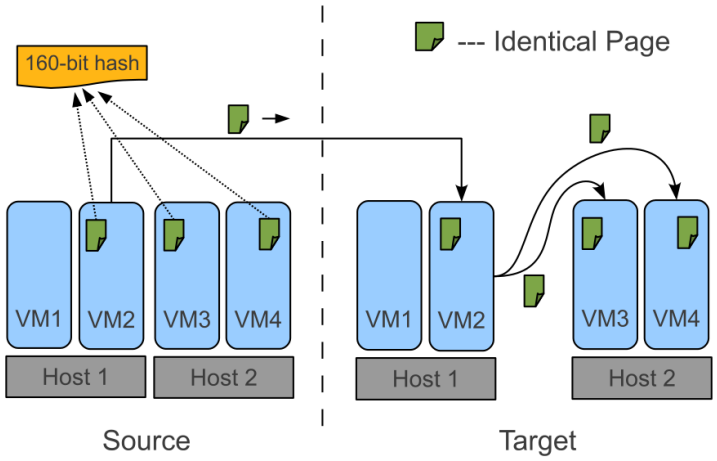


Figure 2.12 – Architecture of inter-rack live migration [DKG12].

The electricity price vary over time and across different locations, several prior studies [Liu+15; Rao+12; Yao+12; Guo+11; Pol+14; XL12] explore the opportunity of scheduling the jobs in multiple geographically distributed data centers. Polverini *et al.* [Pol+14] proposed a provably-efficient online algorithm which migrates computation across the geographical locations and employs real-time electricity prices.

Since in Cloud environments, each job has different arrival time and lifetime, the static VM placement algorithm is no more satisfying for the dynamicity of the system. It necessitates adequate optimization operations to adjust the number of ON PMs in the current system. The dynamic VM consolidation techniques have now been widely employed in modern data centers as they enable the data center to decrease the number of ON PMs [OAaL14].

As shown in [LWW07; FWB07; MGW09], an idle or underutilized server consumes nearly 50% of total server power. The study [BH07] shows that the average utilization of servers in data center are roughly between 10% and 50%. It is crucial to reduce the number of underutilized or idle servers to improve the data center energy-efficiency. On the purpose of switching-off several underutilized servers, the cloud resource manager needs to consolidate the workload into less servers.

Hermenier *et al* propose a constraint-programming based approach [Her+09], called *Entropy*, for consolidation for homogeneous clusters. Entropy mainly focus on solving the problem of VM placement and the problem of VM migration to reduce the use of physical machines. Entropy first computes a placement plan which aims at optimizing the number

of used physical machines. It then takes into account the migration overhead for consolidation to improve the plan and check the plan's feasibility, where the number of migrations required is decreased. The two phases are based on constraint solving [RVBW06; BJO13]. The first phase takes into account the VM CPU and RAM constraints and the second phase is based on sequential constraints and cyclic constraints. Their results show that Entropy can reduce by 50% the energy consumption of clusters as compared to static allocation.

Verma *et al* [Ver+09] analyze the server workload from a large data center. A number of characteristics relevant for semi-static (days, weeks) and static (monthly, yearly) consolidation is investigated. The study shows that there is a significant potential for energy savings if consolidation uses off-peak metrics for applications demand. The authors indicate that there is a strong correlation between applications and different hosted servers. The proposed correlation-aware consolidation method *Peak Clustering based Placement (PCP)* that takes into account the off-peak metrics can achieve energy savings and low SLA violation across powered-on servers.

Beloglazov *et al* [BB10a] propose an adaptive threshold-based dynamic server consolidation framework to ensure a high level of meeting the SLA. T-distribution is used for modeling the distribution of the server's utilization. Using statistical analysis of the historical data collected from each VM and the inverse cumulative probability function for the t-distribution enables the framework to auto-adjust the upper and lower utilization thresholds for each server. The idea of setting the lower utilization threshold is to migrate some VMs to other servers and switch-off the under-utilized server when the server's utilization lower threshold is violated. If the server utilization exceeds the upper threshold, some VMs are migrated from the server to others to avoid potential SLA violation. The VM placement problem is modeled as bin packing and exploit a Modification of the Best Fit Decreasing (MBFD) algorithm in order to solve the problem. The experimental results demonstrate that the proposed technique can reduce energy consumption and maintain the level of SLA violation $< 1\%$.

Chaisiri *et al* [CLN09] assume that every cloud provider offers two payment plans (i.e., reservation and on-demand plans) for users. Initially, the price in reservation plan is cheaper than on-demand plan. If the amount of reserved computational resources is not sufficient to meet the real-time demand, the user can buy additional resources in on-demand plan. The problem is formulated as a stochastic integer programming (SIP) to optimize the trade-off between the two plans and to minimize the number of used servers. The proposed VM placement algorithm solves SIP with two-stage recourse. Speitkamp and Bichler *et al* [BSS06; SB10] investigate the problem of server consolidation on using linear programming (LP) formulations. The authors formulate the problem as Static Server Allocation Problem (SSAP) and prove SSAP is strongly NP-hard. They propose SSAPv (i.e., v = with variable workload) model and design extension constraints for allocating VMs onto a set of servers, that ensures each VM is assigned to a target server without exceeding server's capacity and limiting the number of migrations so as to minimize the server cost. As NP-hard is hard to solve in polynomial-time, they propose an LP-relaxation-based heuristic to solve the linear programming formulations.

Meta-heuristics

Meta-heuristics have also been employed to solve the energy-efficient VM placement problem. Feller *et al* [FRM11] propose a dynamic workload placement algorithm based on the Ant Colony Optimization (ACO) meta-heuristic, to solve the workload consolidation problem. In their study, the VM placement problem is modeled as a multi-dimensional bin-packing (MDBP) problem. They propose a probabilistic decision rule that is used to guide

the ants choice towards the optimal solution. In each iteration of ACO, the ant chooses the VM with highest amount of pheromone and an heuristic information is employed to place the VM on the current server.

Pacini *et al* propose a particle swarm optimization (PSO) [PMGG14] scheduler to allocate VMs to servers under the IaaS model. In PSO approach, a VM is seen as a bee and a server corresponds to a space in a field of flowers. The density of flowers in the field represents the load that refers to total CPU utilization of servers (e.g. a lower density of flowers means that this server has more available resources). The results demonstrates that the PSO scheduler yields a better balance between throughput and response time than some simpler policies such as random policy. It also requires higher CPU, RAM and network usages in PSO thus leading to higher demands for energy. In addition, the authors also investigate the performance in terms of serviced users and created VMs when using the priority-based policy at the VM-level.

Over-commit resources technology

The above proposals investigate the bin packing problem to save energy by switching off unused resources. Another recognized reason of wasting the server resources (e.g., CPU and RAM) is due to lower VM resource utilization. Recent studies show that data center servers' average utilization are between 10% and 50%. As mentioned in [BH07], an idle or underutilized server consumes nearly 50% of total server power. A classical approach consists in running a fix amount of jobs using less servers thus optimizing the energy consumption of the data center. The resource over-commitment technique can directly increase the resource utilization of servers to optimize their energy-efficiency. Over-commit resource technique allows the broker to allocate VMs with a total configured resources, such as CPU and RAM, to the servers that exceeds their actual capacities. It allows servers to place more VMs than permitted. Consequently, it increases these servers' resource utilization and reduces the number of powered-on servers. However, over-commit resources technique may lead to servers overloading. Because the resource requirement of VMs placed on the server exceeds the server's available capacity, the VMs actual resource utilization varies with time and potentially increases the risk of server overloading. As a result of server overloading, it implies performance degradation of the VMs which are executing on this server. The overloading servers then need to migrate some of their VMs to other underutilized servers. The more the resource requirement of VM exceed the servers' capacity, the larger the probability of overloading. Further, it potentially increases the number of VM migrations that leads to an additional energy overhead.

In [Zha+12], Zhang *et al.* propose an online migration algorithm, called Scattered, to minimize migrations in the context of over-committed cloud environments. When a server is overload due to an aggressive resource over-commitment, Scattered measures the correlation between VMs and selects the best VM to migrate to the appropriate target server. The simulation results show that Scattered can maximize the number of over-committed VMs and reduce the number of migrations.

Ghosh *et al.* [GN12] study the risks associated with over-commit resources techniques. The authors present a statistical approach to quantify the risk of server overloading due to the fact that the VMs exceeded the capacity of allocated server. They determine the predicted upper bound on aggregate utilization by leveraging the one-sided upper tolerance limit [Gor+11]. The risk of exceeding server's capacity is then defined as the relative difference between the predicted upper bound and the set threshold (i.e., the value of empirical probability).

In [Dab+15a; Dab+15b], Dabbagh *et al.* propose an efficient resource allocation frame-

work for over-committed clouds to reduce the number of active servers while the migration overhead is minimized. On one hand, the predictor in this framework based on Wiener filter prediction approach predicts VM resource utilization, and takes resource over-commitment decisions to increase server utilization. On the other hand, the predictor predicts that the server will be overloaded due to over-commit resources and prepares to trigger VM migrations in advance to prevent SLA violations. The VM migration problem is formulated as as an integer linear program (ILP) and solves this problem by using fast heuristic with taking into account both the VM migration overhead and the server operating overhead. The obtained results show that the proposed heuristic uses less active servers and decreases the total migration overhead, thus reducing the amount of energy.

Over-commit technology allows to allocate a given amount of resources for more applications. However, while too many resources are committed, the resources consumed by applications may exceed the server's capacity, and it can result in a crash of this server. Hence, it is desirable to avoid committing too many resources in order to decrease the probability of crash, it refers to the practice of giving out an upper bound on virtual resource over-commitment.

2.3.4 A little more green

In today's world, the environment and climate change is a major topic of concern. Particularly, the global warming problem indicates an increase in the temperature of the atmosphere close to the Earth's surface. Burning fossil fuels, and subsequently CO_2 emissions, warm up the planet.

Exploiting renewable energy

The sun is one of the most promising clean energy technology, as it does not cause environmental pollution and it is inexhaustible by nature. Besides, renewable energy in the world has grown strongly in recent years. One reason is the solar-power generation efficiency significant increase. It enables the small-/medium-scale data centers to generate their own renewable energy [BCT16]. Thus they become self-sustainable and allow to reduce the fossil fuels (brown energy) consumption. As a consequence of the renewable energy success, the cost of producing green energy is becoming cheaper than brown energy [SS09; Goi+11]. The direct result is that the cost for the cloud users to accomplish their tasks in this kind of data centers is falling in a similar way when renewable energy is available [BCT16].

Unlike traditional infrastructures where energy sources are controllable, integrating renewable energy into a data center becomes difficult due to its intermittent and variable nature. Solar energy is considered as an admissible renewable source as solar panels are easy to install, they present a reasonable efficiency and the variations in their electricity production are not too abrupt (as for wind) [Goi+13]. Usually, most electricity generated by solar panels is during the day and its peak power always near the midday.

Goiri *et al.* [Goi+11] propose a parallel batch job scheduler called GreenSlot for a single data center partially powered by green energy (e.g., solar energy). The system assumes that there is no batteries and that conventional energy (e.g., brown energy which usually refers to fossil fuel energy) is only consumed when green energy is unavailable. GreenSlot is based on predictions of the availability of solar energy in the near future [Sha+10], it schedules jobs to maximize the utilization of solar energy without missing their deadlines. While the solar energy is not available, the scheduler switches to brown energy from the regular grid to avoid deadline violations. Furthermore, the scheduler selects the period when the electricity is cheaper than the other time period. One of the most valuable contribution here is

the testbed: the authors have built a prototype solar-powered micro-data center called Parasol [Bia+12] at Rutgers University, United States. Parasol comprises a small container and a set of solar panels. It also enables the users to monitor the power consumption of the infrastructure and to quantify how much energy is consumed from each available source. The results demonstrate that the data center reduces significantly brown energy consumption by implementing GreenSlot, compared to a baseline scheduler. In their recent work [Goi+12], Goiri *et al.* have proposed a MapReduce framework *GreenHadoop* for a data center partially powered by solar energy. Greenhadoop predicts the amount of solar energy in near future to schedule the MapReduce jobs in order to maximize the use of solar energy. When the solar energy becomes unavailable, GreenHadoop takes into account the price of brown energy and schedules the jobs during a cheaper time period. The experimental results show that GreenHadoop is able to increase the utilization of green energy and minimizes the electricity bill.

As the companies and individuals are inclined to move their workloads to the cloud, Haque *et al* [Haq+13] point the way of quantifiable green cloud services for environmentally conscious clients. GreenSLAs are proposed for this demand: it provides users the percentage of renewable energy used to run their workloads. The authors assume that the racks which provide GreenSLAs services are controlled by software. These racks can dynamically switch between the green and mix power supply (green and brown power). In particular, each job is submitted with an expected GreenSLAs value that represents the minimum percentage of green energy for completing the job. The providers decide to accept or reject the job depending on the computing capacity and green energy availability.

Supply-following load approaches

Some works [Bro+10; Kat+11] have studied the supply-following electric load approach where the electric load is scheduled depending on the availability of green energy supply. Unlike the previous work, Krioukov *et al* [Kri+11] propose an alternative model of supply-following loads for integrating renewable energy into data center. They investigate the problem of matching the variable and intermittent green energy with the workload energy demand. In their proposed load-following supply approach, loads can consume electricity while the green energy supply is available and not otherwise. A workload is adapted for this approach: it must have a large scheduling slack [Kat+11] and it can be suspended or rescheduled. The slack enables the energy consumption of workload to align with the variable green energy supply. The simulation is based on real-world traces (batch job traces from a cluster of 576 servers at UC Berkeley and wind power data from the National Renewable Energy Laboratory (NREL) database). The results show that supply-following job schedulers can increase by 40-60% the green energy consumption in their use-case.

Unlike the supply-following loads approach, Zhang *et al* [ZWW11] address the problem of scheduling the jobs across multiple geographically distributed data centers. The proposed middleware system *GreenWare* takes into account the time-varying electricity prices and the ratio of green energy among the geographically distributed cloud data centers for scheduling the jobs. The geographical locations enable the Internet service operators to get the peak power limit and availability of green energy of each data center. Benefiting from this information, GreenWare seeks the data center which has maximum green energy ratio or cheaper electricity cost. The authors model the request dispatching problem as a constrained optimization problem and solve it by using a linear-fractional programming (LFP). The experimental results demonstrate that GreenWare is able to maximize the use of green energy within a desired cost budget. However, when scheduling jobs far from the costumers, it may lead to an increased latency caused by the limited speed of telecommunication networks.

More recently, Chen *et al* [CHT12] have proposed a holistic workload scheduling algorithm *MinBrown* for distributing jobs among data centers in different geographical locations. Different from the work of GreenWare, the objective of MinBrown focuses on minimizing the total use of brown energy in all the data centers and ensure each job meets its QoS requirements (e.g., response time). The scheduling algorithm MinBrown considers the connection between green energy and the outside temperature of location. Further, it allows the jobs to move to other data centers via VM migration when the green energy is not available at the current location. Their studies have demonstrated that the combined green-/cooling-aware optimization can achieve up to 40% reduction in the brown energy consumption.

Storing and producing green energy

The intermittent and variable nature of green energy sources make them difficult to manage and control. Apart from the supply-following loads approach, the energy storage devices (ESDs, typically re-chargeable lead-acid batteries) can be seen as an another solution for green energy integration. ESDs are able to smooth out variations in energy generation. In this scenario, ESDs are used to store the surplus green energy production and to use it while the green energy supplies become unavailable. As the purpose of batteries is to store electrical energy in the form of chemical energy into battery and to convert that energy into electricity for later use, there is an inevitable loss of energy due to the chemistry nature of batteries. Ghiassi-Farrokhfal *et al* [GFKR15] investigate the performance of ESD-based system with different types of batteries. The energy can be seen as a continuous-time and fluid stochastic process [Wan+12b]. The authors first build a general model (non-ideal and ideal) for a class of non-ideal ESDs that includes all battery technologies. Employing stochastic network calculus enables this model to compute the analytic performance bounds on loss of power and waste of power probabilities due to the non-ideal ESD behavior or limited ESD size. The experimental results shows that the performance bounds relying on numerical simulations are quite close to the analytic performance bounds. The authors also use this model to evaluate the performance of various ESD technologies with their unique characteristics via simulations.

Photovoltaic (PV) cells convert the sunlight into DC electricity, the output is depending on the temperature and uniform irradiation. Due to the low conversion efficiency (13% 19%) of current solar cell technologies, maximum power point (MPP) [LH11] is usually used to describe the maximum output power of PV array. Li *et al* propose SolarCore to exploit green energy on using PV array for multi-core processors-based servers. The authors built PV equivalent circuit models to present the MPP with taking into account both the various insolation and temperature conditions. Each core uses on-chip voltage-regulator module (VRM) to perform per-core DVFS [Kim+08]. Per-core DVFS enables SolarCore to manage the power consumption of cores at a fin granularity to match the PV power output. Based on throughput-power ratio (the throughput speedup of a processor), SolarCore selects the most adaptive cores to meet the power budget. By applying load optimization and extracting additional solar energy, SolarCore yields 43% better performance than a fixed-power control scheme in terms of both green energy utilization and workload performance.

2.3.5 Novel cloud architectures

Distributed Clouds and Internet of Things

Nowadays, most of cloud providers implement their commercial clouds in large-scale data centers and operate them in a centralized fashion. Although they enable to achieve high performance computing ability and manageability, a powerful cooling system is needed to

lower the temperature of this large infrastructure equipment. Yet, the cooling system is expensive and consumes huge amounts of energy. Instead, previous work [CGH08] point out that small-size data centers have numerous advantages compared to large-scale data centers. First, small size data centers limit the amount of heat-dissipation and it can thus be more easier to manage. Then, smaller power consumption usually uses smaller power supplies and lower heat-dissipation overhead, which also reduces the cost and area of infrastructure equipment. Further, a small-scale data center is more suitable to build highly geographically distributed infrastructures.

Besides, a new model emerges: decentralized cloud infrastructures [Ber+14]. Cloud providers expect to improve the performance of their cloud and to leverage their available infrastructure. Indeed, telecommunication operators, like Orange, try to deploy micro data centers (20 to 50 servers by micro-DC) at the network border, closer to customers. In this new model, by deploying data centers closer to the user, the response time would greatly improve. This dissertation focuses on a small-/medium-sized data centers as they continue to keep increasing their share of the market. Placing computing and storage nodes at the Internet's edge has grown more and more popular in the recent years. These nodes are often placed in a small data center which is near mobile devices. In particular, *edge computing* [Sat17] enables to provide response time-critical cloud services for users.

The development of IoT (Internet of Things) community, the popularization of mobile devices, and emerging wearable devices bring new opportunities for context-aware applications in cloud computing environments [AF+15]. Since 2008, the U.S. National Intelligence Council lists the IoT among the six technologies that are most likely to impact U.S. national power by 2025 [Int08]. The disruptive potential impact of IoT relies on its pervasiveness: it should constitute an integrated heterogeneous system connecting an unprecedented number of physical objects to the Internet [AF+15]. A basic example of such objects includes vehicles and their numerous sensors.

Among the many challenges raised by IoT, one is currently getting particular attention: making computing resources easily accessible from the connected objects to process the huge amount of data streaming coming out of them. Cloud computing has been historically used to enable for a wide number of applications. It can naturally offer distributed sensor data collection, global resource and data sharing, remote and real-time data access, elastic resource provisioning and scaling, and pay-as-you-go pricing models [Abd+14].

However, it requires the extension of the classical centralized cloud computing architecture towards a more distributed architecture that includes computing and storage nodes installed close to users and physical systems [VRM14]. Such an edge cloud architecture needs to deal with flexibility, scalability and data privacy issues to allow for efficient computational offloading services [Hu+16].

While computation offloading to the edge can be beneficial from a Quality of Service (QoS) point of view, from an energy perspective, it is relying on less energy-efficient resources than centralized Cloud data centers [Var+16]. On the other hand, with the increasing number of applications moving on to the cloud, it may become untenable to meet the increasing energy demands, which are already reaching worrying levels [Ind]. Edge nodes could help to alleviate slightly this energy consumption as they could offload data centers from their overwhelming power load [Var+16] and reduce data movement. In particular, as edge cloud infrastructures are smaller in size than centralized data center, they can make a better use of renewable energy [Goi+13].

Offloading data to edge

Processing data streams analysis consumes enormous computational resources and the response time is usually crucial for many applications. Moving the data to the cloud for analysis can be a solution [IS11] in a variety of application scenarios that require enormous computational resources as well as QoS guarantees. However, it might pose a risk of network bottleneck if thousands data streams are produced from IoT devices at the same time and then transmitted to a central cloud (**core**) for quick analysis. Although lowering the analysis time profits large computational resources from cloud, it cannot avoid the time for data transferring through the network from user to the physical location of cloud, which might be thousands miles away [Bac+16]. Furthermore, the increasing number of data streams over the network consume a large amount of energy [Ned+08; FSR10; Fig+09; Bac+16].

To meet the demand of low latency response times, computation offloading to edge can be an answer [Zhu+11]. The edge represents small-scale data centers that are close to the data source. The concept of processing data at the edge is based on the advantage of lower latency than core, therefore been able to quickly return result to the device. Nevertheless, considering the large amount of data streams that needs to process, the core which has more computational resources may be a more energy-efficient choice.

2.4 Summary

This chapter has presented an overview of recent green technologies relating to resource management and power consumption optimization in the context of cloud data centers. Many green computing efforts focus on optimizing energy consumption at server-level through DVFS and power-aware scheduling based techniques. Since the virtualization technology begins to be widely used in data centers, the servers relying on virtualization technology are able to yield much better energy-efficiency than before. Recent works have addressed the problem of scheduling, placement and consolidation that enable the computation to run on fewer active servers, the energy consumption of data center has further reduced.

While there have been considerable works on optimizing power management, even though we build a more energy-efficient system, we still consume the energy from fuel fossil sources. Meanwhile, research interest continues to grow in integrating renewable energy into data centers. These studies show that integrating renewable energy can effective reduce dependence on traditional energy source and minimize CO_2 emissions. However, matching the variable and intermittent nature of many renewable energy sources with continuous energy demand is still a challenge. We believe that building a data center partially powered by renewable energy is emergent and necessary to minimize the environmental impact of Cloud infrastructures.

3

EpoCloud data center

This thesis has been done in the context of the EPOC project (Energy Proportional and Opportunistic Computing systems, Labex CominLabs, http://www.epoc.cominlabs.ueb.eu, 2013-2017). This project has funded four PhD students:

— Bárbara Dumas Feris on optical ultra high speed interconnection network for reconfigurable data centers.
— Sabbir Hasan on smart management of renewable energy in Clouds: from application to infrastructure.
— Gilles Madi Wamba on mixing constraint programming and behavioral models to manage energy consumption in data centers.
— Yunbo Li (this thesis) on resource allocation in a Cloud partially powered by renewable energy sources.

In the project of EPOC (Energy Proportional and Opportunistic Computing systems), we are focused on energy-aware task execution from the hardware to the application's components in the context of a mono-site and small DC (all resources are in the same physical location), which is connected to the regular electric Grid and to local-renewable-energy sources (such as windmills or solar cells). In this section, we present our prototype *EpoCloud* data center architecture, from hardware layer to middleware layer. This thesis dedicates to design the infrastructure of this prototype within the context of the EPOC project . The other components of this prototype have been defined with the other partners.

The rest of this section is organized as follows. Section 3.1 sets the principles of EpoCloud, the hardware architecture is detailed in Section 3.2. Section 3.3 presents an overview of real-world workload from a single data center. Section 3.4 describes several key components of our trace-driven simulator. Lastly, we summarize this section.

3.1 EpoCloud principles

The goal of EpoCloud is to design an energy-proportional computing system, which implies no energy consumption, whenever there is no activity. First, EpoCloud needs to be capable of switching on/off servers dynamically, thus leading to a lower aggregated-power consumption of the data center during periods of reduced workload. In this infrastructure, a vary-on/vary-off (VOVO) policy enables the broker to quickly adjust the number of active servers to meet the workload resources demand. By combining VOVO with dynamic

consolidation, an effective VMs placement plan and live migration, the broker can turn off the underutilized servers to reduce energy. EpoCloud also makes advantage of renewable energy (e.g., solar energy), that partially powers the data center. In this scenario, we are focused on aligning the workload with renewable energy supply. Later, we employ energy storage devices in EpoCloud, and explore the trade-off between system performance and energy savings.

3.2 EpoCloud hardware architecture

Figure 3.1 shows the architecture of EpoCloud. It is assumed that the data center is powered by both the regular grid and a renewable energy source. A switch is installed to mix both sources and to enable the data center resources to be connected with only one power cable (instead of two). The renewable energy is produced on-site and is used in priority while it is available. Particularly, the surplus renewable energy cannot be consumed immediately and will be directly considered as a waste (i.e., we assume that the production surplus is not re-send to the grid).

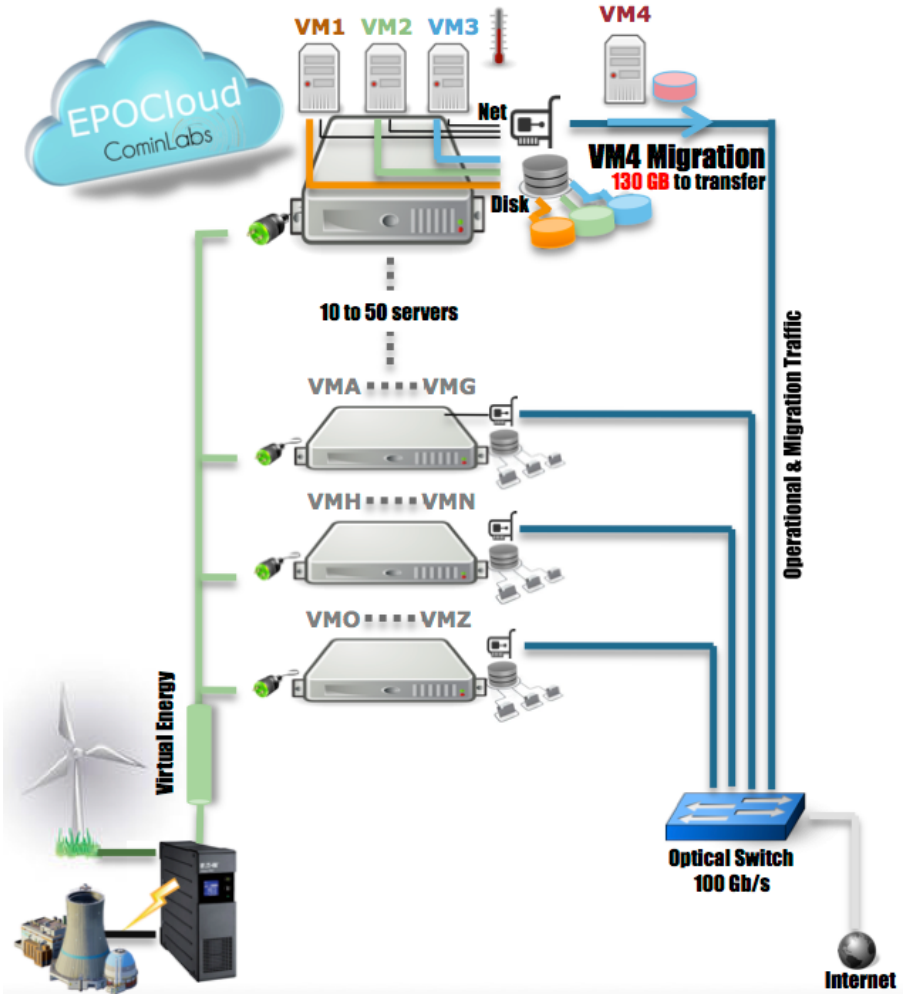


Figure 3.1 – EpoCloud architecture

3.2.1 High throughput optical networks for VM migration

For a network that is composed of 1 Gb/s bandwidth, migrating a VM with 4GB RAM and 128 GB of storage requires at least 17.6 minutes. Moreover, the virtualized data centers uses consolidation to run several VMs per server. According to the VOVO policy that we are considering, the data center has to migrate all the running VMs from one server to others while the data center manager requests to switch this server off to save energy.

In traditional data centers, each rack normally possesses one optical port. Thus, the total bandwidth is shared by the servers which are from the same rack. Assuming that a server can use the network with the entire 10 Gb/s bandwidth, it can take up to 2 hours to move all the VMs from current server and to redistribute them to others. If a rack is made up of 32 servers that are sharing the common 10 Gb/s bandwidth, migrating as such all 32 servers' VMs would take 53 hours. Thus we notice that the bit rate of interconnection network becomes a bottleneck for server consolidation.

Besides, storage is also one of the biggest energy consumer among diverse components of a data center. Since Storage Area Network (SAN) is primarily used for live migration in classical dynamic consolidation system, the VM storage is shared between all servers so that live migration is limited to transfer VMs' memory. However, a SAN typically has its own network of storage devices that impacts on the overall DC energy consumption. It can account up to 37% of the total power consumed by a data center [KT12]. EpoCloud proposes to liberate networks from SAN to optimize data center energy consumption. Instead, we introduce a strong hypothesis on the hardware architecture: EpoCloud does not have SAN and has a high throughput optical network (100 Gb/s) for each server. Particularity, for accessing data of applications and systems, we only use servers' local disks and the regular network linking the servers. This network employs a passive optical pod interconnect with pulse amplitude modulation, and its power consumption has been estimated to be roughly 20% of the classical electrical packet switch architecture [Fer+16].

3.2.2 Disk throughput

As mentioned previously, we assume that each server is connected through a 100Gb/s link. Each server has its own disk and pre-defined bandwidth. While the server is switched to sleep mode, its disks are also deactivated. We need now to make sure that disk throughput is high enough (at least as high as network bandwidth) not to become the bottleneck for VM migration.

We expect that the time of VM live migration is as short as possible. To realize this, a high network bandwidth is needed and Input/Output (I/O, e.g., read and write speed) rates respectively must be able to archive at the same rate as this bandwidth. Otherwise, it leads to a waste of network resources and the local disk I/O rate will become a new bottleneck. Due to the low I/O capacity of the traditional hard disk drive (HDD), it can no longer be applicable in this case. In contrast, SSD (Solid-state drive) relies on its excellent physical features, particularity with PCI-E (PCI Express) interface, that has a much higher read/write speed and a lower power consumption compared with HDD. There are four primary factors which affect the I/O performance of SSD.

The first one is the capacity. Different from HDD, a higher volume of SSD leads a higher read/write speed (e.g., 800GB >= 400GB with same interface).

As shown in 3.1, under the same interface (SATA), the read speed maintains its rate

1. Performance varies by capacity. (S = Sequential)
2. Performance measured using IOmeter with queue depth equal to 32. (R = Random)
3. Random 4 KB write performance using an out-of-the-box SSD

Capacity	S Read/Write (upto) [1]	R 4KB R [2] /W (upto) [3]	Form Fact
100 GB	500 MB/s / 200 MB/s	75,000 IOPS / 19,000 IOPS	2.5-inch SATA
200 GB	500 MB/s / 365 MB/s	75,000 IOPS / 32,000 IOPS	2.5-inch SATA
400 GB	500 MB/s / 460 MB/s	75,000 IOPS / 36,000 IOPS	2.5-inch SATA
800 GB	500 MB/s / 460 MB/s	75,000 IOPS / 36,000 IOPS	2.5-inch SATA
400GB	2 GB/s / 1 GB/s	180,000 IOPS / 75,000 IOPS	PCI Express* x8
800GB	2 GB/s / 1 GB/s	180,000 IOPS / 75,000 IOPS	PCI Express* x8

Table 3.1 – Different Volume of SSD

at 500 MB/s. The write speed is increased up to 460 MB/s by increasing the capacity of SSD. Once the interface is transited from SATA to PCIe (with 8 channels), the speed for reading is up to 2GB/s and 1GB/s for writing respectively. Indeed, the interface affects the performance significantly. The PCIe interface with multiple channels has a incontrovertible performance compared with other interfaces (e.g., IDE or SATA). Table 3.2 shows us the speed with different number of channels for PCIe.

Version	x1 [4]	x16	Speed / lane	Year	Encoding schema
1.0	250MB/s	4GB/s	2.5GT/s	22/07/2002	8B/10B [5]
1.0a	250MB/s	4GB/s	2.5GT/s	15/04/2003	8B/10B
1.1	250MB/s	4GB/s	2.5GT/s	28/03/2005	8B/10B
2.0	500MB/s	8GB/s	5.0GT/s	20/12/2006	8B/10B
2.1	500MB/s	8GB/s	5.0GT/s	04/03/2009	8B/10B
3.0	1GB/s	15.75GB/s	8.0GT/s	11/10/2010	128B/130B [6]
4.0	2GB/s	31.51GB/s	16.0GT/s	2014-2015	128B/130B

Table 3.2 – Performance with different number of channels for PCIe

We have:
PCIe serial bandwidth (MB/s) = Clock rate (MHz) $\times$ 1/8 (bit/8 = B) $\times$ number of lanes $\times$ encoding schema $\times$ transfer cycle (=1)
e.g. (1) PCI-E 1.0 x1 $bandwidth = 2500 \cdot 1/8 \cdot 1 \cdot 8/10 \cdot 1 = 250MB/s$
e.g. (2) PCI-E 3.0 x16 $bandwidth = 8000 \cdot 1/8 \cdot 16 \cdot 128/130 \cdot 1 = 15753.8461538MB/s$

From Table 3.2, the main bottleneck of SSD I/O performance is the interface version and its corresponding number of channels. Another important factor of SSD performance is the type of NAND Flash Memory Technology. We list 3 mainstream types of NAND Flash Memory:

— SLC (Single-level cell): fastest, highest cost, longest life among the 3 types. 1 bit of data was stored in each cell.
— MLC (Multiple-level cells) lower cost per unit of storage than SLC. It stores 2 bits per cell and has an expected longer life than TLC.
— TLC (Triple-level cell): slowest, least cost. It stores 3 bits of information per cell, with eight total voltage states.

In general, SLC NAND has faster speeds, lower power consumption and higher cell endurance than the others. However, SLC NAND stores less data per cell than MLC and TLC NAND. So, in terms of manufacturing, it costs more per megabyte of storage. Relying

4. Each direction.
5. Uses 10-bit symbols to encode 8-bit words, so 2 useless 20% efficiency lost.
6. PCI Express 3.0 introduced 128b/130b encoding, which uses 130-bit symbols to encode 128-bit words.

on faster speeds and longer expected life, SLC NAND flash memory technology is typically used in high-performance memory cards [SLM16]. MLC presents a compromise between SLC and TLC, with a sufficient performance and reasonable lifetime. This study shows that a 100 Gb/s I/O rate is reachable with current disk technologies. In addition, we can use RAID (Redundant Array of Independent Disks) to further increase the I/O speed.

3.3 Real traces

In this section, we describe the traces that are used for analysis and validation purposes for EpoCloud. These traces consist in two sets: one workload trace from a small Cloud data center, and one energy trace produced by photovoltaic panels.

3.3.1 Workload trace

This workload trace records the activity of VMs and servers separately for a small-scale data center[7] from the 25th of March 2014 to the 6th July 2014 (roughly 3 months and a half).

Processor (# core)	RAM (GB)	Frequency (per core)	count
12	48	2925	1
4	16	1994	2
8	24	2659	2
8	32	1995	2
12	102	2666	2
8	48	2526	2
8	52	2400	2
20	256	2393	2
24	256	2699	2
12	32	2792	4
12	64	2393	8
12	96	2792	8
20	192	2393	8
8	72	2526	9

Table 3.3 – Initial server's hardware configuration

As shown in Table 3.3, there are in total 55 servers with different hardware configurations. On average, each server has 30.5 cores and 96 GB of memory.

The Figure 3.2 shows the server average CPU and RAM utilization over the 3 months. We can see that the CPU utilization remains at a very low level: around 20% and a similar behavior is occurred on RAM usage. Although the RAM utilization is roughly twice higher than CPU utilization, it only uses half of the memory resources.

Figure 3.3 displays the overall utilization of VMs. The CPU curve represents the average CPU utilization among all VMs. We can find that VMs' CPU utilization remains under 30% and the average RAM usage is more variable. Meanwhile, the average VM CPU usage is extremely low, this is in accordance with the poor utilization of server. This observation of workload trace brings us new opportunities to reduce the overall energy consumption, in term of optimizing the server CPU utilization.

7. This trace is from a small-scale data center provided by EasyVirt.

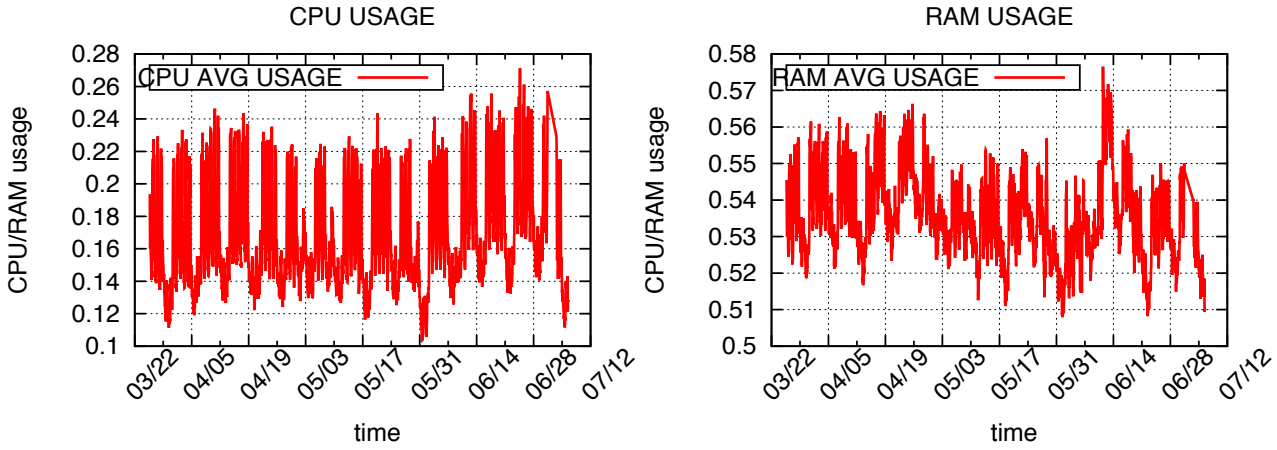


Figure 3.2 – Server CPU (left) and RAM (right) average utilization (%)

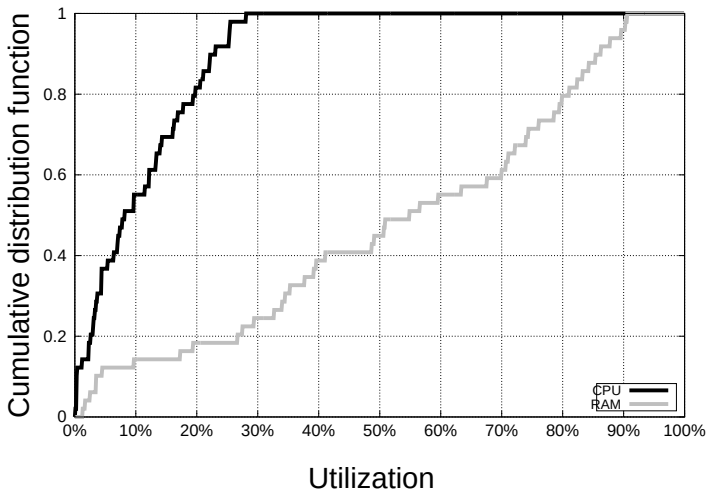


Figure 3.3 – VM CDF

3.3.2 Solar energy trace

Photovolta [8] is a solar farm for scientific research placed in the University of Nantes campus. It is composed of 4 *Sanyo HIP-240-HDE4* panels. Each panel takes 1.38 m^2 area and so the 4 panels takes in total 5.52 m^2. The theoretical peak power of the 4 panels is 960 Watts.

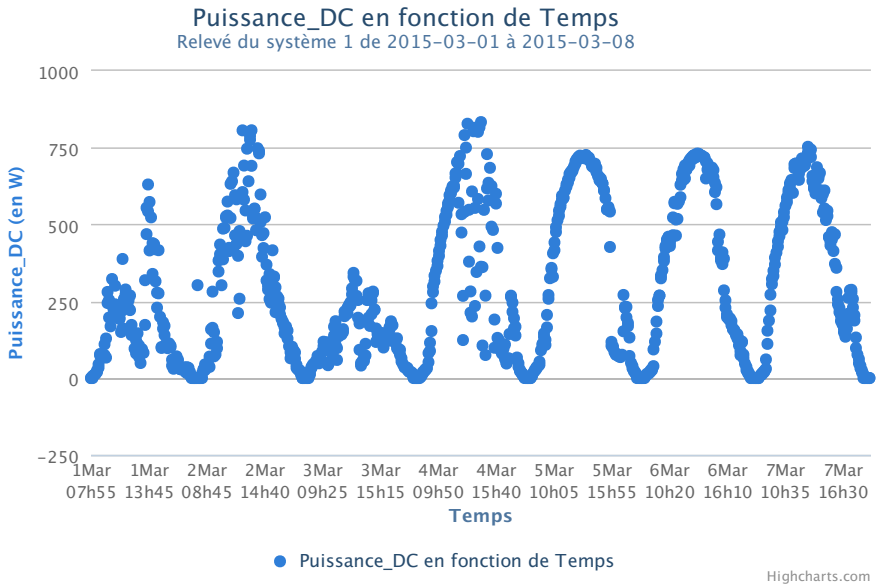


Figure 3.4 – A week records of solar energy (from March 1 to March 8, 2015).

Figure 3.4 shows a week of solar energy from the database provided by Photovolta. Both the environmental conditions and the photovoltaic production data are logged every five minutes. We can observe that the peak power of solar power generation occurs at noon every day and there is no solar power generation at night (thus data are cut during nights on the graph). Particularly, the weather was cloudless and sunny on March 5, thus generating more renewable energy than the other days of the week.

3.4 Trace-driven simulator

At the beginning of this thesis, we made a short state-of-the-art review of Cloud simulators able to provide the energy consumption of a data center and to integrate several energy sources. None of the literature simulators (SimGrid, CloudSim, GreenCloud, etc.) were fulfilling these requirements. Thus, we decided to develop a java-based simulator for experimentation, particularly with regard to scheduling for energy-aware infrastructures and application services. We list several basic functionalities of our simulator:

— enable users to model cloud data center infrastructure environment (heterogeneous or homogeneous clusters);
— enable users to use their customized energy module for server and network;
— enable users to simulate resource allocation policies. For instance, VM-mapping-PM algorithm;
— enable users to integrate their workload and renewable energy supply traces;
— enable users to simulate workload scheduling algorithm in geographically distributed data centers.

8. http://photovolta2.univ-nantes.fr/accueil.php

The structure of simulator is shown as follows.

```
Battery
├── Battery.java
Epoc
├── InitialSimulator.java
├── Job.java
├── Process.java
├── Vmm.java
├── impl
│   ├── CostDetailOnServer.java
│   ├── Datacenter.java
│   ├── DatacenterBroker.java
│   ├── JobB.java
│   ├── JobT.java
│   ├── MipsRam.java
│   ├── Server.java
│   ├── Vm.java
│   └── result
│       ├── ResultAlgorithm.java
│       ├── ResultEnergy.java
│       ├── ResultIteration.java
│       ├── ServerSwichState.java
│       └── UpdateMigrationProcess.java
├── mainClass.java
Powermodule
├── PowerModel.java
├── PowerModelBench.java
├── PowerServerEx1.java
├── PowerServerEx2.java
Utils
├── AlgorithmUtils.java
├── CSVUtils.java
├── Consolidation.java
├── FunctionUtils.java
├── GraphicUtils.java
├── ListUtils.java
├── LoggerUtils.java
├── MyFolderReader.java
├── PrintUtils.java
4963 total line
```

More details about each module can be found in Appendix B. Moreover, the simulator is modular and can support user-contributed contributions.

3.5 Summary

This chapter describes the concept of EpoCloud. EpoCloud represents the prototype of a micro-DC integrating on-site renewable energy supply. EpoCloud focuses on reducing energy consumption by: (1) optimizing the energy consumption of distributed infrastructures and service compositions in the presence of dynamic service applications, while maintain-

ing SLAs; (2) designing an intelligent resource management, which performs opportunistic scheduling by leveraging renewable energy availability, then exploring the trade-off between system performance and energy saving. Within the context of EpoCloud, we will now detail the energy-aware resource allocation algorithms that we proposed.

4

Opportunistic scheduling (PIKA) for maximizing renewable energy consumption in Cloud's data centers

Contents

From an energy point of view, these micro-data centers allow the study of new power supply solutions based on renewable energy, like wind or sun. Using these renewable energy sources can reduce the operating cost but, unfortunately, this kind of energy stays intermittent by nature.

To address this problem, we envision two solutions: investing in heavy expensive battery systems to smooth over the day the renewable energy production, or developing new applications management solutions adapted to the electricity production.

In this section, we propose to design a disruptive approach to Cloud's resource management which takes advantage of renewable energy availability to perform opportunistic tasks.

The micro-DC receives a fixed amount of power from the regular electrical grid. This power allows it to run the usual tasks. In addition, the micro-DC is also connected to renewable energy sources (such as windmills or solar cells) and when these sources produce electricity, the micro-DC uses it to run more, less urgent, tasks.

In order to achieve this energy-aware resource allocation, we distinguish two kinds of jobs to be scheduled on the data center: the web jobs which represent jobs requiring to run continuously (like web servers), and the batch jobs which represent jobs that can be delayed and interrupted, but with a deadline constraint. This second type of jobs are the natural candidates of the opportunistic scheduling algorithm.

This section presents PIKA, a framework aiming at reducing the brown energy consumption (i.e., from non-renewable energy sources), and improving the usage of renewable en-

ergy for mono-site data center. It exploits jobs with slack periods, and executes or suspends them depending on the renewable energy availability. By consolidating the virtual machines (VM) on the physical servers, PIKA adjusts the number of powered-on servers in order for the overall energy consumption to match with the renewable energy supply. Using simulations driven by real-world workloads and solar power traces, we demonstrate that PIKA consumes 44.9% less brown energy and increases by 110.1% the renewable energy integration ratio in comparison with the baseline greedy algorithm from literature.

The remainder of this section is organized as follows. Section 4.1 formalizes the problem. We give a brief overview of PIKA framework in Section 4.2 and we present our formalization of PIKA in Section 4.3. The experimental setup is explained in Section 4.5. Section 4.6 evaluates the various policies based on simulations. Section 4.7 concludes this work.

4.1 Problem formulation

PIKA targets small/medium-size mono-site data center (typically between 20 and 150 servers). The data center consists of several *Physical Machines* (PMs). We assume the computing environment in the data center to be heterogeneous, meaning that the PMs can have different hardware. Each PM has limited resources (CPU, RAM, network) and has its own disk storage. We assume that the data center has no centralized storage system (such as a Network Attached Storage for instance) as described in Chapter 3. The PMs with different capacities and performance may potentially lead to different energy consumption values for a given VM.

We assume the data center has dual brown and renewable energy power supplies. Each PM has a switch connected with both energy supplies and opts for renewable energy only if there is enough of it. Otherwise, the PMs consume the brown energy from the regular grid. Meanwhile, we assume there is no battery or the batteries are only used for emergency cases (like power outage).

4.1.1 Job

Our system does not only accommodate periodic jobs which means that jobs may have different lifetimes and can arrive at anytime. Once a job is submitted by a user, it is encapsulated into an individual Virtual Machine (VM). In our system, a VM is considered as the basic unit of resource allocation. It demands two types of resources from PM: CPU and RAM. The lifetime of a VM depends on the job it accommodates. When a job is finished, the VM is destroyed and its reserved resources on the PM are released.

4.1.2 Workload

We studied anonymized traces provided by the EasyVirt from a small-size Cloud data center as described in Section 3.3. These traces concern a VM hosting provider with 55 servers. The traces stretch from the 25th of March 2014 to the 6th of July 2014. They consist in the logs for real CPU, RAM, network and disk utilization of each server every 90 seconds. They also contain the client's requests for VMs with CPU and RAM sizes, and the submission dates. These traces present a realistic scenario in our context.

The Figure 4.1 illustrates the average CPU and RAM utilization of all the PMs during a normal week in the data center. Note that the average CPU utilization keeps low state and far below the average RAM utilization thus leading to a waste of resource. In such a context, Cloud providers usually resort to over-commitment. In particular, we will use a

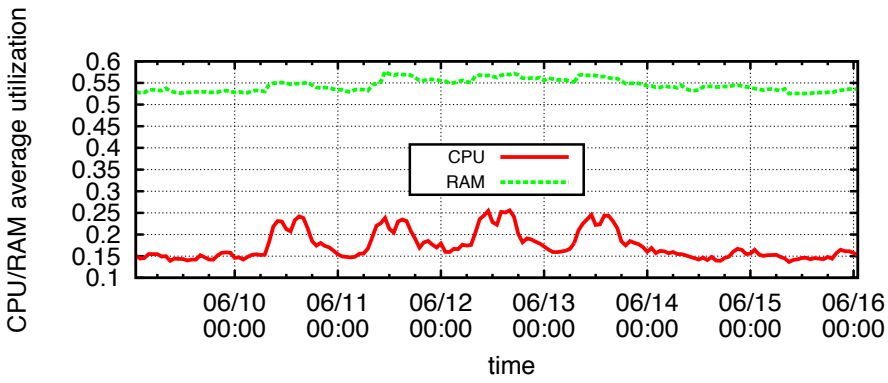


Figure 4.1 – CPU and RAM real utilization over one-week of real trace

CPU over-commit policy that will be described in Section 4.4. To reduce the number of ON PMs (powered-on Physical Machines), we now describe our proposed PIKA framework in the next section.

4.2 PIKA overview

Our proposed framework PIKA is designed as a centralized solution. It focuses on minimizing the brown energy consumption in a single small/medium-size data center. As the system is dynamic, PIKA performs the optimization operations periodically. The *optimization cycle* is defined as a slot, such that the *time* in our system is divided into a series of continuous *slots*. As shown in Figure 4.2, at the beginning of each slot, the broker executes three main steps. First, the broker checks each PM's state and suspends some jobs from the overloaded PMs. Second, the renewable energy predictor predicts the amount of renewable energy for the current slot and informs the broker about it. Then, the broker determines the number of ON PMs that can be supported by the renewable energy supply. Finally, according to the available resources from these ON PMs, the broker schedules the jobs that can be executed during the current slot. Each component of PIKA is described in the remainder of this section.

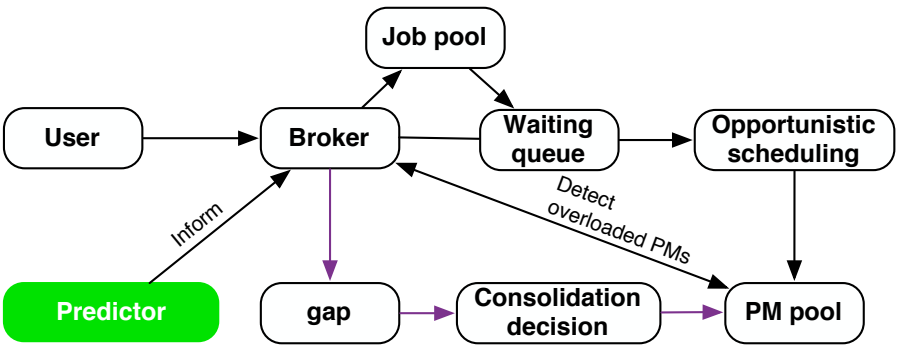


Figure 4.2 – PIKA framework

Gap: PIKA is designed to be aware of the variable and intermittent nature of renewable energy supply, which makes the *gap* module in PIKA play an important role for the other operations. The renewable energy predictor provides the energy availability that can be consumed in the current slot. The key feature of *gap* enables the broker to dynamically adjust the number of ON PMs following by the renewable energy availability. We detail how the *gap* works in Section 4.3.2.

Renewable energy predictor: we make several simplifying assumptions. The renewable energy prediction is performed at the beginning of each slot and the predicted renewable energy amount is used for only one slot. The short-term prediction's advantage is that it significantly reduces prediction errors caused by varying weather. Given an accurate prediction on renewable energy, the broker dynamically switches on and off PMs to adjust the energy consumption in order to maximize the renewable energy integration ratio.

Job pool: the key insight of PIKA is to align the workload energy consumption with the renewable energy supplies. We classify the jobs into two types according to their characteristics: *web job* and *batch job*. Each job is submitted with the following parameters: (t_b, T, t_e, t_d, v_i^{cpu}, v_i^{ram}): beginning time, type (web or batch), execution time, deadline, corresponding VM i's CPU and RAM requirements. They are pushed into the job pool by the broker. Once the broker gets the energy availability through the renewable energy predictor, it estimates the maximum number of ON PMs via the gap function in the current slot. To increase the chance to exploit more renewable energy, we employ the *slack time* for jobs into PIKA.

The *slack* is the most crucial factor in affecting the renewable energy integration ratio. It is given in Equation 4.1:

$$j_i^{\text{slack}} = t_d - t_b - t_e \tag{4.1}$$

Through j_i^{slack}, the broker enables a job to be delayed and this increases the chance of exploiting the renewable energy.

The *web job* (e.g., background job for web services) is uninterruptible with a little slack ($<$ 1slot). It has a higher priority than other jobs either on resource allocation or scheduling. For instance, when a web job is submitted, the broker pushes it into a specific job pool with higher priority than the waiting queue of the batch jobs. The web job is then placed on a PM which has sufficient resources to meet its VM resources requirements. Unlike the web job, the *batch job* (e.g., HPC job, typically compute intensive) can be interrupted or delayed within a slack. Furthermore, because web jobs have higher priority, the batch jobs must wait for all the web jobs to be placed before being allocated only to a ON PM (PM already hosting some web jobs). The batch jobs are not allowed to switch on an OFF PM if there is not enough renewable energy to power it during the next time slot. Specially, as shown in Figure 4.3: when a batch job's slack is strictly inferior to 1, it mutates as a web job that has the same priority as a regular web job in order to meet its deadline constraint. So, the *job pool* in PIKA is divided in two corresponding pools: the *web pool* and *batch pool*. Note that the VM placement process for web and batch jobs is distinct.

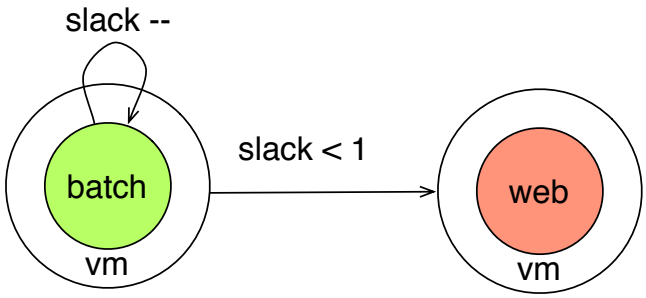


Figure 4.3 – The mutation of a batch job when its deadline is approaching

Flexible pooling of ON/OFF PMs: PIKA offers a mechanism for dynamically adjusting the number of ON/OFF PMs that tries to follow the variable renewable energy supply if possible (according to the workload). Firstly, we divide the PMs into two categories: ON PM and OFF PM for a given time slot. Both web jobs and batch jobs are capable of being allocated on an already ON PM if it has sufficient resources to meet their VM resources requirements. An OFF PM can be switched on in the following two cases: 1) there is no more

resources to meet a web job's VM resource requirements; 2) there is sufficient renewable energy for all the already ON PMs at that time and an extra amount from renewable energy supply allows to switch on new OFF PMs (called *potential ON PM* later in the section). A batch job is allowed to switch on an OFF PM when there is no renewable energy surplus if and only if it is mutated into a web job. The general VM placement process works as shown in Figure 4.4.

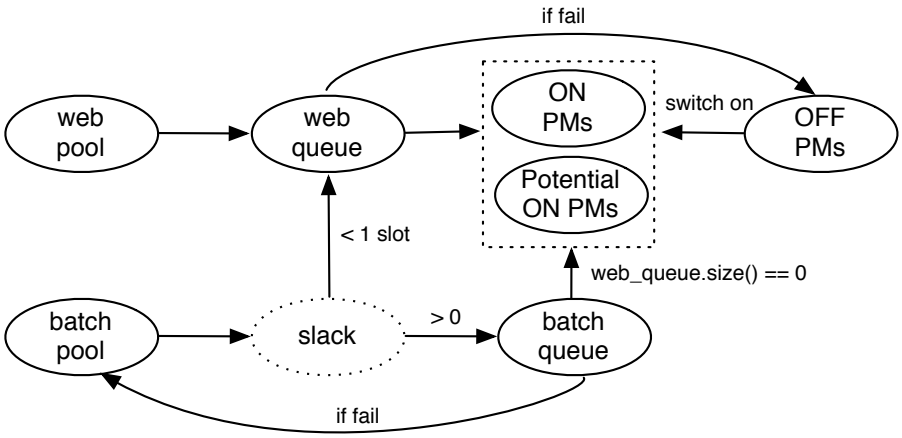


Figure 4.4 – VM placement for batch and web jobs

In this section, we have given an overview of PIKA's architecture. The next section is dedicated to the formalization of the various algorithms used by PIKA for the resource management and job scheduling.

4.3 Resource management and job scheduling

The general process of PIKA consists of four major steps detailed in Algorithm 1.

Algorithm 1 General process of PIKA

1: Step 1: Detect the overloaded PMs;
2: Step 2: Launch calculation of *gap* process and make the decision on whether to consolidate depending on the *count* (number of ON PMs);
3: Step 3: Update each batch job's slack;
4: Step 4: Select the jobs that can be executed in the current slot and place them adequately on PMs (VM placement algorithm). The decision of VM consolidation relies on the result of 2nd step.

The *count* represents the theoretical maximum number of PMs can be switched on, in terms of actual renewable energy availability.

At the beginning of each slot, the first step aims at finding the CPU/RAM utilization of a PM which exceeds its capacity. The gap value is calculated at step 2, the broker makes the decision of either proactive consolidating or switching-on OFF PMs. Then, the broker updates the slack of each batch job and places all the web jobs and the selected batch jobs in this slot. The consolidation decision at step 4 is based on the renewable energy availability.

4.3.1 Overloaded PM detection

Since we introduce the resource over-commit policy in PIKA, the varying CPU/RAM load of the jobs at each slot may lead the PMs to overload situations. At the beginning

of each slot, if a PM utilization exceeds its capacity, the broker first verifies the batch jobs utilization on this PM and suspend them if necessary.

$$\begin{cases} \text{Suspend the corresponding batch jobs,} & \text{If } U_{batch} \geq U_{exceed}. \\ \text{Push the web job to the waiting queue,} & \text{otherwise.} \end{cases} \tag{4.2}$$

We first consider suspending the current running batch job on an overloaded PM until the PM returns to normal state. If all the batch jobs on an overloaded PM has been suspended and the PM still exceeds its capacity, the broker migrates the web jobs to other ON PMs. Assume, for example, there is a PM which exceeds it CPU capacity. Firstly, the broker sorts the VMs in ascending order of CPU utilization. Then, the broker pushes the web job with largest CPU utilization to the web job waiting queue until the PM state becomes normal. If the PM returns back to normal state, the broker finishes the overload detection process. The queued web jobs are scheduled onto other ON PMs (or switches OFF PMs if resources are not enough on ON PMs) before any new allocation. The suspended batch jobs are pushed to the batch job pool, then they are either migrated to other ON PMs if possible or put in the waiting list of the next slot to run.

4.3.2 Gap

At the beginning of each slot, the broker receives the total electricity generated from renewable energy for the next slot. Ideally, if there is sufficient renewable energy to cover all the electricity need of all already executing jobs, the broker keeps the current PM state. Otherwise, the broker proposes a plan for VM consolidation in order to decrease the number of ON PMs. The metric $gap(t)$ is the key function of PIKA to aid the broker to dynamically adjust the number of ON PMs. The $gap(t)$ function is shown in Algorithm 2.

The $gap(t)$ (line 6) describes the difference between the renewable energy and current energy consumption in one slot. It is calculated as follows:

$$gap(t) = E_r(t) - \sum_i E_{s_i^{on}}(t) \tag{4.3}$$

Where $E_r(t)$ denotes the predicted amount of renewable energy at the t^{th} slot and $\sum_i E_{s_i^{on}}(t)$ denotes the current energy consumption of all the ON PMs (lines 3-5). According to the $gap(t)$, the broker decides whether to consolidate at the t^{th} slot.

If $gap(t) \geq 0$ (lines 8-17), it means there is extra renewable energy that may be used to switch on n OFF PMs and to run more jobs (opportunistically). Otherwise, it is necessary to switch off m PMs in order to reduce the current energy consumption. To determine the value of n, the broker first sorts the powered-off PMs list (func: powerOffServerList.`sortByServerScore()`)as follows:

$$e_{\text{efficient}} = \frac{1}{\frac{\max(E_s)}{\max(s_{mips})}} = \frac{\max(s_{mips})}{\max(E_s)} \tag{4.4}$$

Where s_{mips} is the CPU performance that is presented on megahertz (MHz), the $\max(E_s)$ is the maximum energy consumption of the PM s and the $\frac{\max(E_s)}{\max(s_{mips})}$ denotes the energy consumption per unit of MHz. The lower the value of $\frac{\max(E_s)}{\max(s_{mips})}$, the more efficient the PM is. The higher is the value of $e_{\text{efficient}}$, the more efficient it is. Each time the broker sorts the OFF PMs list in decreasing order and selects the n first PMs to switch on where n is equal to *count* (line 27).

Algorithm 2 *gap* function

```
1: INPUT: pmList, renewable energy availability in the current slot: sumGreenPower OUTPUT: the number of PMs to be switched-on or -off
2: pmList.splitTwoSubList() →powerOnPmList and powerOffPmList
3: for each pm ∈ powerOnPmList do
4:    sumPmPower += CurrentPower(pm);
5: end for
6: gap = sumGreenPower - sumPmPower
7: count=0;
8: if gap >= 0 then
9:    powerOffServerList.sortByServerScore();
10:   for each server ∈ powerOffServerList do
11:      while gap > 0 do
12:         gap -= pm.getMaxPower();
13:         pm.setPotentialOn(true);
14:         count++;
15:      end while
16:   end for
17: else
18:   powerOffServerList.sortByNumVM();
19:   for each server ∈ powerOnServerList do
20:      while gap < 0 do
21:         gap += pm.getCurrentPower();
22:         pm.setPotentialOff(true);
23:         count−−;
24:      end while
25:   end for
26: end if
27: return count;
```

When $gap(t) < 0$ (lines 18-26), there is no sufficient renewable energy to cover all the ON PMs energy consumption. It necessitates performing a consolidation to decrease the number of ON PMs. Due to VM migration also having an energy overhead, we need to minimize the number of migrations (func: powerOffServerList.`sortByNumVM()`). Thus, the broker seeks a set of PMs which has fewer number of VMs. The size of this set is equal to the value of *count* (line 27).

4.3.3 VM placement

As described in Section 2.3.3, by means of resource over-commit policy, the VM placement problem is transformed from 2-dimensional to 1-dimensional bin packing problem. In 1-dimensional bin packing problem, FFD (First Fit Decreasing) is a classic greedy algorithm which is proved to use: maximum $11/9 \times n + 1$ bins where n presents the number of bins in the optimal solution [Yue91]. The FFD and over-commit policy are combined in PIKA to optimize the resource allocation, the broker first places all the web jobs and then places all or a part of the batch jobs depending on the remaining ON resources. If there are some batch jobs that cannot successfully be scheduled in this step, the broker updates their slack time and reschedules them in a later slot.

But if there is not sufficient resources to place all the web jobs at the first step, the broker suspends a part of or all the batch jobs which are executed on current ON PMs. Then the broker places the web jobs on the current ON PMs. If it still cannot meet all the web jobs resources requirements, the broker activates one or more OFF PMs directly.

4.3.4 Consolidation and migration

Through the above analysis presented in Section 4.3.3, we detail our heuristic for the problem of dynamic consolidation in context of renewable energy. We subdivide the consolidation problem into the two following issues: when and how to consolidate.

When to consolidate

There is not enough renewable energy to cover all the servers energy consumption for the current slot, i.e., the *count* is negative. In this way, the broker attempts to decrease the number of current ON PMs if possible.

How to consolidate

The broker gets the number of PMs that should be switched-off via *count*. The broker seeks the PM which has the least number of web jobs and tries to migrate all the web jobs from this PM to others. While a PM is selected to be switched-off, the broker tries to migrate all the web jobs from this PM to the others and interrupts all the batch jobs on this PM. The affected batch jobs are pushed into the batch pool and wait for the broker to complete the web job placement. If there is not enough free resources on other ON PMs to migrate all the VMs of this PM, the broker does not perform the migration and aborts the consolidation process. Otherwise, if each web job on this PM can find another ON PM to host it, the broker migrates all of them and pushes all the batch jobs on this PM to the batch pool. Once all the VM migrations have been completed, the broker switches off this PM. The broker repeats this process until it reaches the expected number $count$ if possible.

4.4 Over-commit resource policies

As stated in Section 2.3.3, the resource over-commitment is an efficient method to increase the CPU/RAM utilization in order to minimize the number of ON PMs. Figure 4.1 provides an analysis on the real world workload traces. The VMs' average CPU utilization is around 15%, and around 50% for RAM utilization in comparison with their original VM resource requirement. It leads to a significant wastage of the CPU/RAM resources. The over-commitment can be used to further optimize the utilization of PMs. We define four resource over-commitment policies, which cover all the possible cases.

4.4.1 Non over-commit policy

This is the basic case, the resources are actually allocated to the VMs as requested initially.

$$\begin{cases} \sum\limits_{i} v_i^{\text{cpu}} \leq S_n^{\text{cpu}}, & \forall v_i \in S_n. \\ \sum\limits_{i} v_i^{\text{ram}} \leq S_n^{\text{ram}}, & \forall v_i \in S_n. \end{cases} \tag{4.5}$$

Where $\sum\limits_{i} v_i^{\text{cpu}}$ denotes the CPU resource request of all the VMs v_i^{cpu} on PMs S_n^{cpu}. The first inequation denotes that the PM n can simultaneously hosts multiple VMs i if and only if both the CPU/RAM resource requests of these VMs cannot exceed the PM n's capacities.

4.4.2 Over-commit RAM policy

In this case, the broker only guarantees the VM CPU resource requirement and over-allocate the RAM.

$$\sum v_i^{\text{cpu}} \leq S_n^{\text{cpu}}, \forall v_i \in S_n. \tag{4.6}$$

4.4.3 Over-commit CPU policy

In contrast with the over-commit RAM policy, the over-commit CPU policy over-allocates CPU resources instead of RAM resources. The broker considers the RAM capacity of the server and there is no upper-bound on the CPU capacity:

$$\sum v_i^{\text{ram}} \leq \alpha \cdot S_n^{\text{ram}}, \forall v_i \in S_n. \tag{4.7}$$

Where α represents the maximum consolidation rate.

4.4.4 Optimal Over-commit CPU/RAM policy

The last policy is the optimal over-commitment policy that involves both CPU and RAM resources. This solution needs to analyze the history of each job in order to predict the job resource utilization for the near future. It is designed to keep the PM utilization always close to the upper bound for both CPU and RAM resources. However, there is a risk that it may lead the PM to be overloaded if the prediction is not accurate. Consequently, a huge number of migrations may occur and lead to an additional energy consumption and performance degradation.

4.5 Experimental setup

4.5.1 Trace-driven simulator

To evaluate the proposed algorithms in the PIKA framework under different scenarios, we built a novel trace-driven simulator described in Section 3.4. The simulator is using a single energy-aware data center. It allows to simulate different resource allocation and scheduling policies.

4.5.2 Real-world traces

Job workload trace: we use the trace described previously in Section 4.1.2 and use it for all the simulations in this scenario. In this trace, each job consists of job id, time stamp, initial VM resource requirements, the instantaneous CPU and RAM utilization. To eliminate the noise of CPU and RAM utilization for each job (due to resource utilization variability), the CPU and RAM utilization is averaged over the interval T (T = 1 slot = 1 hour).

Table 4.1 – Job characteristics (Hour)

Type	Number of jobs	Execution time	Slack time
Web	150	24	< 1
Batch	600	6	12

As shown in Table 4.1, we extract a non-holiday week. It contains 750 jobs per day including 150 web jobs and 600 batch jobs. The slack time of a web job is defined as less than a slot. The slack time of batch job is defined as double time than its execution time. Half of jobs are submitted at anytime before noon (*0h - 12h*), and the other half are submitted from noon to night (*12h - 24h*).

Solar energy traces: to build the renewable energy workload in the simulator, we use the database provided by the University of Nantes (France) as described in Section 3.3. This database records the solar power data every five minutes, and we have averaged these values per time slot (i.e. per hour, so 24 values per day). We choose the trace from the same week as the trace we used for the jobs.

4.6 Evaluation

This section describes our evaluation. First, we compare the performance of the different resource over-commitment policies. Then we present the energy model based on our real power measurements. Finally, we provide simulation-based results comparing PIKA's scheduling with a baseline solution.

4.6.1 The performance of resource over-commitment policies

We implement the aforementioned resource over-commitment policies on two PMs. PM-1 consists of 12 cores and 48GB RAM (2933 MHz each core). PM-2 consists of 16 cores and 192 GB RAM (2933 MHz each core). We submit the same number of jobs for the two PMs. We combine these resource over-commitment policies with FFD algorithm as mentioned in Section 4.3.3.

Tables 4.2 and 4.3 show the number of hosted VMs for different resource over-commitment (OC) compared to the solution combining both over-commitment techniques (OPT). On

Table 4.2 – Number of VMs on PM1

Policy	Mean	St.dev.
NON.OC	6.356	2.154
RAM.OC	6.724	2.651
CPU.OC	27.035	2.744
OPT	60.994	9.775

Table 4.3 – Number of VMs on PM2

Policy	Mean	St.dev.
NON.OC	8.141	5.337
RAM.OC	11.451	2.656
CPU.OC	44.893	4.515
OPT	100.441	5.904

PM1, the first two policies (NO OC and RAM OC) have nearly identical performance (i.e. consolidation factor here). CPU OC hosts more than 3 times more VMs than both NO OC and RAM OC policies. On PM2, CPU OC also offers 400% better performance than both NO OC and RAM OC. This can be explained by an analysis of the trace: for the vast majority of VMs, the total CPU resource requirements of VMs reach the PM's CPU upper bound before the RAM reaches its upper bound. Therefore, the result of CPU OC is better than the NO OC and RAM OC policies. However, the OPT outperforms CPU OC by up to 50%. Recall that, OPT dynamically adjusts the over-commitment threshold under assumption with an high accurate VM utilization predictor.

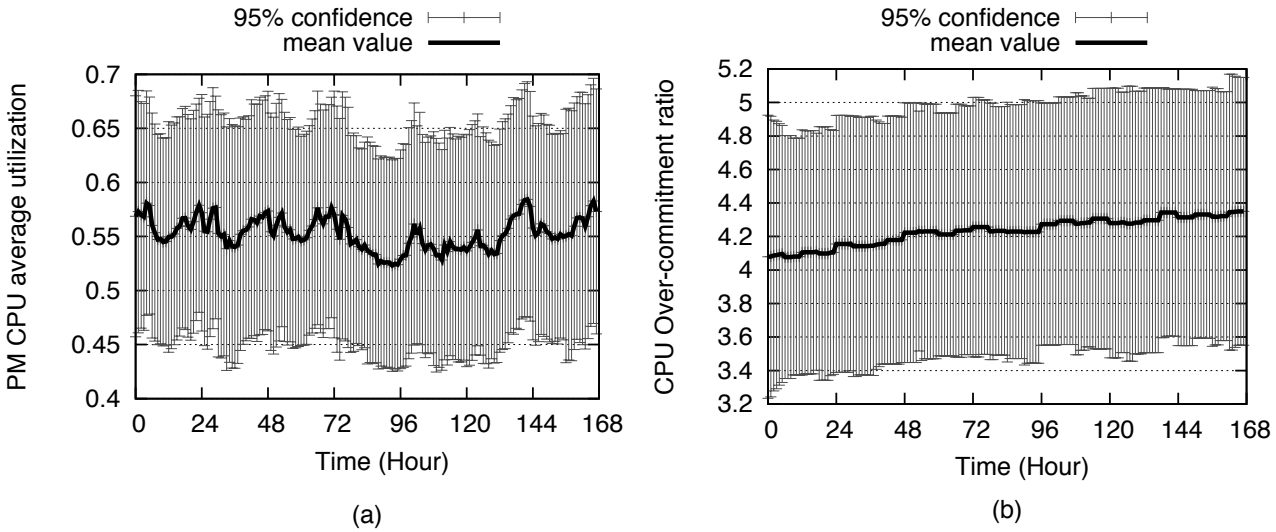


Figure 4.5 – (a) CPU real-time utilization; (b) total VM allocation ratio for the CPU OC policy

Figure 4.5 (a) shows the actual average CPU utilization of 54 servers by using CPU OC policy over 1 week (168 hours). The 54 servers average utilization has increased by 30% than non-OC policy solution (the server average utilization of original trace varies between 15%-25%). In Figure 4.5 (b) CPU OC policy over-commits 4.1X CPU resources than the PM original CPU capacity.

Similarly, Figure 4.6 shows the RAM over-commit ratio and the mean value of real PM RAM load of CPU OC policy. The actual average RAM utilization of the 54 servers is between 34% and 44%. CPU OC policy does not over allocate the RAM resources due to the average of actual RAM load is near 95%.

The above Figures 4.5 and 4.6 illustrate the fact that the CPU OC is offering a reasonable performance without a complete knowledge of future and rarely leads a PM to be overloaded. For this reason, we chose to implement CPU OC into PIKA. Consequently, in the rest of the manuscript, over-commitment will refer to the CPU OC policy.

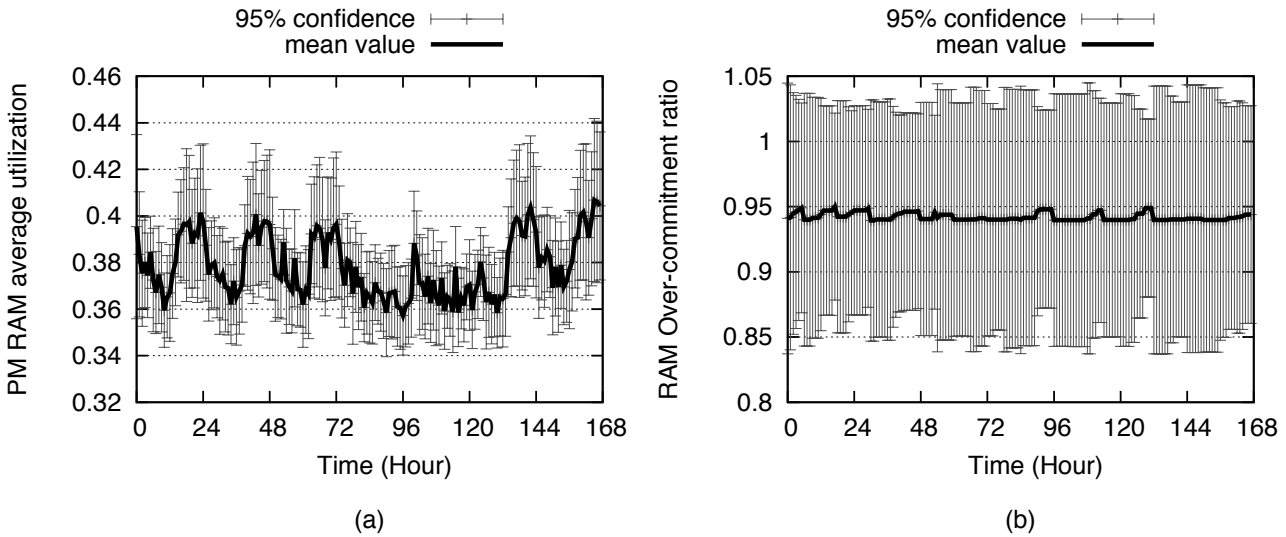


Figure 4.6 – (a) RAM real-time utilization; (b) total RAM allocation ratio for the CPU OC policy

4.6.2 Energy model

In order to simulate the energy consumption of various scheduling policies, we need to provide energy models for:

— physical machines (depending on their workload);
— VMs' basic operations (i.e. creation and migration).

Physical machine model

The variation in energy consumption of a PM mainly depends on CPU utilization [OAaL14]. We experiment multiple tests on Taurus nodes at the Lyon site of Grid'5000, a large-scale and versatile testbed for experiment-driven research [Bal+13]. Each node has 12 cores, so each core consumes 8.3% of the overall CPU utilization. We use the *stress benchmark* to vary CPU utilization in order to estimate the server's energy consumption according to its load. We activate one more core each time and keep the core at 100% utilization during 300 seconds. The test begins with 0 cores (*idle state*).

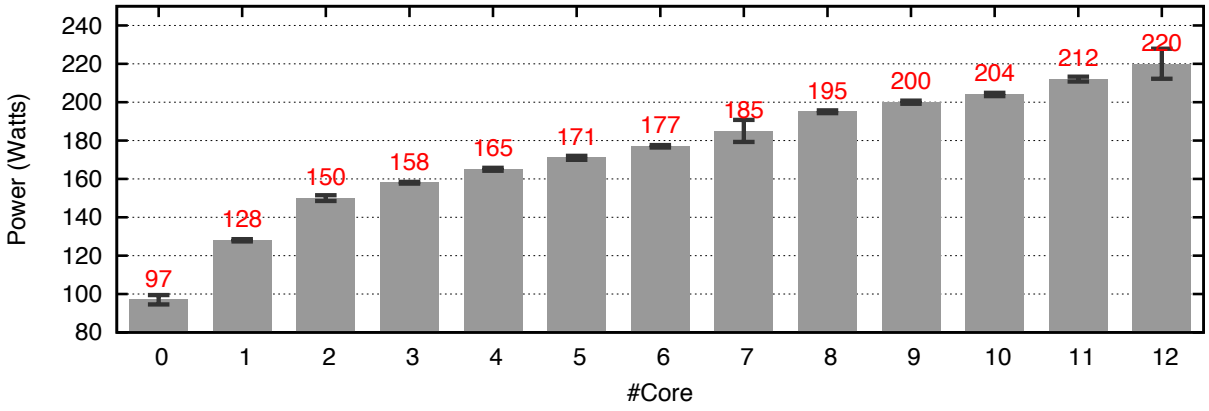


Figure 4.7 – Energy consumption of a Taurus node with different number of activated cores in Watts on Grid'5000 Lyon site. It owns 12 cores, each running at 2,933 MHz.

Figure 4.7 presents the energy consumption with different numbers of active cores. Note that an idle PM (0% CPU utilization) consumes about half of fully charged PM energy consumption (100% CPU charge). Similar observations can be found in literature [QKMM13]. We model the PM energy consumption as successive steps. The total CPU resource is divided into 12 intervals (i.e. number of cores). The number of cores transforms on CPU utilization (i.e. 0 core = 0%, 1 core = 8.33%, 2 cores = 16.66% and so on up to 12 cores = 100% CPU utilization). We create a sequence of numbers corresponding to the cores. The real PM utilization is easily falling into an interval of two consecutive numbers *a*, *b* ($a < b$) from this sequence. The PM energy consumption is the difference between a's energy consumption and b's energy consumption. Then, we construct a linear model between the two numbers *a*, *b* to calculate the PM energy consumption, this model is similar to the one used in [BB10b].

$$E_n = E^a + (E^b - E^a) \times \frac{u_n - a}{b - a}\text{, where } a < u_n < b \tag{4.8}$$

where E^a denotes the lower bound a energy consumption and E^b denotes the upper bound b energy consumption. u_n represents the PM CPU utilization and $\frac{u_n - a}{b - a}$ denotes the percentage of difference between PM utilization and the lower bound.

Moreover, there is a switching cost for PMs, as the broker needs to dynamically switch on PMs or switch them to sleep mode. We define a fix energy overhead in the later simulation when broker switches on a PM, this value has been measured experimentally on the same Taurus node.

VM energy consumption model

The VM energy consumption model consists of two modules: VM creation energy overhead and VM migration overhead. The VM creation energy overhead is defined as a fixed value in the simulator. The energy consumption of VM migration depends on the following parameters: migration duration time and the energy consumption per unit of time. The study of [Hua+11] shows that the duration of live migration depends mostly on the memory and disk used by the migrated VM. The VM migration will sightly increase the CPU utilization on destination server. As above-mentioned, the energy consumption of PM increases almost linearly with CPU utilization, we formalize the linear model for energy consumption of VM live migration as follows:

$$E_{Migration} = \frac{C_{RAM} + C_{DISK}}{B} \times E_{CPU} \tag{4.9}$$

where B denotes the bandwidth between the PMs. $C_{RAM} + C_{DISK}$ denotes the migration overhead that is the sum of the memory size and the disk size of a VM , $\frac{C_{RAM}+C_{DISK}}{B}$ denotes the duration of migration and E_{CPU} is the extra CPU energy consumption on destination PM per unit of time. These models are used in the simulator to evaluate PIKA.

4.6.3 Simulation results

PIKA is compared with a baseline algorithm: a simple FFD algorithm allocating the VMs for each time slot independently from their type (web or batch) and without considering renewable energy availability. The result of energy consumption for both baseline algorithm and PIKA are shown in Figure 4.8. The top curve presents the baseline result and the bottom corresponds to PIKA.

The energy consumption of baseline is flat. The workload scheduling is not affected by the variable renewable energy supply (the green curve). The energy consumption of PIKA

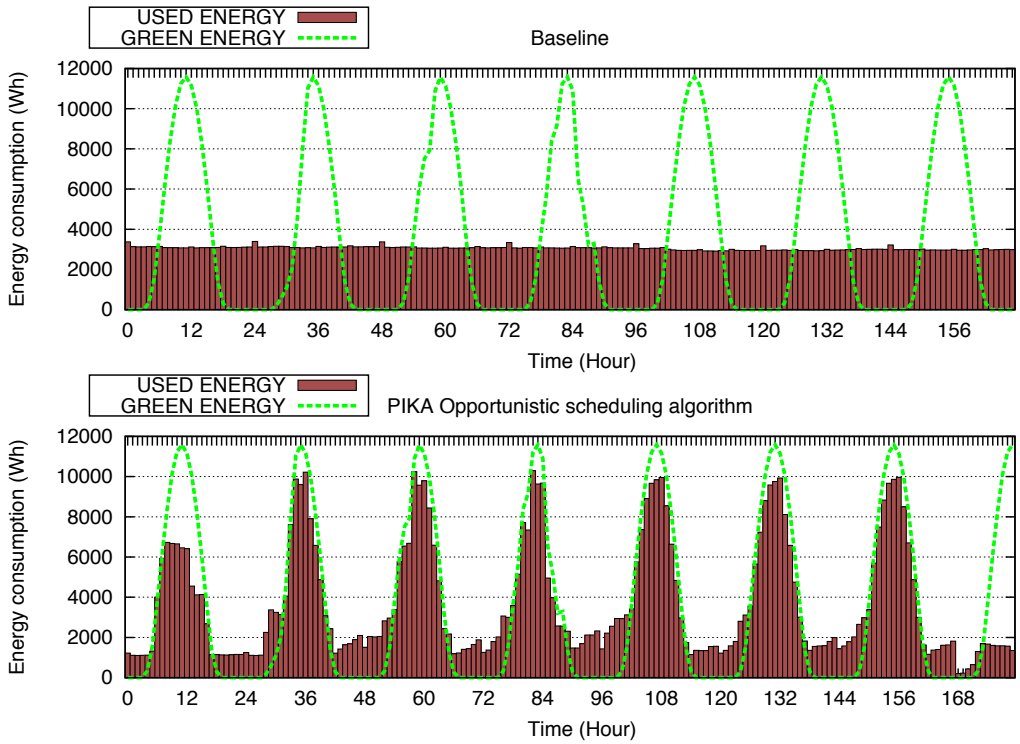


Figure 4.8 – Energy consumption : baseline vs PIKA

is following the renewable energy variations. PIKA significantly increases the renewable energy integration into the data center. While the renewable energy becomes unavailable, the broker switches off some ON PMs and only launches some essential jobs (web job, mutated batch job and a few of batch jobs on remaining ON resources). Due to this behavior, PIKA finishes all the jobs 11 hours later than the baseline. Indeed, in PIKA, the broker opportunistically schedules the batch jobs. So, some batch jobs are delayed when there is no sufficient renewable energy.

Table 4.4 shows the result of brown-, renewable- and total-energy consumption for the baseline and PIKA.

Table 4.4 – Energy saving results (kWh)

Algorithm	Total E. C.	Brown E. C.	Renewable E. C.
Baseline	513.633	259.559	254.073
PIKA	676.895	142.957	533.938
	31% ↑	44.9%↓	110.1%↑

Compared to the baseline, PIKA reduces by 44.9% brown energy consumption and increases by 110.1% the renewable energy integration. The results show that PIKA significantly reduces the brown energy consumption in comparison with the baseline, representing a typical energy-efficient algorithm (but not renewable-aware). The results also indicate that PIKA consumes 31% more energy in total. This is because PIKA performs dynamic VM consolidation to adjust the number of ON PMs and that leads to a large number of VM migrations compared with baseline (the migration in baseline is only in case of overloading PM). Moreover, PIKA needs more time to execute all the jobs as explained before. But all of this extra energy consumption comes from renewable energy supply. So, this extra energy is not used and thus wasted in the baseline case.

This work shows the opportunity created by medium-sized data centers partially powered by on-site renewable energy in order to save energy for Cloud infrastructures, such

as the one promoted by the Future Internet. However, the core of opportunistic approach mainly depends on the percentage of batch jobs and their corresponding slack time. The percentage of batch jobs represents the quantity of jobs that can be suspended and resumed later. A low percentage means that there are fewer jobs that can be delayed, leading to potential huge mismatch between the renewable energy supply and the pwoer consumption. Besides, the batch job slack time determined by their deadlines and renewable energy availability impact the performance of this approach. A small slack time cannot provide enough waiting time until the renewable energy became available.

4.7 Conclusion

Data centers partially powered by renewable energy become attractive for the new generation cloud architectures. It significantly reduces the traditional energy consumption and CO_2 footprint. The work of this section is dealing with resource allocation and opportunistic job scheduling in a small/medium mono-site data center without battery. Our proposal framework PIKA and the preliminary results outperform the classical energy-efficient VM management algorithms.

In the case of a single data center, follow-the-sun approaches are not feasible. But instead, opportunistic scheduling algorithms can make advantage of renewable energy availability to perform jobs with low priorities [LOM15]. Opportunistic policies distinguish two kinds of computing jobs: jobs requiring to run continuously (like web servers) and jobs that can be delayed and interrupted, but with a deadline constraint (such batch jobs include monthly payroll computation for example). The jobs of the second type wait for renewable energy surplus to be scheduled, thus reducing the overall consumption part of brown energy. However, such scheduling policies make use of virtual machine migrations and suspend/resume functions that have a cost in terms of energy consumption [OAaL14].

Another possible method for improving the effective utilization of intermittent and fluctuating renewable energy sources consists in using batteries to store green production surplus, and to use it during low production periods [Goi+13]. Typically for solar sources, energy can be stored during the day – if not fully consumed – and be utilized during nights when there is no production. However, batteries have an inherent energy efficiency (their yield) that leads to energy losses. So, is it greener to use opportunistic scheduling or batteries? We will illustrate the performance of these two approaches in the next chapter.

5

Balancing the use of batteries and opportunistic scheduling policies

Contents

In this chapter, we discuss two approaches for maximizing the utilization of renewable energy in small and medium data centers: opportunistic scheduling (as discussed in the previous chapter) and energy storage device (ESD, i.e. on battery). We compare these two solutions in terms of renewable energy utilization and total energy consumption in order to estimate whether the losses due to the battery efficiency balances or not the losses due to migration costs incurred by opportunistic scheduling policies. We also evaluate an intermediate solution mixing both approaches. This study investigates two types of batteries (lead-acid and lithium-ion, but can be easily generalized to other types of ESD), the optimal size of photovoltaic panels, several sunlight profiles and real-world workload traces from a medium-sized data center. Like in previous chapters, we only consider on-site renewable energy production (with photovoltaic panels) and we do not sell the produced energy to other actors: only self consumption is considered.

The remainder of the chapter is organized as follows. Section 5.1 describes Energy Storage Devices (ESDs) characteristics. Section 5.2 presents the context and used models. Section 5.3 describes the job scheduling algorithms: baseline algorithm with ESD, opportunistic scheduling without ESD and opportunistic scheduling combined with ESD. Section 5.4

presents our experimental setup including an analysis of real-world workload traces and the simulation-based methodology to find the optimal solar panel dimension and battery size for a given data center and a given workload. Section 5.5 presents the results of our simulations which show the relationship between brown energy consumption and different solar panel dimension/battery size. Lastly, Section 5.6 concludes the work of this section.

5.1 Energy Storage Devices

The variable and intermittent nature of renewable energy – like solar energy – makes it difficult to manage. In order to increase the usage of renewable energy, one way consists in carefully scheduling the workload to align with the time-varying available renewable energy. An alternate solution consists in using ESDs [GFKR15] to store the renewable energy and generate electricity for later usage.

The main parameters to be considered when dealing with ESDs are:

1. Efficiency: The energy used to charge a battery is higher than what can be used at a later time.
2. Battery charging and discharging rate limit: This charging/discharging rate limit is determined by the type of battery. Typically the discharge/charge ratio is larger than 1 for most batteries.
3. Self-Discharge: There is an energy loss which is proportional to the storage duration.
4. Depth-of-Discharge (DoD): Many factors may impact the battery lifetime such as the charging/discharging cycles [Che+09; DØ09]. DoD can also impact the battery lifetime: in order to extend the battery lifetime to a reasonable time, we cannot use the full capacity of battery.

We now present the ESD that includes re-chargeable batteries technologies (Electrochemical). In this section, we consider two kinds of batteries: Lead-Acid battery (LA) and Lithium-Ion (LI) which are prevalent in current data centers. Table 5.1 shows the different constraints per battery kind.

	LA	LI
DoD	0.8	0.8
Charge rate / ESD size (%)	12.5	25
Efficiency	0.75	0.85
Self-discharge (per day)	0.3%	0.1%
Discharge rate / charge rate ratio	10	5
Price ($/kWh)	200	525

Table 5.1 – The battery characteristics (data from [Wan+12a; Che+09; DØ09])

In comparison with LA, LI battery has higher energy density, energy efficiency and lower discharge rate, but also higher cost. In the rest of the chapter, battery and ESD terms are interchangeable.

5.2 Context and assumptions

This section is focusing on maximizing the use of renewable energy in a small/medium-scale data center with on-site solar panels. This section describes the context of this work.

5.2.1 Small and medium data centers

As in the previous chapters, the considered data center comprises between 20 to 150 servers. Each server has limited resources in terms of CPU and RAM. We assume that there is no centralized data storage system in the data center: each server has its own hard disk [Bel+16]. The data center is equipped with photovoltaic (PV) panels and an ESD. It has dual brown (from regular grid) and renewable energy supplies. If the renewable energy cannot be entirely consumed by the data center, the ESD stores the surplus of renewable energy for future use. We also assume that each server has a switch connected with renewable and brown energy supplies and the ESD. Specifically, the server can only opt for using one of the three sources at the same time.

A job can be submitted to the data center at anytime and it consists of an individual Virtual Machine (VM) to execute for a given duration. A VM is considered as the basic unit of resource allocation. We assume each VM has two constraints, namely CPU and RAM, and each job has its own duration and a predefined deadline. When a job finishes, the VM is destroyed and it releases its reserved resources back to the server. As previously, the job management system assumes that time is divided into slots. The VM resource allocation operations are performed periodically at the beginning of each time slot.

5.2.2 ESD model

As shown in Figure 5.1, photovoltaic (PV) panels turn solar energy into electrical energy which can be directly supplied for the data center or collected by the ESD. The ESD is composed of rechargeable batteries which first collect and store energy (generated from solar energy only), and then power the data center when scheduled.

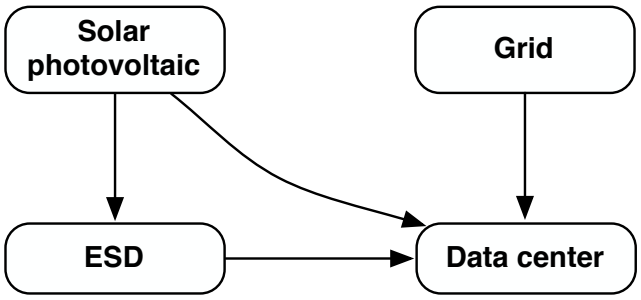


Figure 5.1 – Energy sources of the data center

The capacity of the ESD is finite. Herein, we use parameter C to express the maximum capacity of an ESD. At a given time t, $C_{available}(t)$ represents the energy that has been collected and is stored by the ESD. In order to keep a longer battery lifetime, we take into account the DoD constraint [DØ09], which stipulates that the remaining energy stored in an ESD has to be larger than the DoD threshold. So, in other terms, the available stored energy is smaller than a higher bound $\eta\ C$ ($0 < \eta < 1$, e.g, $\eta = 0.8$). By considering the DoD constraint, one can see that $C_{available}(t)$ never reaches C. Formally, we have $0 \leq C_{available}(t) \leq \eta C$.

An ESD has two significant functionalities: charging (collects energy from solar panels) and discharging (powers the data center). In our system, we consider that charging and discharging are two independent procedures. It implies ESD is never under charging and discharging states simultaneously. The charging rate has an upper bound λ depending on the ESD type and capacity. During a given time period $[t_i, t_j]$ ($t_j > t_i$), if we suppose the available green energy (supplied by PV cells) is $E(t_i, t_j)$, we use formula 5.1 to compute the amount of energy $E_{in}(t_i, t_j)$ that can be collected by an ESD.

$$E_{\text{in}}(t_i, t_j) = \min(E(t_i, t_j), \lambda(t_j - t_i), C_{\text{available}}(t_i)) \times \sigma \tag{5.1}$$

Parameter σ is constant and expresses the energy efficiency of the battery's charging procedure. The discharging rate also has an upper bound denoted μ. During a consecutive time period $[t_i, t_j]$, we use formula 5.2 to compute the amount of energy $E_{out}(t_i, t_j)$ that can be provided by the ESD. Parameter $E_{\text{self-discharge}}(t_j - t_i)$ represents the energy loss due to the self-discharging of batteries.

$$E_{\text{out}}(t_i, t_j) = \min(\mu(t_j - t_i), \eta\, C - C_{\text{available}}) - E_{\text{self-discharge}}(t_j - t_i) \tag{5.2}$$

In this scenario, we consider only solar energy as renewable energy source. Due to the variable and intermittent nature of solar energy, an energy production prediction is performed when a job scheduling decision has to be taken. It only predicts the solar energy for the following time slot (1 hour), so that such short-time prediction may have a high accuracy [Goi+13]. To simplify the problem, here we assume that there is no prediction error in our validation methodology.

5.3 VM scheduling

5.3.1 Baseline algorithm

Now we describe the baseline algorithm which will be used as a comparison reference. Whether the renewable energy is sufficient for the workload energy or not, we expect to minimize the total energy consumption of the servers. The minimization can be done at different levels such as infrastructure or application for instance. As mentioned previously, a server in idle state consumes roughly half of its peak power. Therefore, an effective approach consists in reducing the number of powered-on physical servers through consolidation.

Given a set of VMs with different resources requirements and a set of servers with fixed capacities, we want to find the minimum number of servers needed to contain all VMs, such that the amount of VMs' resource requirements assigned to each server does not exceed its capacity. The VM placement problem can be modeled as a n-dimensional bin-packing with finite number of bins, where the different VM resource requirements can be modeled as different sizes of items and the various server's resources correspond to different bin sizes. In this scenario, we consider CPU and RAM as the constraints for both servers and VMs. The VM placement problem then becomes a 2D bin-packing problem which is an NP-hard problem. As described in Section 4.3, to solve this problem, we adopt the First Fit Decreasing (FFD) heuristic algorithm.

The regular FFD scheduling algorithm usually considers VM resource requirements as the resources' constraints for its placement. However, Cloud jobs typically have resource utilization levels well below their resource requirements on average over time [LOM15]. The resource over-commitment technique increases the server utilization by considering lower bounds than the actual user requirements for allocating resources to VMs and thus, putting more VMs on a single server. As a consequence, when this lower bound is reached by each VM running on the server, some VMs have to be suspended or migrated in order to free resources. As resource over-commitment is widespread in Cloud infrastructures [Zha+12; Dab+15a], we combine FFD and CPU over-commit in order to increase server resource usage and reduce the number of powered-on servers. However, as the over-commitment configuration can greatly increase both CPU and RAM utilization, it can lead to overload the server. Consequently, we will need to migrate the VMs from the over-loaded servers to others thus

incurring an extra energy consumption and performance degradation. Hence, for these reasons, the jobs need to be provisioned for their peak draw by analyzing the history of jobs behaviors and seeking a safety over-commitment configuration.

We take into account the VM creation and VM live migration energy overhead. Unlike the VM creation, the energy consumption of VM migration depends on the VM disk size and the number of dirty pages in RAM that impacts the migration time.

The considered baseline algorithm implements both FFD and over-commit resources techniques, as our opportunistic algorithm (PIKA) does. At any time, the jobs are submitted and the broker directly places them on the servers. The baseline consumes the solar energy when it is available. The battery is charged when a surplus solar energy appears. Otherwise, the workload first discharges the battery and then uses the brown energy. Note that, there is no opportunistic job scheduling mechanism in the baseline algorithm. In the rest of this chapter, opportunistic scheduling and PIKA terms are interchangeable.

5.3.2 Opportunistic job scheduling

For evaluating the opportunistic approach, we use PIKA, our framework described and evaluated in Chapter 4. As time in our system is discrete, the optimization operations are performed periodically at each slot. According to the job characteristics, opportunistic scheduling approaches classify the jobs into two types called here web jobs and batch jobs. The web job is defined as non-uninterruptible job. It has the highest priority on scheduling. When both types of jobs arrive, the broker pushes them respectively to the web job pool and then to the web queue, the batch job to the batch pool. We adopt the FFD algorithm to place the web jobs, all the jobs in web queue are immediately placed on the servers which have sufficient resources. Unlike the web jobs, batch jobs can be suspended with a slack time that may increase the potential chance to exploit the renewable energy. When its slack time reaches 0, the batch job is promoted as a web job. After all the web jobs have been placed, the broker seeks among the running servers which meet the batch jobs resource requirement. The web job placement and batch job placement are independent algorithms.

Recall that, opportunistic scheduling targets two problems: 1) when workload energy consumption is higher than the solar energy supply, it runs partially the workload: it suspends the batch jobs which have a non-null slack time and performs VM consolidation in order to switch-off more servers. This is to reduce the brown energy consumption. 2) Otherwise, it runs the entire workload and the batch jobs which were delayed before. This is to maximize the solar energy usage.

Due to the ESD efficiency, there is an energy consumption during battery charging. In contrary, opportunistic scheduling can reduce the effect due to battery efficiency by delaying the batch jobs to consume the solar energy directly instead of storing it in the battery. However, the more batch jobs are delayed, the more numerous VM migrations will be due to consolidation.

5.3.3 Battery charge/discharge model

Figure 5.2 displays two curves: the purple curve $w(t)$ denotes the workload energy consumption and the green curve $g(t)$ denotes the solar power. We observe in Figure 5.2 that for areas a_1, a_2 and b_1, b_2, the workload energy consumption is higher than the solar energy supply.

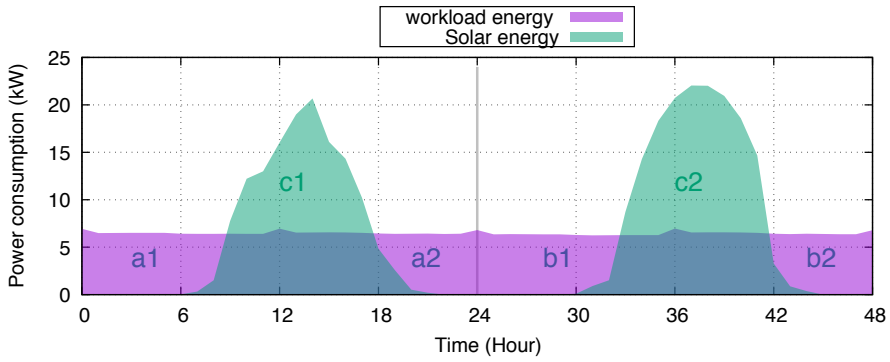


Figure 5.2 – Workload energy consumption and solar energy production

If there is no ESD, the total energy consumption from the grid can be expressed as:

$$E_{\text{brown}} = \sum_{t=0}^{t_1}(w(t) - g(t)), \forall(w(t) > g(t)), t \in T \tag{5.3}$$

When the workload energy demand is less than the solar energy, the amount of surplus solar energy is defined as:

$$E_{\text{surplus}} = \sum_{t=t_1}^{t_2}(g(t) - w(t)), \forall(w(t) < g(t)), t \in T \tag{5.4}$$

For day 1 on Figure 5.2, $E_{brown} = a_1 + a_2$ and $E_{surplus} = c_1$. The battery has to be charged when $w(t) < g(t)$. When the solar energy is not sufficient to supply for the current workload energy, we first discharge the battery. Once the battery runs out, the servers then consume the brown energy from the grid.

In the rest of this chapter, we seek for the relationship between c_1 and $(a_2 + b_1)$ in different cases. Ideally, if c_1 is much larger than $(a_2 + b_1)$, the amount of energy produced by solar panels is sufficient to offset the whole workload energy consumption that takes into account an ideal ESD (it can store all the surplus solar energy). In a real case, the battery often has limited size and solar energy is not sufficient to compensate the workload energy consumption ($c_1 < (a_2 + b_1)$).

If we assume the ESD is ideal for both solutions, then in baseline, the energy loss is mainly caused by the battery efficiency. The needed solar energy can be formulated as:

$$c_1 \times \sigma > (a_2 + b_1) \tag{5.5}$$

where σ denotes the battery energy efficiency.

5.4 Experimentation conditions

We use our trace-driven simulator (described in Section 3.4) to compare different resource allocation and scheduling policies and estimate their energy consumption using a power model based on real measurements. The simulator integrates the data center and ESD models described in Section 5.2. We use it to evaluate the impact on energy consumption with different configurations and ESD technologies. This section presents the experimentation conditions used in the simulator.

5.4.1 Workload trace

For all the simulations, we use the same real-world trace as in previous chapter from a medium-scale private Cloud data center provided by Easyvirt. The original trace was collected from 26th of March 2014 to 5th of July 2014. We extracted a non-holiday week: the data consists of 787 web jobs and 3148 batch jobs. It precises each job's initial VM resource requirement, the instantaneous CPU and RAM utilization. Similarly to validation conditions of Chapter 4, in our scenario, the CPU and RAM utilization is averaged over 1 hour in order to eliminate the noise. Each web job takes roughly 12 hours on average, and each batch job takes about 6 hours with a deadline equal to 12 hours (it has to be executed within the 12 hours following its submission).

5.4.2 Solar energy trace

As in Chapter 4, for solar energy production, we use real traces collected at the University of Nantes (described in Section 3.3.2). As shown in Figure 5.3, we choose the trace of a random week (22-28 June 2015) which is mostly sunny.

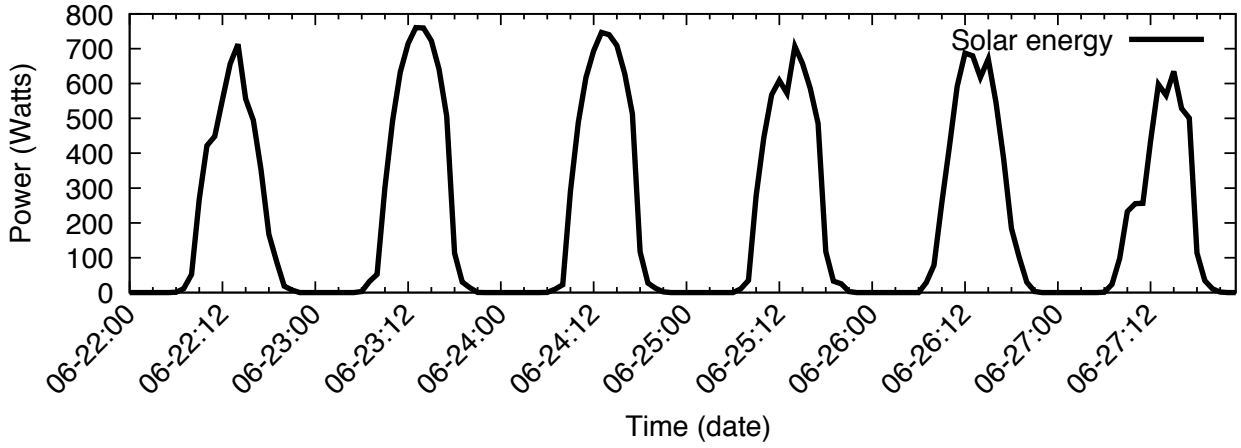


Figure 5.3 – Solar energy production with solar panels of 5.52 m^2

5.5 Results

In this section, we compare the opportunistic scheduling approach and the battery approach for maximizing solar energy utilization in a data center. First, we determine the optimal solar panel and the optimal battery size depending on the approach. Then we compare both approaches under various conditions, and we combine them.

5.5.1 Find the optimal solar panel dimension

First, we assume that the battery size and the charging/discharging rates of both approaches are infinite in this early experiment to determine the optimal solar panel dimension for the given workload. We seek for an ideal solar panel size that can supply the entire workload energy consumption. We first find a solution for this problem in the baseline case. Since the workload energy consumption can be estimated statistically, the area of solar panels can be trivially determined via calculation.

In this scenario, we assume an infinite battery size. So, it enables to store all the surplus solar energy in the region c_1 and c_2 as shown in Figure 5.2. Due to the battery efficiency

and limited charge/discharge rates, the amount of surplus solar energy has to be strictly greater than the required workload energy in order to compensate for these intrinsic ESD losses. Recall that, we are looking for c_1 which satisfies $c_1 \times \sigma \geq a_2 + b_1$ where σ indicates the battery efficiency. This formula describes the energy stored during the day which furnishes the workload energy consumption until the solar energy becomes directly available again.

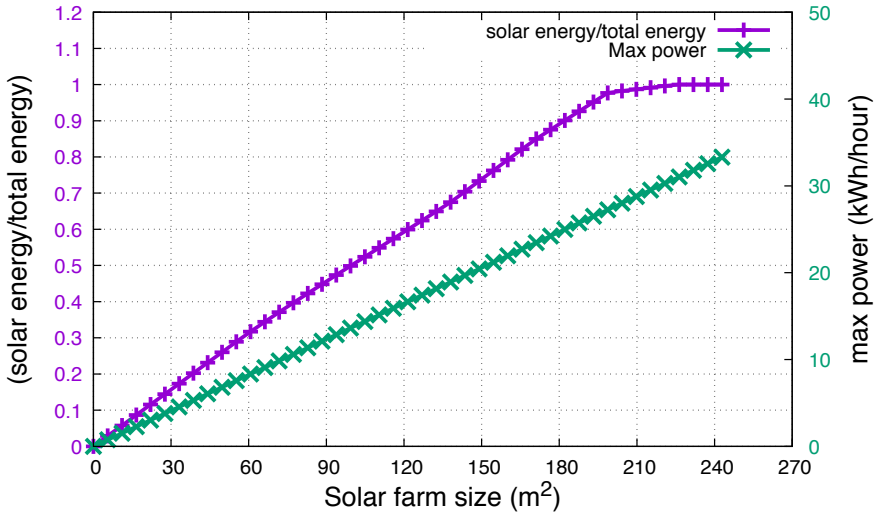


Figure 5.4 – Workload energy/solar energy supply ratio and maximum solar energy that can be generated per time unit

Figure 5.4 illustrates the workload energy/solar energy supply ratio. As the size of solar panels increase, the brown energy consumption of the entire workload decreases. When the solar panel area increases to 226.32 m^2 (about 15×15 m^2), the workload's brown energy consumption reaches 0, under the assumption of a battery with infinite size and infinite charging/discharging rate. We later determine the optimal battery size.

5.5.2 Find optimal battery size in ideal case

Now, we assume that the size of solar panels is ideal, so equal to 226.32 m^2. Thus, it can provide sufficient energy for the workload needs. Figure 5.5 shows the brown energy consumption of baseline algorithm and of opportunistic scheduling (PIKA) with the same type of battery (LI battery) depending on the battery size (purple and green curves).

We also indicate the corresponding volume taken by the batteries depending on their capacity for both LI and LA types (blue and yellow curves). The brown energy of both solutions decreases when the battery size increases. We can see that opportunistic scheduling always requires a smaller battery size than baseline, as expected. We can also see that the workload does not consume anymore brown energy when battery size is larger than 90 kWh (600 L for a LI ESD) for PIKA and 140 kWh (950 L for a LI ESD) for baseline. So, using opportunistic scheduling decreases the ideal battery capacity and its corresponding volume by 36% (for LI type).

With opportunistic scheduling, the batch jobs are delayed in order to be executed when solar energy is available. This has two effects: 1) a part of solar energy is directly consumed instead of storing it on the battery; 2) despite the batch jobs delayed run, the surplus solar energy is reduced. This is why it can use a smaller battery size than baseline to reach the same brown energy consumption. Figure 5.5 also shows that LA volume is larger than LI. Using the price values provided in Table 5.1, for a 90 kWh battery, for LI, it represents 600 L and 47,250 $, while for LA, it represents 1,150 L and 18,000 $. Both price and volume can

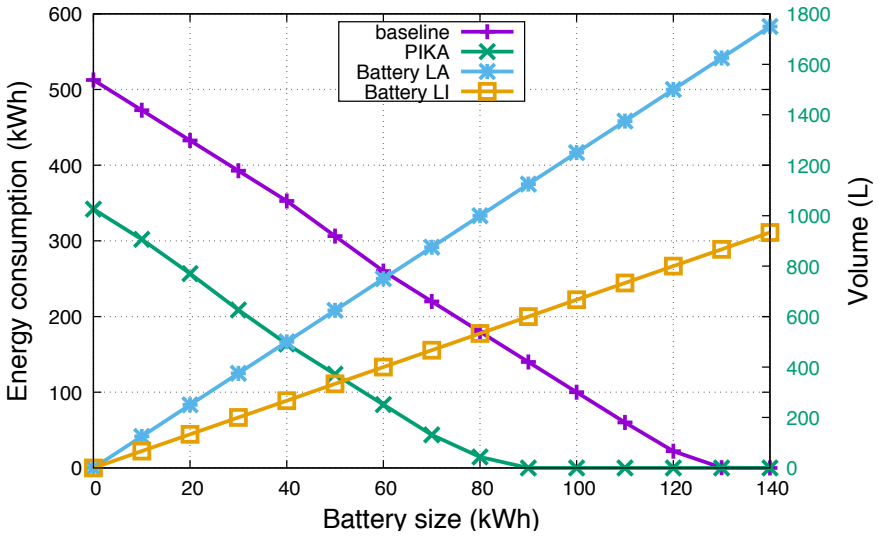


Figure 5.5 – First two curves (purple and green): brown energy consumption with an LI ESD. Last two curves (blue and yellow): corresponding ESD volume.

impact the adoption of such an approach for logistics and financial reasons. While LI and LA batteries exhibit different characteristics, their loading and discharging schemes behave similarly. In the remaining of this chapter, for clarity's sake, we use LI batteries without loss of generality.

5.5.3 Opportunistic vs. baseline when solar energy is not sufficient for the entire workload consumption

In this scenario and for all the experiments presented below, we assume that there is not enough solar energy to fulfill the entire workload needs. We consider the solar panel dimension is 160 m^2 which is not able to provide sufficient solar energy to compensate the entire workload energy consumption. We compare the brown energy consumption with variable battery sizes for the both solutions, opportunistic and baseline.

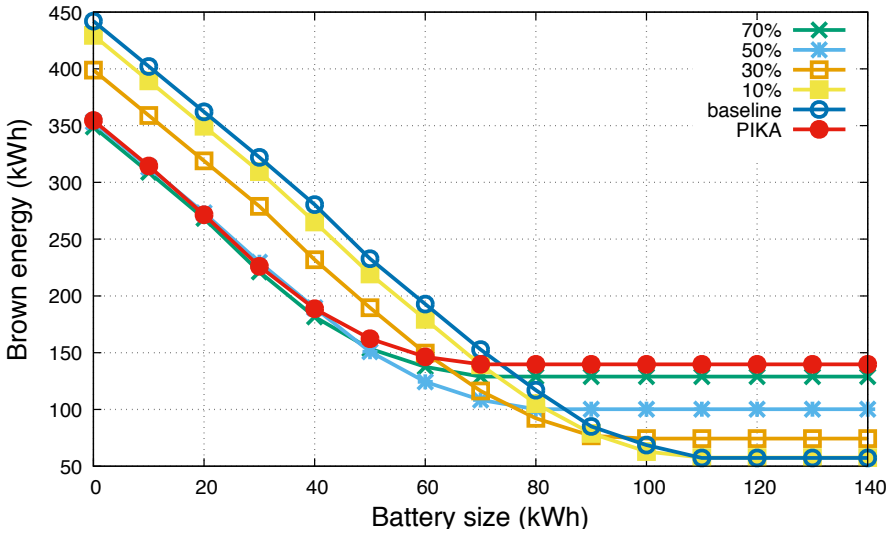


Figure 5.6 – Brown energy consumption with varying battery size

Figure 5.6 demonstrates the different battery sizes with multiple configuration for the opportunistic and baseline algorithms. The different configurations of the opportunistic

scheduling are defined as delaying 30%, 50%, 70%, 90%, 100% batch jobs (instead of directly executing them when solar energy is available, and thus storing it as normally done by PIKA). This represents possible trade-offs between the opportunistic scheduling approach and the ESD-based approach. We can see that both opportunistic and baseline approaches reduce brown energy consumption when the battery size becomes large. For a given battery size which is inferior to 73 kWh, the opportunistic approach always consumes less brown energy in comparison with baseline. After this point, the brown energy consumption of opportunistic approach does not decrease any more when battery size becomes larger. In contrast, the baseline brown consumption continues to decrease while the battery size increases up to 110 kWh. Although all the other configurations of opportunistic approach can get lower brown energy with a larger battery compared with pure opportunistic scheduling, they are still higher than baseline in this particular case.

5.5.4 Solar energy losses with variable battery size

Now we examine the renewable energy losses due to battery size limit. Figure 5.7 shows the energy losses with varying battery sizes. This energy losses stem from the battery limited charging rate and size.

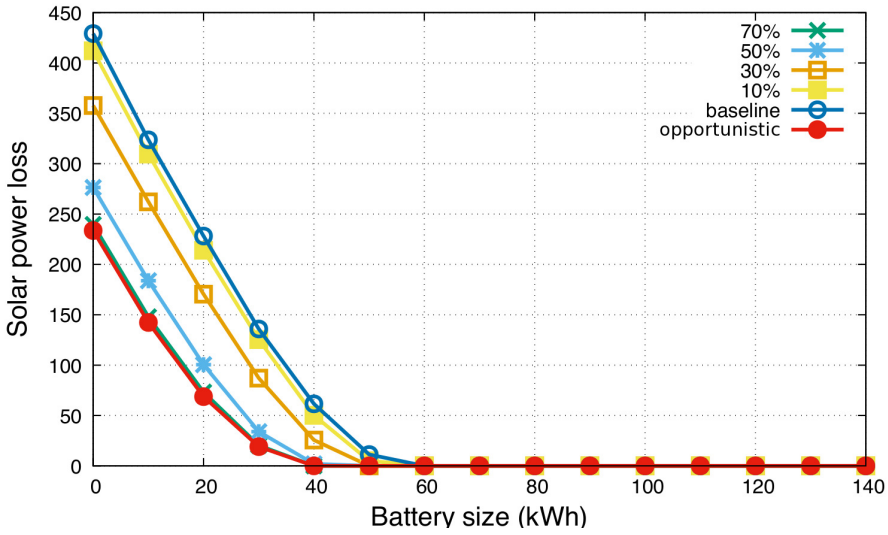


Figure 5.7 – Solar energy lost due to the limited battery size

Since the opportunistic approach delays the batch jobs until the solar energy becomes available, the workload here consumes solar energy directly and stores the rest of solar energy to the battery. Thus, PIKA can use a smaller battery while achieving the equivalent effect as baseline. Finally, when battery size is 80 kWh for opportunistic and 110 kWh for baseline, they both reach zero solar energy loss caused by the battery's limited charging rate and size.

5.5.5 Opportunistic scheduling migration vs. baseline battery loss

In order to fairly compare the opportunistic scheduling approach and the ESD-based solution, we now evaluate their respective energy losses: migration cost for opportunistic scheduling and battery efficiency for ESD solution.

Figure 5.8 shows the amount of energy lost with variable battery sizes. Here, we focus on two types of energy loss: 1) battery energy efficiency; 2) VM migration energy cost. In baseline, as the size of battery increases, the energy loss increases; this happens because of

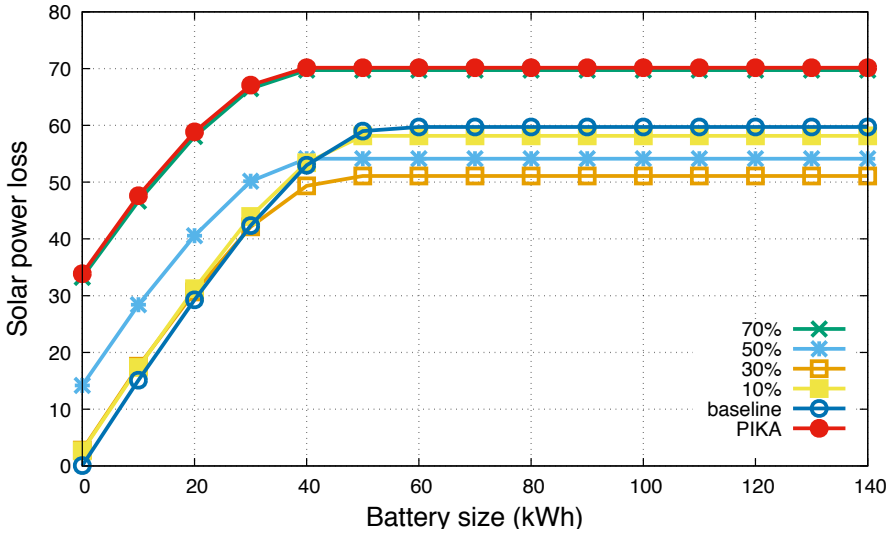


Figure 5.8 – Migration cost vs. battery efficiency loss

battery energy efficiency and of very few VM migrations due to overloaded servers (overcommit policy). Hence, the choice of battery type highly impacts the energy waste.

For the opportunistic approach, the energy loss mainly depends on: 1) migrations caused by consolidation; 2) the rest is same as baseline that is the battery efficiency. Since the solar energy was not sufficient enough for the workload needs, the opportunistic algorithm has to suspend some batch jobs and to perform consolidation in order to keep a low number of powered-on servers. And the delayed batch jobs are executed when solar energy becomes available again. The delayed workload directly consumes the solar energy and the remaining solar energy is stored in the battery. Thus the opportunistic approach stores less energy than baseline in the ESD, and thus the losses due to battery efficiency are lower with the opportunistic approach. However, the total solar energy is not sufficient for the entire workload. In this case, the opportunistic approach periodically performs VM consolidations that may lead to numerous VM migrations. This migration energy cost compensates that battery-related gain. For this reason, it can be better to partially delay the batch jobs (respectively 10, 30, 50 and 70% are delayed). In fact, when we delay less batch jobs, it leads to less migrations by consolidation, but more energy will be stored in the ESD. There is a balance for the opportunistic approach between the energy loss caused by migrations and by battery efficiency. We observe that when there are 30% batch jobs delayed, the energy loss which contains the migration energy overhead and battery efficiency, the opportunistic approach gets a lower energy loss than baseline when battery size is greater than 40 kWh.

5.5.6 FFD scheduling impact

For a given workload such as ours, the opportunistic approach has a lower energy loss in comparison with baseline when it delays 30% batch jobs. Unfortunately, in Figure 5.6, the opportunistic approach consumes more brown energy than baseline. It seems that this result is in conflict with the result in Figure 5.8. After our analysis, there is also an impact from the two consecutive FFD algorithms (for web and batch jobs) that leads the opportunistic approach to need more servers to place the same amount of jobs compared with baseline.

There is a performance degradation when we change the input list size; e.g. for any list L with a length l, $FFD(L) \leq 11/9OPT(L)+1$ as mentioned in Section 5.3.1. If we divide the list L into 2 sub-lists L_1 with a length $0 < l_1 \leq n-t$ and L_2 with length $0 < l_2 \leq t$, $FFD(L_1) \leq 11/9OPT(L_1)+1$ and $FFD(L_2) \leq OPT(L_2)+1$. The performance may be different between

FFD(L) and (FFD(L_1) + FFD(L_2)). In case of our simulation results, we observe that there is a performance degradation with the number of bins needed: FFD(L) $\ll$ (FFD(L_1) + FFD(L_2)). This explains why the opportunistic scheduling has a lower energy loss than baseline in Figure 5.8, but it consumes more brown energy than baseline in Figure 5.6 when the battery size is large enough.

5.5.7 Comparison of the approaches on a realistic scenario

Figure 5.9a illustrates a realistic scenario with 4 different cases using the same battery size and the same solar panel dimension. The first case shows the baseline without ESD, this case leads to a wastage of solar energy. The second case shows the baseline with ESD, and in this case, solar energy can be partially stored. However, a part of solar energy is still wasted due to the limited battery size and charging rate. The third case presents PIKA without battery, so solar energy is partially consumed by opportunistic scheduling, but the surplus solar energy is wasted. The fourth case exhibits PIKA with battery, and it consumes almost all the available solar energy.

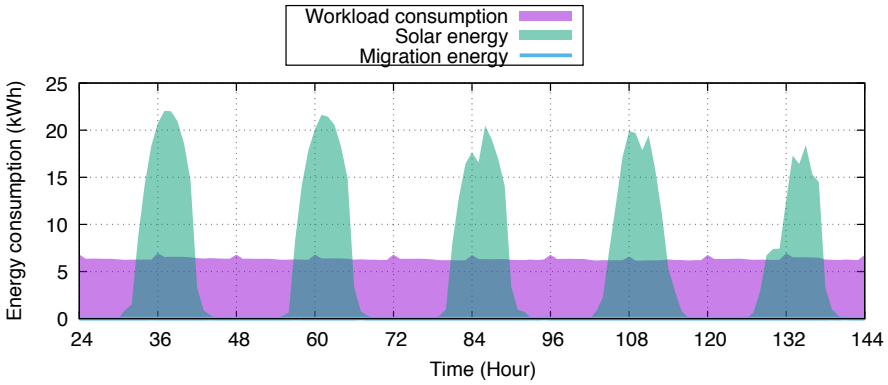
Policy	Total Energy (Wh)	Brown Energy (Wh)	Green Energy (Wh)
Baseline	768,724	442,085	326,639
Baseline + ESD	792,155	280,441	511,714
PIKA	892,458	378,569	513,889
PIKA + ESD	914,944	209,935	705,009

Table 5.2 – The energy consumption with a 160 m^2 solar farm and 40 kWh LI battery

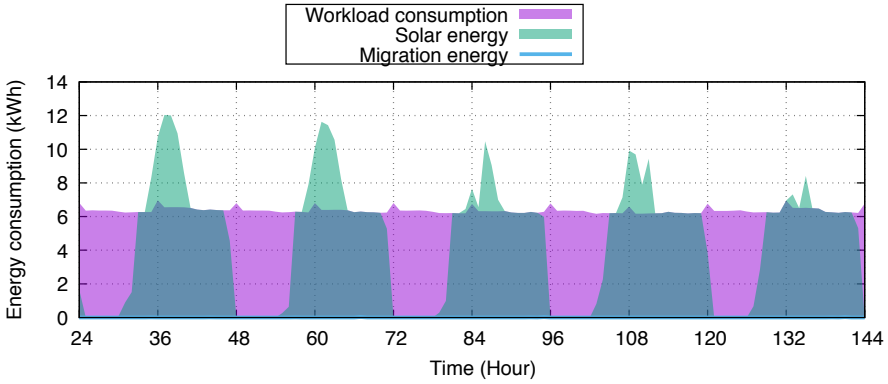
Table 5.2 summarizes the results from the experiments displayed on Figure 5.9. It shows that PIKA-ESD approach is the most energy-efficient among all approaches. Particularity, we found that the brown energy consumption of Baseline-ESD was approximately equal to original PIKA (i.e., no ESD). In fact, by aggregating from the electricity lost due to ESD efficiency and better FFD performance thus leading to higher energy efficiency at server-level (i.e., this refers to Section 5.5.6) in Baseline-ESD, the sum is equivalent to the total energy consumed by VM migrations in PIKA.

5.6 Conclusion

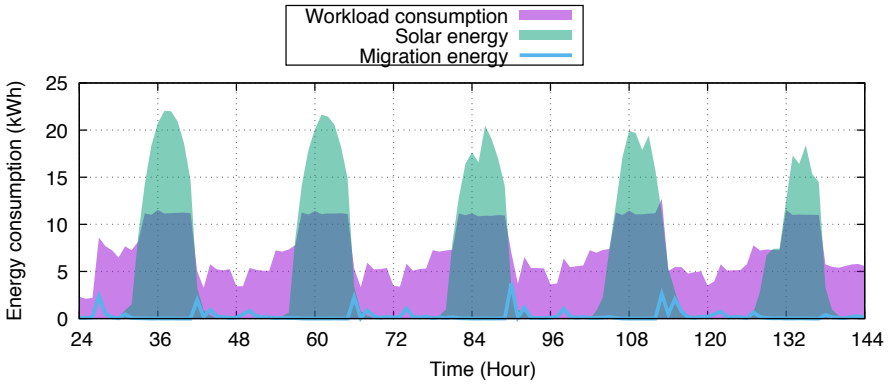
Integrating renewable energy into data centers significantly reduces the traditional energy consumption and carbon footprint of these energy-hungry infrastructures. As renewable energy is intermittent and fluctuates with time, it is usually under-utilized. In this chapter, we address the problem of improving the utilization of renewable energy for a single data center by using two approaches: opportunistic scheduling and energy storage. Our first result deals with analyzing the workload to find ideal solar panel dimension and battery size, this is used to power the entire workload without any brown energy consumption. However, in reality, either the solar panel dimension or the battery size are limited, and we still have to address the problem of matching the workload consumption and the renewable energy production. The second result shows that opportunistic scheduling can reduce the demand for battery size while the renewable energy is sufficient. The last results demonstrate that for different battery sizes and solar panel dimensions, we can find an optimal solution combining both approaches that balances the energy losses due to different causes such as battery efficiency and VM migrations due to consolidation algorithms. We believe



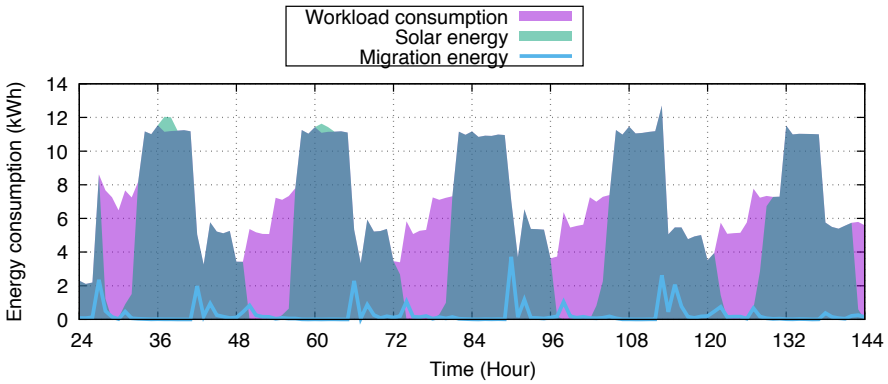
(a) Baseline without ESD



(b) Baseline with ESD



(c) PIKA without ESD



(d) PIKA with ESD

Figure 5.9 – The energy consumption with fixed solar panel dimension (160 m^2) and 40 kWh LI battery

these works provide a valuable method for right-sizing the on-site energy production infrastructure, when we know the workload characteristics.

This work can be extended by studying the pertinence of both approaches with other renewable energy sources, like wind for instance. As wind energy presents a completely different production profile compared to solar energy, we could also investigate whether the trade-off between the opportunistic scheduling and energy storage approaches which is proposed in this chapter remains the same or would be different.

6

Leveraging Renewable Energy in Edge Clouds for Data Stream Analysis in IoT

Contents

Regardless of the considered opportunistic task scheduling or ESD-based approach, the usage of renewable energy is significantly increased with non-negligible energy losses due to their energy costs (see Chapters 4 and 5). By mixing these two approaches, the quantity of wasted energy could be minimized in a single data center. In this chapter, we extend our previous work to leverage on-site renewable energy production in the different edge cloud nodes, in particular to deal with green IoT (Internet of Things).

In this chapter, we propose an analytical model for deciding whether to offload computation from the objects to the edge or to the core Cloud, depending on the renewable energy availability and the desired application QoS. In order to conduct an in-depth analysis with relevant assumptions leading to concrete contributions, we decide to focus on a particular use-case with precise requirements and a realistic modeling. Our validation use-case targets the Internet of Vehicles (IoV) that can be seen as a convergence of the mobile Internet and the IoT [Yan+14]. In particular, we focus on video streams from cameras that need to be analyzed usually for object detection and tracking. In this particular case, as it is often the case with IoT applications, a high QoS level is required. Indeed, data lose their value when they cannot be analyzed fast enough.

The rest of this chapter is organized as follows. Section 6.2 presents the system model and assumptions. Section 6.1 explains our use case. Section 6.3 gives the experimental setup

and evaluates the performance and corresponding energy consumption. A conclusion is presented in Section 6.4.

6.1 Driving Use Case

The edge typically has less computing capacity (e.g., compute servers) than the resources available in the cloud core. However, theses edge servers are closer to the edge-users and therefore, for users, the latency to edge servers is lower than the latency to the core. We consider that edge servers have dual energy supply which include traditional brown energy and renewable energy that is produced on-site and embeds a reasonably sized Energy Storage Device (ESD) to store the surplus renewable energy.

The core Cloud represents the federation of large data centers where each data center is composed of thousands of servers. Such a model of data centers [DM+13] with federation of resources and autonomic management mechanisms offers a large pool of computing resources. While the core has more powerful servers the energy costs associated to data movement present different tradeoffs that need to be investigated.

The motivation of this work is to provide a framework that can balance performance and energy cost tradeoffs for real-time data analysis of high-rate data from many sensors. A typical use-case scenario consists in cameras that can be embedded in small devices as such Google Glass, GigaSight [Sim+13] or any other devices. The camera captures frames continuously that can be seen as a high-rate data stream. Since such a video analysis, that detects interesting objects (i.e., areas of interest) from it, consumes computing power, it thus requires energy. To increase the computation performance and to reduce energy consumption on the end-device, data is often offloaded to the Cloud to be analyzed. Although data offloading to high performance servers at the Cloud can accelerate the analysis processing, the efficiency of the whole procedure is highly dependent to the network condition and to the costs associated to the network service.

In this contribution, we make the assumption that all the considered vehicles are equipped with an on-board camera and are capable of uploading the video captured by their cameras continuously to edge and core clouds. The edge/core analyzes each data stream in real time and returns the road condition to the user. The application goal is preventing traffic jam and possible traffic accidents by sharing the produced information to users in an online manner. Integrating this into next generation of vehicles with autopilot technology can help improving the road safety for the drivers (i.e., the users). As shown in Figure 6.1, an object is detected by analyzing the data stream from the first car, the resulting analysis identifies an object in the middle of the road which may be dangerous for the other vehicles behind on this road. The edge-1 immediately informs all the vehicles that are in section BC of the road. At the same time, a message is sent from edge-1 to the edge-0 in order to inform the vehicles in section AB of the road.

6.2 System model and assumptions

6.2.1 Edge and Core model

Inspired by the previous work on video stream analysis [Sim+13; Anj+16] and edge-computing [MZL16], our model involves two types of computing resources. In particular, the renewable energy in this contribution refers to solar energy as in the previous chapters of this manuscript.

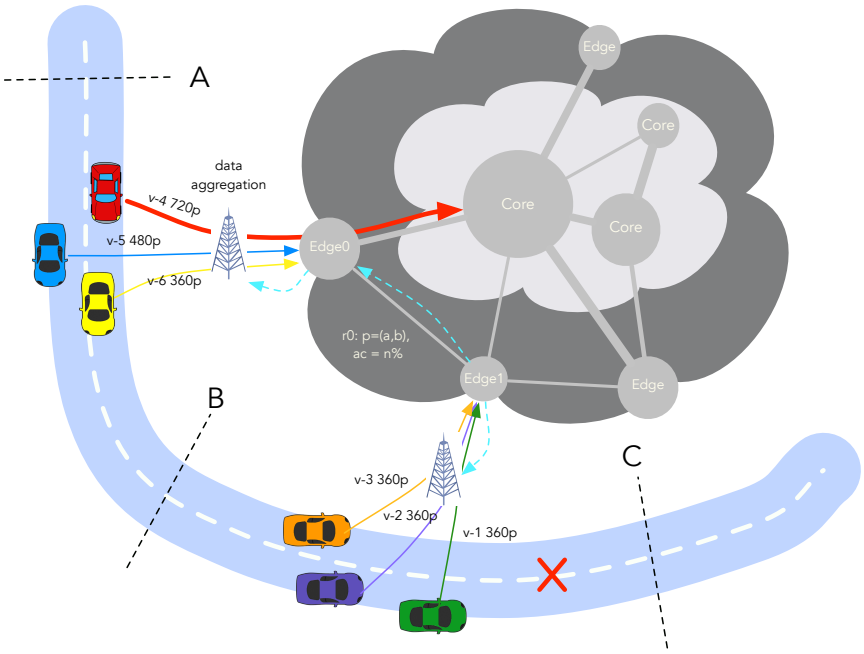


Figure 6.1 – Use case example for IoV with edge and core clouds

Computation at edge

Because a user is physically close to the edge, the servers placed at the edge enables low latency for users. The data transfers from users to edge can have a lower latency than direct transferring to the core Cloud. Conversely, the computation capacities at the edge Cloud is limited and can be seen as a small-scale data center, the considered edge comprises between 20 to 50 servers. Each server has limited physical resources in terms of CPU, RAM and ingress bandwidth. We assume that there is no centralized storage system at the edge cloud: each server has its own hard disk [Bel+16]. Once the edge cannot satisfy the computational task QoS requirement, it transfers the task to core where sufficient computing resources are available.

The edge is equipped with a number of photovoltaic (PV) panels and an ESD. It has dual brown (from regular grid) and renewable energy supplies. If the renewable energy cannot be entirely consumed by edge servers, the ESD stores the surplus of renewable energy for future use. We also assume that each server has a switch connected with renewable, brown energy supplies and the ESD. In particularly, the server can only opt for using one of the three sources at the same time. These assumptions are the same as the ones done in Chapter 5.

Computation at core

The core represents a federation of inter-connected data centers which are usually far from users. Although the servers placed at the core cloud have higher latency than edge servers, either their number or their performance (of core servers) are higher than at the edge. From the energy cost perspective, the data processing at the core is faster than data processing at the edge. However, a large volume of data needs to be transferred to core to process such that the communication cost between user-core through the Internet cannot be ignored.

A job is a request from a vehicle that requires computing resources for processing. It can be submitted to the edge and the core at anytime. Once the request is accepted, a Virtual

Machine (VM) is created on a server at the edge or core to process the analysis. A VM is considered as the basic unit of resource allocation. Each VM is created with its specific CPU and RAM requirements. When the vehicle leaves this section of road, the VM is destroyed and it releases its reserved resources back to the server.

6.2.2 Renewable energy and ESD model

Due to the variable and intermittent nature of solar energy, an energy production prediction is performed while a job scheduling decision has to be taken. We take the same assumption as in Section 4.2. The solar energy is predicted only for the following time slot (1 hour), so that such short-time prediction may have a high accuracy [Goi+13]. To simplify the problem, we assume that there is no prediction error in our validation methodology.

6.3 Experimentation

The first half of our experiments consists in measuring the power consumption and performance degradation with different application resolutions on our experimental test-bed Grid'5000. We use the same servers as mentioned in Section 4.6.2 (i.e. the used servers are Dell PowerEdge R720 from the *Taurus* cluster at Grid'5000 Lyon site. Each server is composed of two Intel Xeon E5-2630 processors (2.3GHz) each with 6 cores, 32 GB of RAM and 600 GB of disk space.) The processors support hyper-threading technology thus the total of 12 physical cores servers can provide 24 virtual CPUs. We employed KVM as the virtualization solution along with Linux on x86-based servers. The experiment results are used for building power and performance models. The network energy consumption model is defined in a similar way as it is in [Jal+16] and based on per-bit cost. These models were integrated into our simulator described in Section 3.4. In order to extrapolate to large-scale, the second half of our experiments are held using this simulator.

6.3.1 Setup

The servers are placed at both edge and core. The power consumption of the servers is related to the CPU load as defined in Section 4.6.2. On the side of the solar power production, we employ the traces detailed in Section 3.3.2. The theoretical max power of each panel is 240 Watt. Subsequently, we extract data for a whole week (22-28 June 2015) from the database which is shown in Figure 5.3 (in Section 5.4.2), the days in this week are mostly sunny.

Existing literature has addressed video analysis algorithms and tools. Haar feature-based cascade classifiers [VJ01] is a typical method for object detection which is effective and capable of achieving high detection rates. It is based on machine learning approach AdaBoost [SS99] and trains a cascade function from a large set of positive and negative images. The classifiers used in this chapter are included in the OpenCV distribution [1] 2.4.13. We trained our own Haar classifier which is used to analyze video streams for vehicles detection in this chapter. The video is encoded through H.264 codec in 3 resolutions (360p, 480p and 720p). More details are shown in Table 6.1) and we use the FFmpeg tool [Ffm] for decoding.

1. OpenCV is designed for computational efficiency and with a strong focus on real-time applications http://opencv.org

	resolution	bit rate
360p	640 x 360	514 kb/s
480p	720 x 480	706 kb/s
720p	1280 x 720	1176 kb/s

Table 6.1 – 360p, 480p and 720p represent 3 different resolutions of the same video.

6.3.2 VM size and time analysis

Due to the server limited computational capacity, allocating resources to VMs needs to be carefully done. The goal of our first experiment is to evaluate the video analysis performance and energy consumption on different VM sizes. In this experiment, we create two individual VMs on two servers from the Taurus cluster. The VM-1 is given 2 virtual CPU and 2 GB RAM, and the VM-2 is given 4 virtual CPU and 4GB RAM.

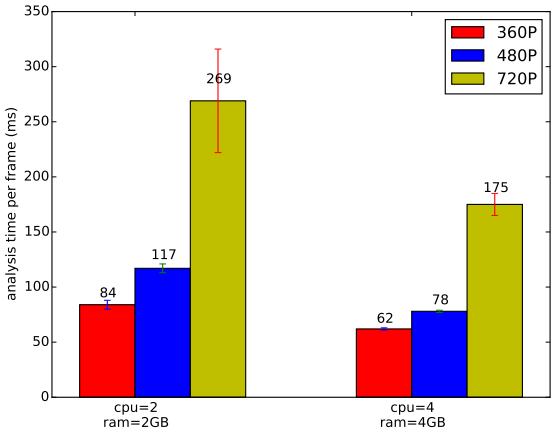


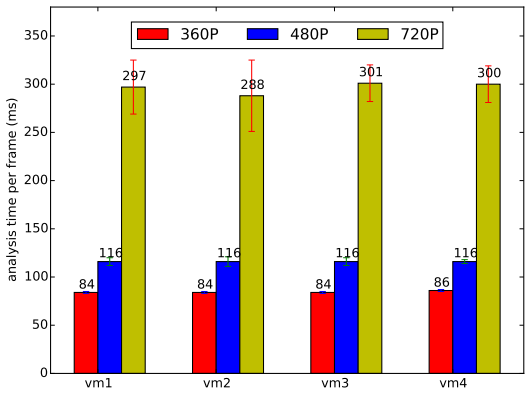
Figure 6.2 – Analysis time on different VM sizes

The analysis time per frame of VM-1 and VM-2 are shown in Figure 6.2. VM-2 is 26%, 33% and 35% faster than VM-1 for resolutions of 360p, 480p and 720p respectively. Clearly, the VM-2 benefits from more computational resources (i.e. the application makes advantage of parallel computations) and it results in a reduced analysis time.

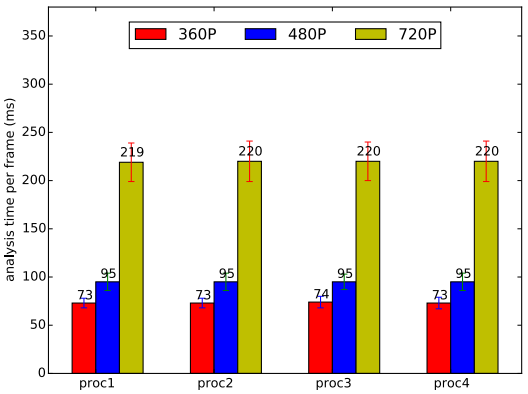
We then move on to another experiment where we vary the number of VMs. We first create VM-1∼4 on server Taurus-12, four identical VMs, and each VM has the same hardware configuration: 2 vCPU and 2 GB RAM. These VMs process only 1 data stream at a time. VM-5 is created on server Taurus-13 with 8 vCPU and 8 GB RAM. Unlike VM-1∼4, it processes 4 data streams in parallel. We conducted the experiments on analyzing the same video. The experiment iterates 10 times and each time only processes 1 video format.

The results are shown in Figure 6.3. Figure 6.3a is when 4 individual small VMs are used on Taurus-12 and each VM only processes 1 data stream. Analysis time is similar for the four VMs for each resolution format. In Figure 6.3b, it shows the processing of 4 data streams in parallel within a large size VM on Taurus-13. We observe that processing 4 streams in 1 large VM is faster than processing in 4 small size VMs. We attribute this to the fact that the KVM virtualization layer adds a penalty. In case of 4 VMs, the computational resources given to each VM from KVM is not always from the same physical cores. In other words, there is a scheduling cost if a VM is not always using at least one physical core. Moreover, as we are executing the same application four times in parallel, there might be a positive memory mutualization effect for the large VM case that does not appear in the case of separate VMs.

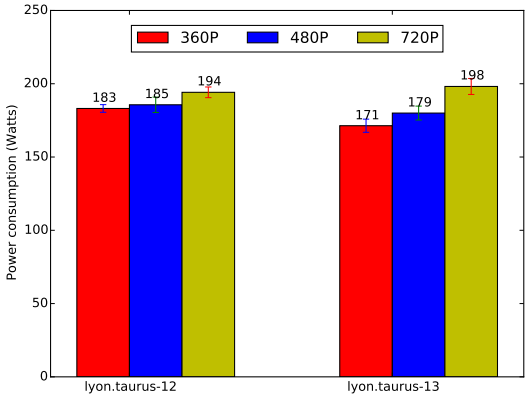
Figure 6.3c shows the power consumption of Taurus-12 with 4 small VMs processing 1



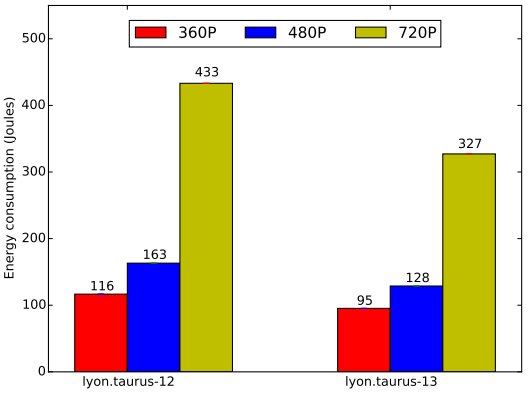
(a) Analysis time for 4 identical VMs with 1 data stream each on the same PM



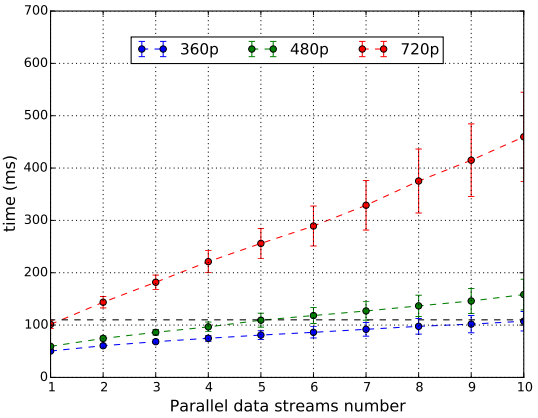
(b) Analysis time for each of the 4 data stream processes in a large VM



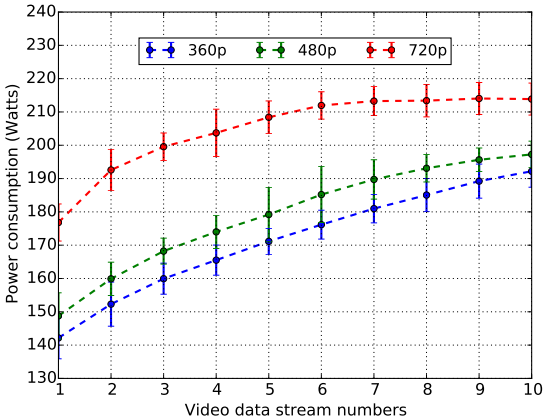
(c) Power consumption for 4 small VMs on Taurus-12 and 1 large VM on Taurus-13 for the same amount of computation



(d) Energy consumption for analyzing a 5 mn video on Taurus-12 with 4 small VMs and on Taurus-13 with 1 large VM



(e) Analysis time with parallel data streams in a large VM



(f) Power consumption with parallel data streams in a large VM

Figure 6.3 – Energy consumption and frame analysis resolution time in 360p, 480p and 720p

data stream each, and Taurus-13 with 1 large VM hosting the same 4 data streams. As shown

in Figure 6.3c, the average power consumption (in Watts) for processing 4 data streams in 1 larger VM is lower compared with 4 small VMs. For analyzing a 5 minute video, as shown in Figure 6.3d, the large VM (VM-5 on Taurus-13) with faster speed of frame analysis and lower instantaneous power consumption, consumes less energy in total.

We also observe that the processing time increases significantly when the resolution format increases. For instance, if we expect to analyze 8 frames per second (e.g., Simoens *et al.* [Sim+13] select 1 out of 24 frames for analyzing) for a relevant application precision: in order to ensure the accuracy of analysis process, this throughput should be as maximum as possible. It means that we have to analyze 1 frame in every 3 frames with a video at 25 fps (frame per second). It means that the average analysis time per frame must be smaller than 125 ms. To compute the maximum number of videos that can be analyzed in parallel, we assume that 1 VM is used for analyzing 1 format of video. We measure the analysis time on VM-5 for a video in the 3 resolution formats.

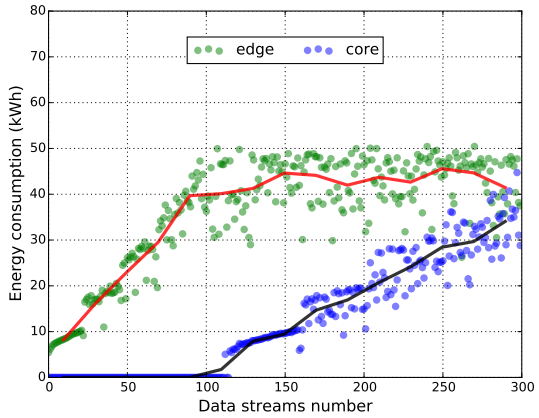
As shown in Figure 6.3e, the large VM (VM-5 on Taurus-13) supports in parallel up to 11 videos streams in resolution 360p, 4 video streams for 480p video and only 1 for 720p video. Figure 6.3f shows the respective energy consumption in the 3 resolution formats for the large VM.

6.3.3 Edge and core clouds' energy consumption

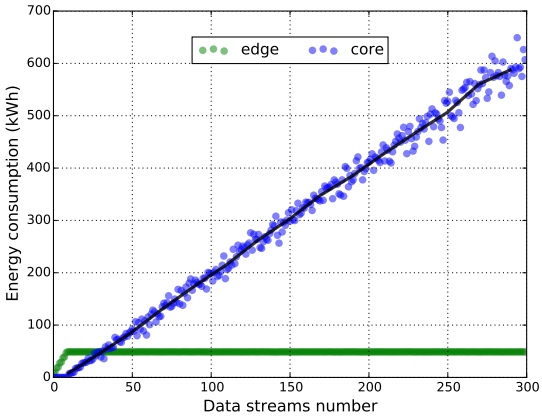
In this subsection, we evaluate the effects of offloading computation tasks at the edge for system performance of our framework and energy consumption at edge and core. We study the scalability of our framework by increasing the number of vehicles (video sources). We assume that there is no bottlenecks in the network between user-edge and edge-core. The experiments in this subsection are performed using simulations based on the real measurements done in the previous subsection.

Edge usually has less computational resources in comparison with core. In initial configuration, edge has 5 servers and dual energy consumption (self-produced renewable energy with ESD and brown) and core has 100 servers without any renewable energy source. Each edge server has 24 virtual CPU and 24 GB of RAM and the core servers are twice as powerful as edge servers. To avoid the energy consumption associated with VM placement, we assume all the VMs are same size that consists of 8 virtual CPU and 8 GB of RAM at edge. The VMs have 24 virtual CPU and 24 GB of RAM at core implying the time analysis is reduced. We only consider 360p and 720p video formats in this scenario in order to illustrate that different resolutions impact energy consumption and performance. As mentioned before (Section 6.3.2), a VM processes one format of video in the experiment thus a VM at maximum processes 1 video stream for 720p, and 10 video streams for 360p in parallel as shown in Figure 6.3e. All the requests of data analysis are processed at the edge by default. If edge does not have sufficient resources for processing, the request is transferred to core.

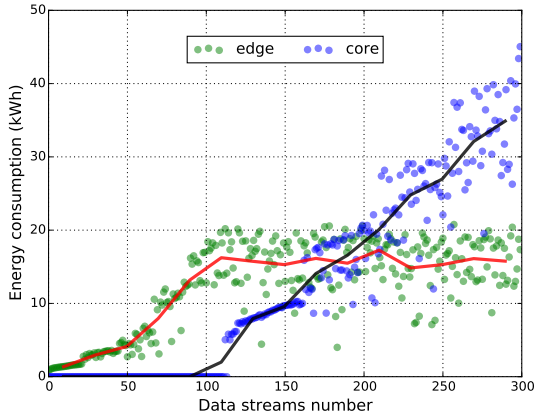
The goal of this experiment is to measure the total energy consumption at both the edge and core. We first assume that all the data streams are 360p and the renewable energy is not available at the edge (e.g., there is no solar energy production during the night). At beginning, there are few vehicles in the system. These vehicles first offload their data to edge to be processed. When increasing the number of data streams, the edge energy consumption increases by processing these data streams. As shown in Figure 6.4a, we observe that the core does not consume any energy before the edge computing resources are exhausted. Core starts to process data when the number of data streams exceeds 112 in the system. In Figure 6.4b, all the 360p videos are replaced by 720p, the edge quickly drained its resources when processing 720p videos as it consumes more computation resources than 360p videos. The core receives the first request of data analysis from the 16th vehicle. From that moment,



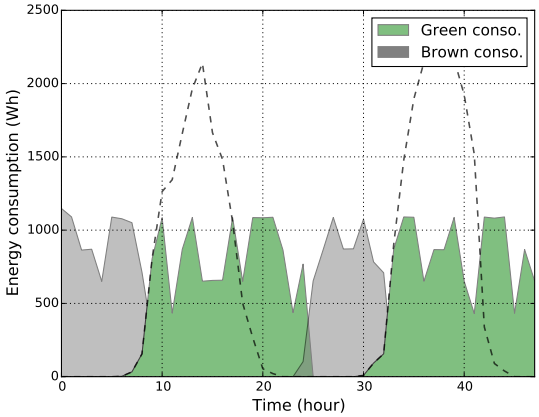
(a) Energy consumption with resolution 360p



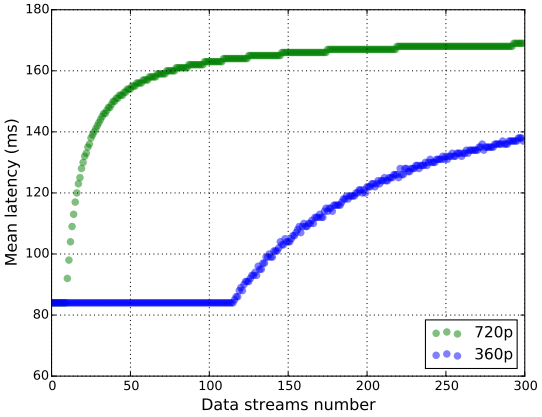
(b) Energy consumption with resolution 720p



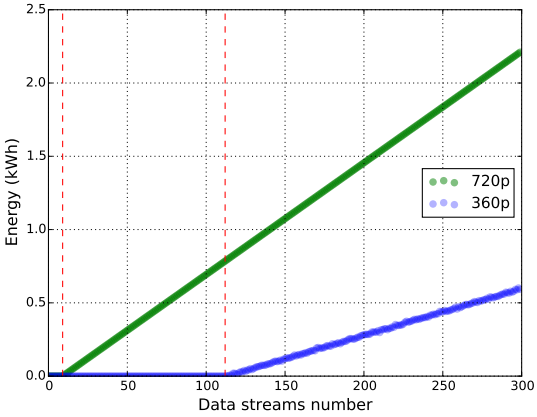
(c) Brown energy consumption with resolution 360p



(d) 2 days of energy consumption at edge



(e) Average application delay in the system



(f) Data offloading affects network energy consumption between edge and core

Figure 6.4 – The renewable energy is not available at edge in Figures (a) and (b) and is available in Figures (c), (d), (e) and (f)

all the new data arrivals are directed to core to be processed.

Once the renewable energy becomes available (i.e., we assume the area of solar panels is 17 m^2 and a 10 kWh ESD), as shown in Figure 6.4d, the edge consumes directly the renewable energy (green) instead of brown energy (gray). The surplus renewable energy produced is stored into its ESD for future usage. Edge is always prior to consume from renewable energy source and then it consumes from its ESD. It consumes the brown energy when both (direct and stored green energy) are unavailable. Figure 6.4c shows, by integrating the on-site renewable energy and ESD at edge that it reduces roughly by half the energy consumption compared with only non-renewable energy configuration.

In Figure 6.4e, we can observe that the average delay of 360p videos is significantly lower than for 720p videos. Indeed most analysis tasks are performed at edge instead of at core. Once the edge has exhausted all its resources, the new arrivals are migrated at the core. On the scale of 300 vehicles, edge is capable of processing 37.3% of 360p videos streams in the system. In contrast to 360p, processing 720p video streams consumes much more computing resources than processing 360p videos. The edge can only process 5% of data streams and all the other data streams have to move to core for processing. Despite the fact that the core possesses more powerful computational resources which might even reduce the time analysis, the latency from the network between edge and core cannot be ignored. Figure 6.4e also demonstrates that the average delay of all videos are mainly depending on the number of data streams offloading to the core. When increasing the number of data streams moving to core, the network energy consumption is also increased as shown in Figure 6.4f.

6.3.4 The detection accuracy and number of cameras

Processing analysis in higher resolution video format often outputs a result with higher detection accuracy. However, it consumes consequently larger computational resources including CPU/RAM and bandwidth for transmission. Reducing the resolution is a clear way to save computing resources and network utilization, and thus energy. Edge servers can process more videos streams in parallel without significant performance degradation. It potentially decreases network usage, thus more video streams can be processed at edge. However, scaling down the video affects the detection accuracy. As mentioned in [Sim+13], lowering the resolution of video significantly reduces the detection accuracy. We show the initial accuracy settings used in this subsection for object detection in Table 6.2.

Classes	720p	480p	360p
car	96.7%	91%	88.5%
body	97.7%	94.9%	90.7%
dog	96.1%	94.9%	90.7%
total	96.7%	92.3%	87.9%

Table 6.2 – The detection accuracy for different objects from [Sim+13]

Assuming that there is only one car in the section *AB* of road, the detection accuracy for car is equal to 96.7%, 91% , 88.5% with 720p, 480p and 360p video format respectively. Now, we assume that there are two cars in the same section, their cameras both capture with resolution 360p. When one of the two cameras detects an object on the road and another did not, one can wonder in this case, which camera should be used for the definitive result? Furthermore, we replace one camera by using 720p resolution. Suppose the two results are still different, should we always believe the result with higher resolution (720p) because of its higher detection accuracy by default?

Unfortunately, we cannot directly conclude which result of the two is more believable. Even though the 720p videos often offers a higher detection accuracy than 360p videos, this

only shows that 720p is more likely to be correct, but it is not conclusive. However, when increasing the number of cameras, we show that the correct probability of result is not only depending on the initial detection accuracy, but also related to the number of cameras in the system. Suppose there are $2n+1$ cars in the same section of road. All the cars upload video streams with the same resolution and then they output $2n+1$ results. Intuitively, if there is a result appearing at least for half of the total number of cars, we prefer to select this result as the final result. We define the **reliability** as the probability of a result that appears to exceed $n+1$ times among $2n+1$ results. In this section, we prove this final result becomes more believable when the number of cameras increase. First, each result is independent with others, so the reliability can be expressed as follows:

$$\begin{aligned}\text{reliability} &= \Pr(X \geq n+1) \\ &= \sum_{x=n+1}^{2n+1} C_{2n+1}^{x} (p)^{x} (1-p)^{2n+1-x} \\ &= \sum_{x=n+1}^{2n+1} \binom{2n+1}{x} (p)^{x} (1-p)^{2n+1-x} \\ &= \sum_{x=n+1}^{2n+1} \frac{(2n+1)!}{(2n+1-x)!x!} (p)^{x} (1-p)^{2n+1-x} \\ &\text{where } p \in (0,1)\end{aligned} \tag{6.1}$$

After simplifying the equation 6.1 , it can be expressed as:

$$\text{reliability} = \frac{1}{1+\left(\frac{1-p}{p}\right)^{2n+1}} = \frac{1}{1+\omega^{2n+1}}, \text{where } \omega = \frac{1-p}{p} \tag{6.2}$$

(For details about simplifying the equation 6.2, see Appendix A.)

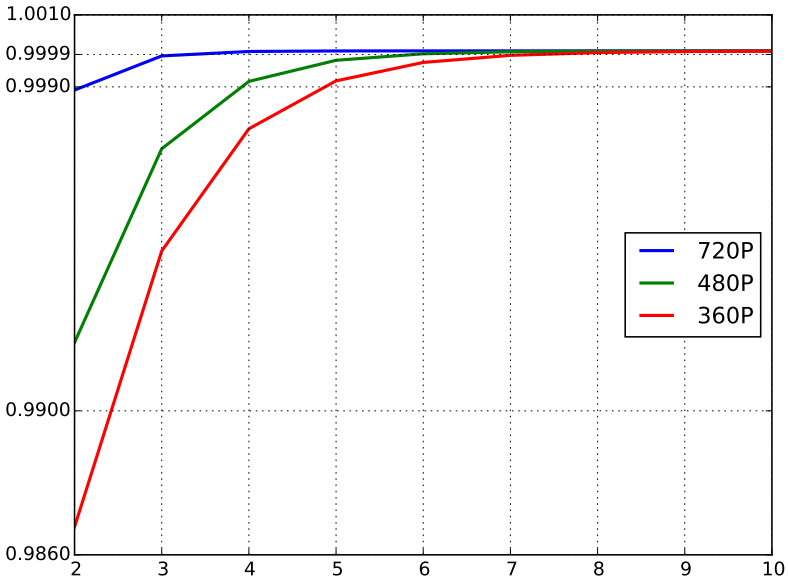


Figure 6.5 – Reliability

From equation 6.2, when $p > 0.5$, ω decreases when increasing n and the reliability increases monotonically. While $n \to \infty$, the value of reliability is infinitely close to 1. In

reverse, with $p < 0.5$, ω increases when increasing n and the reliability decreases monotonically. While $n \to \infty$, the value of reliability is infinitely close to 0. When $p = 0.5$ and $n \to \infty$, the reliability is infinitely close to $\frac{1}{2}$.

The significance of equation 6.2 is that more than half of total results points to the same result, the probability of this result being correct is approaching 100%. It also shows this probability is cumulative by increasing the number of video streams. Although we cannot change the initial detection accuracy for each format, we are still able to reduce the probability of returning an erroneous result. It is able to offer up to 99.999% or even higher probability for proving the result is on the side of 96.7% instead of 3.3%. In other words, we can infinity reduce the probability of occurrence of this 3.3%.

We introduce the **nines** conception which is typically expressed as a percentage with a number of nines (e.g., $99\% \rightarrow$ two nines, $99.9\% \rightarrow$ three nines, etc.). This conception is similar with the *high availability* conception in system design which aims to ensure an agreed level of operational performance. It could thus be used as a negotiated metric between the client and the cloud provider within an SLA (Service Level Agreement). From the cloud provider's perspective, the video resolution format can be seen as a green lever allowing for a controlled application performance degradation in return for lower resource allocation and thus, energy savings.

# **nines**	720p	480p	360p
99.9%	3	4	6
99.99%	4	6	8
99.999%	5	7	11

Table 6.3 – The number of cameras needed for achieving the indicate number of nines.

As shown in Figure 6.5 and Table 6.3, the resolution 360p requires 6 cameras working simultaneously in order to achieve *three nines*, the 480p requires 4 cameras and the 720p requires only 3 cameras to achieve the same level of accuracy. The higher the resolution is, the lower is the number of cameras required for reaching a given level of reliability.

6.4 Conclusion

Data loses its value when it cannot be analyzed quick enough. Offloading the data to process video streams at edge effectively reduces the response time and avoids unnecessary data transmission between edge and core, thus reducing the network energy overhead. Moreover, it can extend for instance the battery lifetime of end-user equipment (e.g., wearable equipment). Meanwhile, the traditional energy consumption and carbon footprint can be reduced by building self-producing electricity edge.

The study in Section 5.6 shows the possibility of using ESDs to further minimize these costs. Hence, we leverage on-site renewable energy production and ESDs, the simulated results shows that the energy saving is up to roughly half of total consumed energy compared with non-renewable energy configuration.

The scenario in this chapter can be seen as a concrete example to demonstrate the advantages of offloading the data to process at the edge or core in an energy saving context. In addition, although this work is camera-based, it can be applied to any other scenario where the data streams need to be processed in real-time as it provides the analytical framework for such applications.

7

Conclusion

Contents

7.1 Summary of the Dissertation

Renewable energy in the world has grown strongly in recent years. One reason is the solar-power generation efficiency significant increase. It enables the small-/medium-scale data centers to generate their own renewable energy. Thus they become self-sustainable and allow to reduce the fossil fuels (brown energy) consumption. As a consequence of the

renewable energy success, the cost of producing green energy is becoming cheaper than brown energy. The direct result is that the cost for the user to use the cloud to accomplish their tasks in this kind of data centers is falling in a similar way when renewable energy is available.

Unlike traditional infrastructures where energy sources are controllable, integrating renewable energy into a data center is difficult due to its intermittent and variable nature. Solar energy is considered as an admissible renewable source as solar panels are easy to install, they present a reasonable efficiency and the variations in their electricity production are not too abrupt (as for wind). Usually, most electricity is generated by solar panels during the day and its peak power occurs always near the midday. However, workloads do not necessarily follow the renewable energy production pattern which may result in a waste of energy.

In order to increase the usage of renewable energy, one way consists in carefully scheduling the workload to align it with the time-varying renewable energy production. We hence proposed **PIKA** framework making use of opportunistic scheduling for optimizing solar energy utilization in a small-/medium-scale data center without energy storage. This approach leveraged two ideas: 1) delay part of jobs which could be suspended within limited time (e.g., batch jobs) until solar energy becomes available; 2) when the renewable energy production cannot fully support the entire workload energy consumption, the system migrates the jobs from under-utilized servers to others and switches-off them with the help of consolidation techniques. However, this approach offers an efficient solution when part of the jobs are delay-tolerant (e.g., batch jobs). In this contribution, we allowed some jobs to be delayed in order for the workload to follow the renewable energy generation that maximizes the green energy usage. In addition, we explained that such a system cannot be satisfied when the workload contains only real-time jobs.

Another way consists in using Energy Storage Devices (ESDs) to store the surplus electricity generated from renewable energy sources. Through integrating ESDs, real-time jobs can always have access to green energy and so they are not forced to be delayed. Nevertheless, a penalty occurs because storing energy into batteries leads to an energy loss due to energy transformation. In this contribution, we explored the opportunistic-scheduling-based and ESD-based approaches to maximize the utilization of on-site renewable energy for small data centers. By using real-world job workload and solar energy traces, the experimental results showed the energy consumption with varying battery size and solar panel dimensions for opportunistic scheduling or ESD-only solution. Meanwhile, we found that using opportunistic scheduling can effective reduce the demand for battery size. Consequently, we proposed a solution mixing both approaches in order to achieve a balance in all aspects, implying minimizing the renewable energy losses (i.e. virtual machine migration cost for opportunistic scheduling and loss due to energy efficiency of battery).

The balance between the opportunistic-scheduling-based and ESD-based approaches can be seen as a good solution for mono-site data centers. As edge cloud infrastructures are smaller in size than centralized data centers, they meet the prerequisites of our previously designed data center. We thus advocate for leveraging on-site renewable energy production in the different edge cloud nodes to green Internet of Things (IoT) systems while offering improved Quality of Service (QoS) compared to core cloud solutions. We proposed an analytic model to decide whether to offload computation from the objects to the edge or to the core Cloud, depending on the renewable energy availability and the desired application QoS. This model was validated on our application use-case that deals with video stream analysis from vehicle cameras. In addition, we proved the relationship between the number of cameras and the application accuracy, opening new research directions on finding relevant trade-offs between application performance degradation and energy consumption of

underlying cloud systems.

So far, we have completed the prototype of integrating renewable energy into a single data center in this dissertation. The prototype is following two approaches: (1) design an intelligent resource management, which takes advantage of renewable energy availability to perform opportunistic tasks and consolidation policies without taking into account batteries, then exploring the trade-off between energy saving and performance aspects in cloud system; (2) provide the reasonable solar farm size and battery size for a given data center through analyzing the workload executed inside, and find the maximum energy quantity that can be saved by an ESD-based approach.

7.2 Perspective

Looking forward, there are still many unanswered challenges. We present several potential future research directions in this section.

7.2.1 Thermal-aware job scheduling

As mentioned in Chapter 2, one of the primary energy consumer in current data centers is related to underutilized or idle servers, and to cooling cost. Since the utilization of servers have been significant increased through consolidation and optimized resource allocation policies, the heat rises primarily due to higher CPU load issues. Because excessive heat conditions consume large amounts of energy, we need to think over the spatial placement taking into account for thermal issue while mapping the tasks to the physical servers. Providing visibility on the impact of energy is crucial, and consequently finding out a reliable model to describe the relationship between performance and thermal. Ideally, we would like to propose a job scheduling algorithm which seeks a balance between thermal constraints and performance, hence avoiding or preventing the extreme heat conditions.

7.2.2 In-transit strategy

As mentioned in Section 6.3.3, the edge could be capable of generating its own energy and storing the surplus energy into an ESD. Our results show that the renewable energy can almost cover its total energy consumption with small-scale solar panels and batteries. Due to its limited computing resources, it cannot support huge amounts of processing that needs to occur at the same time. Thus, new incoming data streams have to move to core cloud for quick analysis. As we conclude previously, in order to reduce the brown energy consumption, it is then better to decrease the resolution format for all videos with a penalty on detection accuracy. From an environmental point of view, if the user expects high accuracy of detection and to consume clean energy instead of brown one, he first needs to ensure that the data is processed at the edge. As the number of users grows, we then have to increase not only the number of edge servers, but also the solar photovoltaic panels that are able to provide as much energy as the servers need.

To reduce the total brown energy consumption, another alternative solution, displayed in Figure 7.1, consists in changing the division of labor between edge and core. The finite computing resources at edge are no longer used for data analysis but for video decoding, sampling and encoding. As such 720p videos in particular consumes a lot of computing resources. Even when taking all the edge servers, it is still far from enough for processing all the 720p videos in the system. Thus, carefully using edge resources is important for the overall framework optimization. As described in Section 6.3, it needs to analyze 8 frames

every second for a video at 25 frames per second. It means that we select 1 frame out every 3 frames for processing. In particular, we expect that the sampling work can be done at the edge. When a new video stream arrives, the edge performs decoding, sampling and encoding successively on this video and then transfers it to core. Although the data has to move to core for processing, their size is reduced and the energy consumption over the network is also reduced. Unfortunately, the result of this scenario is unsatisfactory. Decoding a video at 720p is extremely fast but encoding will take 15 times more than decoding in our experiment. It leads to an additional delay (roughly 100 ms in our experiments) while the latency is crucial in this scenario. This opportunity for data movement could be explored through the development of a framework that couples the tasks and computes partially on the transferring path, thus reducing the network cost [Pet+14].

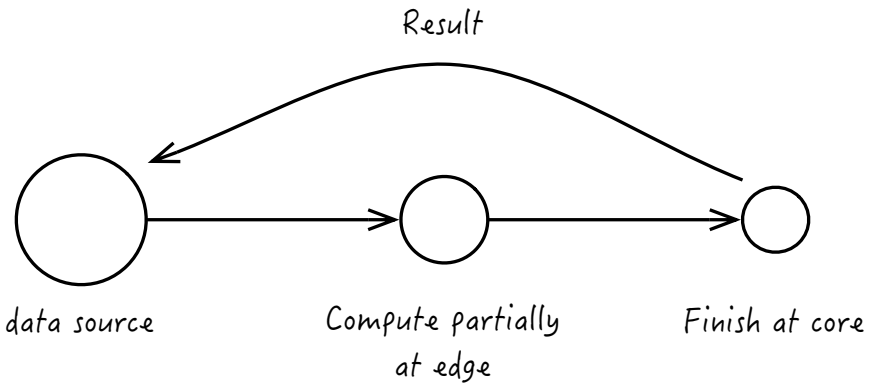


Figure 7.1 – Computing partially at edge

7.2.3 Geographically distributed data centers

Another challenge relates to achieve overall optimization on geographically distributed data centers. Previous studies have explored how to reduce the electricity bill by distributing the workloads to geographically distributed data centers with economical models. Meanwhile, migration enables the virtual machines to look for data center sites with renewable energy supply.

In our scenario, we have assumed that data centers are geographically distributed and each data center has its on-site renewable energy supply. Since the renewable energy for a single data center is intermittent and variable, moving the workload across data centers depending on the availability of renewable energy is an alternative way to solve the problem of mismatch between renewable energy supply and workloads. The main objective is still to increase renewable energy usage without SLA violations, thus leading to lower CO_2 emission footprints. It remains several problems including: (I) when to move the workload to other data centers: the availability of renewable energy is depending on the time zone and physical data center locations. While the renewable energy becomes unavailable in a data center, should it migrate its workload to other data centers or keep it until its completion; (II) which data center should be selected to move: we need to minimize the number of hops between the data centers while moving the workload in order not to widely increase the energy consumption of the network devices. If the renewable energy is predictable, how to decide the workload movement path? Lastly, we believe that building an energy infrastructure with centralized management is suitable for solving the above problems if the number of data centers is limited. Our previous works demonstrated the suitability of a framework integrating renewable energy into a single data center. We plan to expand this framework

to the geographically distributed data centers to further reduce brown energy consumption in cloud environments.

A

Appendix - Simplification of Equation 6.2

This appendix presents the proof of the simplification done in Section 6.3.4 for Equation 6.2.

Proposition: $\sum_{x=n+1}^{2n+1} C_{2n+1}^{x} (p)^{x} (1-p)^{2n+1-x}$ increase monotonically by increasing n while $p > 0.5$. **i.e.,**

$$\sum_{x=n+1}^{2n+1} C_{2n+1}^{x} (p)^{x} (1-p)^{2n+1-x} < \sum_{x=n+1}^{2n+2} C_{2n+2}^{x} (p)^{x} (1-p)^{2n+2-x} \tag{A.1}$$

proof:

$$\Pr(X \geq 0) = \sum_{x=0}^{2n+1} C_{2n+1}^{x} (p)^{x} (1-p)^{2n+1-x}$$

$$\begin{aligned}
&= C_{2n+1}^{0}[\underbrace{(p)^{0} (1-p)^{2n+1}}_{a_0} + \underbrace{(p)^{2n+1} (1-p)^{0}}_{b_0}] \\
&+ C_{2n+1}^{1}[(p)^{1} (1-p)^{2n} + (p)^{2n} (1-p)^{1}] \\
&+ C_{2n+1}^{2}[(p)^{2} (1-p)^{2n-1} + (p)^{2n-1} (1-p)^{2}] \\
&+ \ldots \\
&+ C_{2n+1}^{n}[\underbrace{(p)^{n} (1-p)^{n+1}}_{a_n} + \underbrace{(p)^{n+1} (1-p)^{n}}_{b_n}]
\end{aligned} \tag{A.2}$$

Where $a_0 = C_{2n+1}^{0} (p)^{0} (1-p)^{2n+1} = (1-p)^{2n+1}$. Then we simplify the common ratio $\frac{a_n}{a_{n-1}}$ and separate two sub-sequences a_n and b_n from $\Pr(X \geq 0)$

$$\begin{aligned}
\frac{a_n}{a_{n-1}} &= \frac{\frac{(2n+1)!}{(n+1)!n!} (p)^{n} (1-p)^{n+1}}{\frac{(2n)!}{(n-1)!n!} (p)^{n-1} (1-p)^{n}} \\
&= \frac{(1-p)}{p} \times \frac{(2n+1)n}{(n+1)n} \\
&= \frac{2(1-p)}{p} \times \frac{2n+1}{n}
\end{aligned} \tag{A.3}$$

The sub-sequence a_n can be expressed as:

$$\begin{aligned} a_n &= C_{2n+1}^{n} (p)^n (1-p)^{n+1} \\ &= \frac{(2n+1)!}{(n+1)!n!} (p)^n (1-p)^{n+1} \\ &= a_0 \cdot \left(\frac{2(1-p)}{p}\right)^n \cdot \frac{(2n+1)(2n-1)(2n-3)\cdot ... \cdot 5 \cdot 3 \cdot 1}{(n+1)!} \end{aligned} \tag{A.4}$$

Similarly, the sub-sequence b_n can be expressed as:

$$b_n = b_0 \cdot \left(\frac{2(1-p)}{p}\right)^n \cdot \frac{(2n+1)(2n-1)(2n-3)\cdot ... \cdot 5 \cdot 3 \cdot 1}{(n+1)!} \tag{A.5}$$

where $b_0 = p^{2n+1} \cdot (1-p)^0 = p^{2n+1}$. As we know,

$$(a+b)^n = \sum_{x=0}^{n} C_n^x (a)^x (b)^{n-x} \tag{A.6}$$

So,

$$\begin{aligned} (a+b)^{(2n+1)} &= \sum_{x=0}^{2n+1} C_{2n+1}^{x} (a)^x (b)^{2n+1-x} \\ (p+(1-p))^{(2n+1)} &= \sum_{x=0}^{2n+1} C_{2n+1}^{x} (p)^x (1-p)^{2n+1-x} \\ 1^{(2n+1)} &= \sum_{x=0}^{2n+1} C_{2n+1}^{x} (p)^x (1-p)^{2n+1-x} \end{aligned} \tag{A.7}$$

Let

$$h_n = \left(\frac{2(1-p)}{p}\right)^n \cdot \frac{(2n+1)(2n-1)(2n-3)... \cdot 5 \cdot 3 \cdot 1}{(n+1)!} \tag{A.8}$$

Accordingly, the sum of sub-sequennce a_n can be transformed as following:

$$\begin{aligned} S_a &= \sum_{x=0}^{n} C_{2n+1}^{x} (p)^x (1-p)^{2n+1-x} \\ S_a &= a_0 + a_1 + a_2 + a_3 + ... + a_n \\ &= a_0 + h_1 \cdot a_0 + h_2 \cdot a_0 + h_3 \cdot a_0 + ... + h_n \cdot a_0 \\ &= a_0 \cdot (1 + h_1 + h_2 + h_3 + ... + h_n) \end{aligned} \tag{A.9}$$

b_n is transformed in a similar way:

$$\begin{aligned} S_b &= \sum_{x=n+1}^{2n+1} C_{2n+1}^{x} (p)^x (1-p)^{2n+1-x} \\ &= b_0 \cdot (1 + h_1 + h_2 + h_3 + ... + h_n) \end{aligned} \tag{A.10}$$

The ration between S_a and S_b can be simplified:

$$\begin{aligned}\frac{S_a}{S_b} &= \frac{a_0}{b_0} \\ &= \frac{p^0 \cdot (1-p)^{2n+1}}{p^{2n+1} \cdot (1-p)^0} \\ &= \left(\frac{1-p}{p}\right)^{2n+1} \\ S_a &= \left(\frac{1-p}{p}\right)^{2n+1} \cdot S_b\end{aligned} \tag{A.11}$$

Finally, the equation A.7 can be transformed as:

$$\begin{aligned}S_a + S_b = 1 &= \left[\left(\frac{1-p}{p}\right)^{2n+1} + 1\right] \cdot S_b \\ S_b &= \frac{1}{1 + \left(\frac{1-p}{p}\right)^{2n+1}}\end{aligned} \tag{A.12}$$

B

Appendix - Simulator description

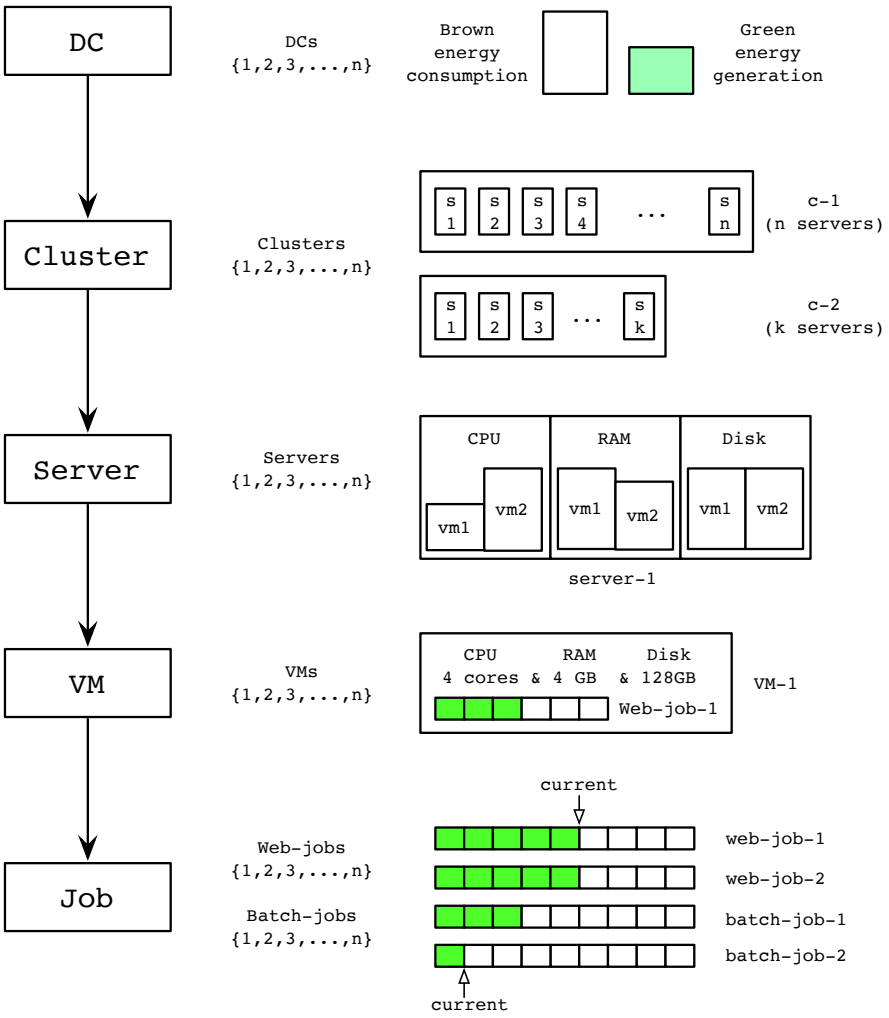


Figure B.1 – Data center architecture

Figure B.1 demonstrates the data center architecture and Figure B.2 shows the general conception of our simulator. We detail several selected components in this appendix.

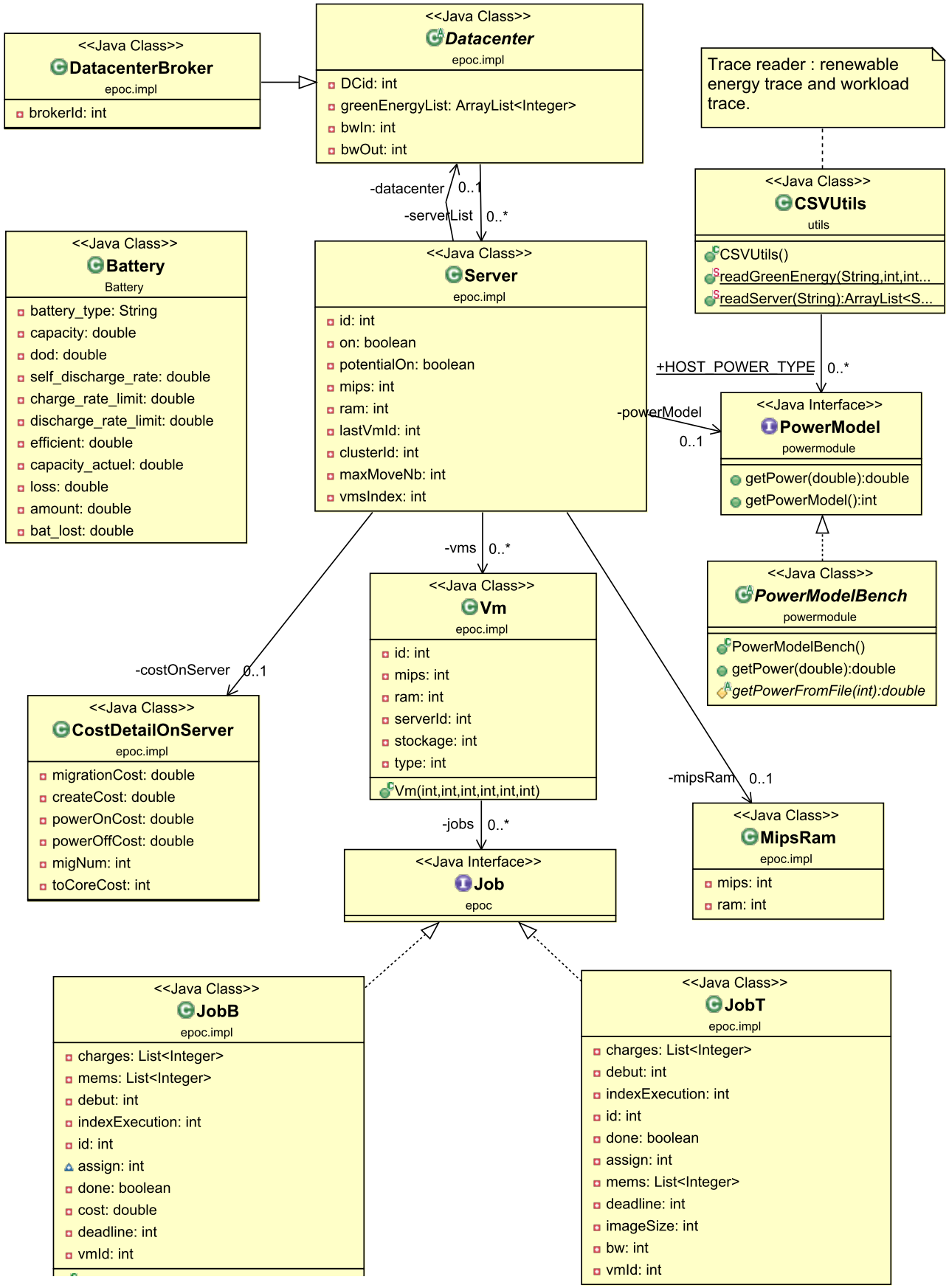


Figure B.2 – UML: Trace-driven cloud simulator included ESD and multiple-data-centers extensions

B.1 Data center module

Listing B.1 – Data center model

```
package epoc.impl;
import java.util.ArrayList;

public abstract class Datacenter {

    private int DCid;
     private ArrayList<Server> serverList = new ArrayList<Server>();
     private ArrayList<Integer> greenEnergyList = new ArrayList<Integer>();

    public Datacenter (int DCid){
       this.DCid = DCid;
    }
     ...
}
```

B.2 Server module

Listing B.2 – Server model

```
package epoc.impl;
import epoc.Job;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import powermodule.PowerModel;
import powermodule.PowerModelBench;
import utils.ListUtils;

public class Server implements Cloneable{
     ...
     private ArrayList<Vm> vms = new ArrayList<Vm>();
    private CostDetailOnServer costOnServer = new CostDetailOnServer();
    private MipsRam mipsRam = new MipsRam();
     ...

     public Server(int id, int mips, int ram, PowerModel powerModel,
                   int clusterId){
        this.id = id;
        this.mips = mips;
        this.ram = ram;
        this.powerModel = powerModel;
        this.clusterId = clusterId;
     }
     ...
}
```

B.3 Energy consumption module and interface

Listing B.3 – Server energy consumption model

```
package powermodule;
```

```
2
public class PowerServerEx1 extends PowerModelBench {
4
   private int modelNumber = 1;
6  private final double[] power = {97,128,150,158,165,171,
    177,185,195,200,204,212,220};
8
   @Override
10 protected double getPowerFromFile(int index) {
      return power[index];
12 }
    ...
14 }
```

B.4 VM module

Listing B.4 – VM model

```
package epoc.impl;
2 import java.util.ArrayList;
import java.util.Collection;
4 import epoc.Job;
import epoc.Vmm;
6
public class Vm implements Cloneable, Vmm {
8
    /**
10   * type: 0 -> web
     * type: 1 -> batch
12   * type: 2 -> mix (future maybe useful)
     */
14   private int type;  // 0: web 1: batch
     private ArrayList<Job> jobs = new ArrayList<Job>();
16
     ...
18
   public Vm(int id, int mips, int ram, int stockage,
20             int bandwidth, int type) {
      this.id = id;
22    this.mips = mips;
      this.ram = ram;
24    this.stockage = stockage;
      this.type = type;
26 }
    ...
28 }
```

B.5 Web/batch Jobs module

Listing B.5 – web- and batch-job models

```
package epoc.impl;
2 import java.util.List;
import epoc.Job;
4
public class JobB implements Job, Cloneable{
```

```
    // batch job
    ...
    private int vmId;

    public JobB(int id, int debut, int deadline,
                List<Integer> charges, List<Integer> mems){
      this.id = id;
        this.charges = charges;
        this.mems = mems;
        this.debut = debut;
        this.deadline = deadline;
    }
    ...
}

public class JobT implements Job, Cloneable {
    // web job
    ...
    private int vmId;

    public JobT(int id, int debut, int deadline,
                List<Integer> charges, List<Integer> mems){
        this.id = id;
        this.charges = charges;
        this.mems = mems;
        this.debut = debut;
        this.deadline = deadline;
    }
    // In this scenario, web job doesn't have deadline thus 0.
    this.deadline = 0;
    ...
}
```

B.6 ESD module

Listing B.6 – ESD Model

```
package Battery;

public class Battery implements Cloneable{
    ...

   public Battery(String battery_type, double capacity, double dod,
                  double self_discharge_rate, double charge_rate_limit,
                  double discharge_rate_limit, double efficient) {
      this.battery_type = battery_type;
      this.capacity = capacity;
      this.dod = dod;
      this.self_discharge_rate = self_discharge_rate;
      this.charge_rate_limit = charge_rate_limit;
      this.discharge_rate_limit = discharge_rate_limit;
      this.efficient = efficient;
   }
    ...
}
```

Résumé en français

C.1 Contexte

Le *Cloud computing* représente actuellement le principal paradigme pour fournir et gérer des ressources informatiques aux utilisateurs via le réseau. Le concept de base du Cloud Computing consiste à regrouper les ressources informatiques dans des centres de calcul distribués, plutôt qu'utiliser des ressources locales. Depuis son apparition, la demande de ressources informatiques et de stockage dans les data centers a connu une croissance rapide et l'utilisation des centres de calcul (DC) continue d'augmenter [OAaL14]. Par conséquent, la partie mondiale de l'électricité consacrée à la consommation des DC a atteint des niveaux sans précédent. En 2012, le nombre de DCs dans le monde était estimé à 509 147 pour une consommation approximativement à la production électrique de 30 centrales nucléaires [Gla12]. En 2016, une autre étude estime que, dans le monde, les DCs utilisent 91 milliards de kilowattheures d'électricité - assez pour alimenter deux fois la ville de New York - et leur consommation continue de croître rapidement [Res16].

Outre l'impact écologique, la consommation d'énergie est un critère prédominant pour les fournisseurs de clouds car elle détermine le coût quotidien de leur infrastructure. En conséquence, la gestion de l'alimentation électrique devient l'un des principaux défis pour les infrastructures de centres de calcul et plus généralement pour les systèmes distribués à grande échelle [OAaL14].

Parallèlement à l'expansion du Cloud computing, depuis plusieurs années, un nouveau modèle émerge : des infrastructures de cloud décentralisées [Ber+14]. Pour améliorer les performances de leur cloud et exploiter leur infrastructure disponible, les opérateurs de télécommunications, comme Orange, tentent de déployer des micro-centres de données (20 à 50 serveurs par micro-DC) au niveau des points de présence réseau au plus près des clients. Dans ce nouveau modèle, en déployant des centres de données au plus près de l'utilisateur, les opérateurs de clouds visent à améliorer le temps de réponse et le débit des applications.

Une façon d'économiser de l'énergie au niveau d'un centre de données consiste à le localiser à proximité d'une source d'électricité, ce qui réduit les pertes de transmission. Par exemple, l'ouest de la Caroline du Nord, aux États-Unis, attire les centres de données grâce au faible prix de l'électricité en raison de la capacité abondante de ses centrales à charbon et nucléaire suite au départ des entreprises textiles de la région [Gre]. En 2011, cette région dispose de trois très grands centres de données de Google, Apple et Facebook avec

des demandes de puissance respectives de 60 à 100 MW, 100 MW et 40 MW [Gre] et ces DCs sont toujours utilisés. Cependant, les installations de cette taille ne représentent qu'une petite fraction de la consommation mondiale des centres de données. En effet, les salles de serveurs de petite et moyenne taille continuent de représenter près de la moitié de la consommation d'électricité du marché [WD14].

D'autres entreprises optent pour des sources d'énergie plus écologiques. Par exemple, Quincy (Washington, États-Unis) fournit de l'électricité aux installations de données de Yahoo, Microsoft, Dell et Amazon avec ses centrales hydro-électriques peu coûteuses laissées inutilisées après la fermeture des industries d'aluminium de la région [Gre]. Plusieurs sources d'énergie renouvelables comme l'énergie éolienne, l'énergie solaire, l'hydroélectricité, la bioénergie, la géothermie et l'énergie marine peuvent être considérées comme alimentant des installations de grande taille. La variabilité de la production de la plupart des sources renouvelables conduit les installations des centres de données à ne compter que partiellement sur elles et à dépendre également du réseau électrique classique en cas de besoin.

Bien que l'utilisation de sources d'énergie renouvelable apporte de nouvelles opportunités pour réduire les coûts énergétiques, réduire les pics de charge, ou les deux [Goi+13], ces sources sont intermittentes et fluctuantes au cours du temps (soleil, vent, etc.). Ces variations peuvent entraîner des pertes d'électricité si la consommation de la charge de calcul ne correspond pas à la production d'énergie renouvelable. Les infrastructures de clouds, en revanche, peuvent profiter de plusieurs emplacements pour augmenter leur consommation verte avec des approches telles que *follow-the-sun* et *follow-the-wind* [Fig+09]. Puisque le soleil et le vent fournissent de l'énergie renouvelables dont la capacité fluctue au fil du temps, l'idée consiste à placer des tâches de calcul sur des ressources utilisant des énergies renouvelables et à migrer ces tâches à mesure que l'énergie renouvelable devient disponible sur les ressources d'autres sites.

Du point de vue de l'énergie, ces micro-DCs permettent d'étudier de nouvelles solutions d'alimentation électrique à base d'énergie renouvelable, comme le vent ou le soleil. L'utilisation de ces sources d'énergie renouvelables peut réduire les coûts d'exploitation, mais, malheureusement, ce type d'énergie reste intermittent par nature.

C.2 Problématique

Au cours de la dernière décennie, il y a eu des améliorations substantielles dans l'efficacité énergétique des centres de données. Une grande partie des progrès sur l'efficacité a été réalisée dans le domaine de l'infrastructure et des équipements. Par exemple, Google et Facebook ont développé leurs propres fermes de serveurs ultra-efficaces avec des solutions de refroidissement performantes. Alors que le serveur moderne est devenu plus efficace en énergie, malgré les nombreux travaux pour améliorer la gestion de l'énergie dans les possesseurs, les serveurs sont encore loin d'un comportement purement proportionnel en énergie. En effet, un serveur inactif peut consommer jusqu'à 50 % de sa puissance maximale [LWW07; FWB07; MGW09].

Peu de progrès ont été réalisés dans le domaine de l'efficacité énergétique des serveurs par rapport à leur utilisation. L'utilisation du serveur représente le rapport entre les ressources physiques (par exemple, CPU, RAM) consommées par la charge de calcul et la capacité maximale du serveur [BH07]. Plusieurs études montrent que l'utilisation moyenne du serveur - en particulier, en termes d'utilisation moyenne du processeur - reste constante autour de 12 à 18% [WK12; Sny10]. Ces serveurs inactifs ou sous-utilisés consomment beaucoup d'énergie alors que la plupart du temps ils effectuent peu de travail. La virtualisation a été rapidement et largement mise en œuvre dans les centres de données modernes. Cette technologie

permet à une seule machine physique d'exécuter plusieurs systèmes d'exploitation isolés simultanément. Cela signifie également que, en utilisant moins de machines physiques, on peut gérer la même quantité de tâches de calcul. Pourtant, même avec la virtualisation, qui est largement déployée dans les centres de données, l'utilisation moyenne des serveurs est généralement inférieure à 40% [WD14].

L'intégration des énergies renouvelables dans les DCs devrait être un facteur important dans la conception de la prochaine génération de DCs. Cela peut compenser une partie de l'énergie consommée par l'approvisionnement traditionnel (par exemple, le combustible fossile), réduisant ainsi les émissions de carbone. Cependant, un défi majeur pour l'utilisation de sources d'énergie renouvelables dans les DCs, comme le solaire et le vent, est leur nature variable et intermittente. Contrairement aux sources d'énergie traditionnelles qui permettent de fournir une puissance contrôlable et régulière, avec l'énergie renouvelable, il est difficile de satisfaire aux exigences de puissance de charge de travail. Une autre méthode possible pour améliorer l'utilisation efficace des énergies intermittentes et fluctuantes consiste à utiliser des dispositifs de stockage d'énergie (par exemple des piles) pour stocker l'excédent de production verte et l'utiliser pendant les périodes de faible production [Goi+13]. Généralement, pour les sources solaires, l'énergie peut être stockée pendant la journée - sinon entièrement consommée - et être utilisée pendant les nuits où il n'y a pas de production. Cependant, les batteries ont une efficacité énergétique inhérente (leur rendement) qui entraîne des pertes d'énergie. Un système pour la gestion des ressources de Cloud de type Infrastructure-as-a-Service (IaaS) pour un centre de données est nécessaire pour améliorer non seulement l'efficacité énergétique, mais aussi pour donner des moyens d'optimiser l'utilisation des énergies renouvelables.

Nos objectifs sont les suivants :

— explorer l'intégration d'énergies renouvelables dans les DCs,
— concevoir des algorithmes de gestion de ressources pour augmenter l'efficacité énergétique et optimiser la consommation d'énergie renouvelable pour un DC de Cloud;
— proposer un système utilisant des dispositifs de stockage d'énergie et des algorithmes de gestion de ressources pour maximiser la consommation d'énergie renouvelable dans un DC de Cloud;
— valider ce système dans un contexte réaliste.

C.3 Contributions

L'objectif principal de cette thèse consiste à maintenir un faible niveau de consommation d'énergie fossile dans un centre de données, réduisant ainsi les émissions de CO^2. Nous commençons par observer de véritables traces de production d'énergie solaire qui montrent la nature intermittente et variable des énergies renouvelables. Une autre analyse est réalisée sur une trace d'utilisation réelle de serveurs d'un petit centre de données. Elle démontre que l'utilisation moyenne des serveurs reste très faible en termes d'utilisation du CPU. Cette analyse montre également que la tendance moyenne de l'utilisation du serveur est moins variable que l'énergie renouvelable qui est intermittente par nature. Nous constatons aussi qu'une partie des tâches de calcul présente des périodes d'inactivité qui permettent de déplacer les calculs dans le temps. Selon ces observations et analyses, nous présentons nos contributions dans cette thèse comme suit :

1. Nous proposons un nouveau système : oPportunistic schedulIng broKer infrAstructure (PIKA) [LOM15] pour économiser de l'énergie dans un petit centre de données mono-site. PIKA vise à réduire la consommation d'énergie brune (c'est-à-dire provenant de sources d'énergie non renouvelables) et améliore l'utilisation des éner-

gies renouvelables sans stockage d'énergie pour le centre de données mono-site. Il exploite des tâches de calcul avec des périodes d'inactivité et les exécute ou les suspend en fonction de la disponibilité des énergies renouvelables. En consolidant les machines virtuelles (VM) sur les serveurs physiques, PIKA ajuste le nombre de serveurs alimentés pour que la consommation globale d'énergie corresponde à l'alimentation en énergie renouvelable.

2. Une autre approche pour améliorer l'utilisation efficace des énergies renouvelables intermittentes et fluctuantes consiste à utiliser des piles pour stocker l'excédent de production verte et à l'utiliser pendant les périodes de faible production. Généralement, pour les sources solaires, l'énergie peut être stockée pendant la journée - sinon entièrement consommée - et être utilisée pendant les nuits où il n'y a pas de production. Cependant, les dispositifs de stockage d'énergie (ESD) ont une efficacité énergétique inhérente en raison des différentes technologies de batterie qui entraînent des pertes d'énergie. Dans la deuxième contribution, on discute à la fois de l'approche opportuniste et des approches basées sur les ESD pour maximiser l'utilisation des énergies renouvelables dans les centres de données de petite et moyenne taille, et nous proposons une solution qui combine les deux approches [LOM17].
3. Enfin, inspiré par les travaux précédents, nous proposons de tirer parti de la production d'énergie renouvelable sur site dans les différents nœuds des Clouds de type *Edge* pour rendre l'Internet des objets (IoT) plus vert [Li+17]. Notre objectif est d'évaluer, sur un cas d'utilisation concret, les avantages du edge computing en matière de consommation d'énergie renouvelable. Nous proposons un modèle analytique pour décider où exécuter les tâches calcul (dans le DC du edge ou du cœur du cloud) en fonction de la disponibilité d'énergie renouvelable et de la qualité de service souhaitée par l'application, en particulier en réalisant un compromis entre performance (temps de réponse) et la fiabilité (précision du service).

Cette thèse a été réalisée dans le cadre du projet EPOC (Energy Proportional and Opportunistic Computing systems) financé par le Labex CominLabs (2013-2017).

C.4 Approche opportuniste

Du point de vue de l'énergie, les micros data centers permettent d'étudier de nouvelles solutions d'alimentation électrique à base d'énergie renouvelable, comme l'éolien ou le solaire. L'utilisation de ces sources d'énergie renouvelables peut réduire les coûts d'exploitation, malheureusement, ce type d'énergie reste intermittent par nature. Pour résoudre ce problème, nous envisageons deux solutions : investir dans des systèmes de batterie coûteux pour faciliter la production d'énergie renouvelable sur site ou développer de nouvelles solutions de gestion d'applications adaptées à ce type de production d'électricité.

Nous proposons de concevoir une approche opportuniste pour la gestion des ressources de Cloud qui profite de la disponibilité de l'énergie solaire pour effectuer des tâches de calcul supplémentaries. Le micro-DC reçoit une quantité fixe d'énergie du réseau électrique ordinaire. Ce genre d'électricité permet d'exécuter les tâches habituelles. En outre, le micro-DC est également relié à des sources d'énergie renouvelables (comme une éolienne ou des panneaux photovoltaïques) et lorsque ces sources produisent de l'électricité, le micro-DC l'utilise pour exécuter plus de tâches moins urgentes. Afin de réaliser cette allocation de ressources, nous distinguons deux sortes de tâches à programmer sur le DC : les tâches web qui représentent des tâches nécessitant une exécution continue (comme les serveurs Web) et les tâches batch qui représentent des calculs pouvant être retardés et interrompus, mais

avec une contrainte de date limite de terminaison (*deadline*). Ce deuxième type de tâches est le candidat naturel de l'algorithme d'ordonnancement opportuniste.

Nous présentons notre framework PIKA (oPportunistic schedulIng broKer infrAstructure), un système visant à réduire la consommation d'énergie non-propre (c'est-à-dire à partir de sources d'énergie non renouvelables) et à améliorer l'utilisation des énergies renouvelables pour un DC. Il exploite la présences de courtes tâches interruptibles les exécute ou les suspend en fonction de la disponibilité des énergies renouvelables. En consolidant les machines virtuelles (VM) sur moins de serveurs physiques, PIKA ajuste le nombre de serveurs allumés afin que la consommation globale d'énergie corresponde au mieux à l'alimentation en énergie renouvelable. À l'aide de simulations générées par les workloads réels et des traces d'énergie solaire, nous démontrons que la PIKA consomme 44,9% moins d'énergie brune et augmente de 110,1% le taux d'intégration des énergies renouvelables par rapport à l'algorithme glouton classique de la littérature.

C.5 Combiner l'utilisation de batteries et l'approche opportuniste

Ensuite, nous abordons deux approches pour maximiser l'utilisation des énergies renouvelables dans un petit ou moyen DC : l'apporche opportuniste (comme indiqué dans la contribution précédente) et les batteries (*Energy storage devices*). Nous comparons ces deux solutions en termes d'utilisation d'énergie renouvelable et de consommation totale d'énergie afin d'estimer si les pertes dues à l'efficacité de la batterie sont plus importantes, ou non, que les pertes dues aux coûts de migration des VMs et d'allumage/extinction des serveurs par l'approche opportuniste. Nous évaluons également une solution intermédiaire combinant les deux approches. Cette étude étudie deux types de batteries (acide à base de plomb et lithium-ion, mais peut être facilement généralisé à d'autres types d'ESD), la taille optimale des panneaux photovoltaïques, plusieurs profils solaires et traces de workload provenant de DCs réels. Nous considérons uniquement la production d'énergie renouvelable sur site (avec des panneaux photovoltaïques) et nous ne revendons pas l'énergie produite à d'autres acteurs : seule l'autoconsommation est considérée ici.

Notre première étude consiste à analyser le workload pour trouver la dimension idéale du panneau solaire et la taille de la batterie, ce qui permet d'alimenter tout le workload sans consommation d'énergie non-renouvelable. Cependant, en réalité, la dimension du panneau solaire ou la taille de la batterie sont limitées et nous devons encore résoudre le problème de l'adaptation de la consommation du workload et de la production d'énergie renouvelable. Notre deuxième étude montre que l'approche opportuniste peut réduire la demande de taille de la batterie alors que l'énergie renouvelable est insuffisante. Enfin, nos derniers résultats démontrent que, pour différentes tailles de batteries et dimensions de panneaux solaires, nous pouvons trouver une solution optimale combinant les deux approches qui équilibre les pertes d'énergie en raison de différentes causes telles que l'efficacité de la batterie et les migrations de VMs causées par les algorithmes de consolidation. Nous croyons que ces travaux constituent une méthode intéressante pour dimensionner correctement l'infrastructure de production d'énergie sur site, lorsque les caractéristiques du workload sont connues.

C.6 Explorer l'intégration d'énergie solaire dans le Edge Computing

Nous avons étendu nos travaux précédents pour tirer parti de la production d'énergie renouvelable sur site dans les différents nœuds des Cloud qui se situent au plus proche des utilisateurs, en particulier pour traiter les applications d'IoT (*Internet of Things*). Nous proposons un modèle analytique pour décider de décharger le calcul des objets vers le Edge ou vers le cœ du Cloud, en fonction de la disponibilité d'énergie renouvelable et de la qualité de service souhaitée. Afin de mener une analyse approfondie avec des hypothèses pertinentes menant à des contributions concrètes, nous décidons de nous concentrer sur un cas d'utilisation particulier avec des exigences précises et une modélisation réaliste. Notre cas d'utilisation cible l'Internet des véhicules (IoV) qui peut être considéré comme une convergence de l'Internet mobile et l'IoT. En particulier, nous nous concentrons sur les flux vidéos des caméras embarqués sur des véhicules qui doivent être analysés habituellement pour la détection d'objets sur les routes. En particulière, comme c'est souvent le cas avec les applications IoT avec un niveau élevé de QoS requise, les données perdent leur valeur lorsqu'elles ne peuvent être analysées assez rapidement.

Nos résultats montrent que, envoyer les données et traiter les flux vidéos dans le edge réduit efficacement le temps de réponse et évite la transmission inutile de données entre le Edge et le cœur du cloud, réduisant ainsi les frais de consommation énergétique du réseau. En outre, cela peut accroître la durée de vie de la batterie de l'équipement de l'utilisateur final (par exemple, un équipement portable). Pendant ce temps, la consommation d'énergie traditionnelle et les émissions de carbone peuvent être réduites en construisant un DC Edge avec production d'énergie renouvelable sur site.

C.7 Organisation du manuscrit

Le manuscrit est organisé comme suit.

Le chapitre 2 examine les efforts récents pour économiser de l'énergie au niveau de l'infrastructure dans les centres de données. Nous mettons en évidence les différents mécanismes au niveau du serveur pour économiser la consommation d'énergie non-renouvelable, notamment en ce qui concerne la proportionnalité énergétique et l'efficacité énergétique. Nous identifions les opportunités pour réduire davantage la consommation d'énergie des centres de données en rapport avec l'intégration des énergies renouvelables. Nous examinons ensuite l'état de l'art technique pour augmenter l'utilisation des énergies renouvelables.

Le chapitre 3 présente l'architecture EpoCloud du matériel aux couches middleware: cette architecture de centre de données a été conçue dans le cadre du projet EPOC. Ce prototype vise à optimiser la consommation d'énergie des centres de données cloud mono-site connectés au réseau électrique ordinaire et aux sources d'énergie renouvelable. Plus loin dans ce chapitre, nous décrivons un simulateur basés sur des traces réelles et développé dans le cadre de cette thèse qui est utilisé pour des parties d'expérimentation faites dans cette thèse.

Le chapitre 4 présente PIKA le système que nous proposons et qui intègre des algorithmes d'ordonnancement opportunistes pour économiser de l'énergie et basés sur la distinction de deux types de tâches (tâches Web et tâches Batch) et qui exploite la disponibilité d'énergie renouvelable pour effectuer des tâches opportunistes sans compromettre les performances.

Le chapitre 5 présente une approche basée sur des batteries pour augmenter l'utilisation

des énergies renouvelables. Nous étudions les performances entre l'approche opportuniste et la solution batterie uniquement. Une solution hybride est proposée plus loin dans ce chapitre pour trouver un compromis satisfaisant entre ces deux approches.

Le chapitre 6 préconise de tirer parti de la production d'énergie renouvelable sur site dans les différents nœuds d'un edge cloud pour rendre les systèmes IoT plus verts tout en offrant une QoS améliorée par rapport à la solution cloud traditionnelle. Nous proposons un modèle analytique pour décider de décharger le calcul des objets au Cloud, edge ou de cœur, en fonction de la disponibilité d'énergie renouvelable et de la qualité de service souhaitée. Ce modèle est validé sur notre cas d'utilisation qui effectue l'analyse de flux vidéo provenant de caméras embarquées sur des véhicules.

Le chapitre 7 conclut et présente des perspectives de recherche à ces travaux.

Bibliography

[Abd+14] S. Abdelwahab, B. Hamdaoui, M. Guizani, and A. Rayes. « Enabling Smart Cloud Services Through Remote Sensing: An Internet of Everything Enabler ». In: *IEEE Internet of Things Journal* 1.3 (2014), pp. 276–288.

[ABP11] Dan Azevedo, Symantec Christian Belady, and J Pouchet. « Water usage effectiveness (WUE™): A green grid data center sustainability metric ». In: *The Green Grid* (2011).

[AF+15] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. « Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications ». In: *IEEE Communications Surveys & Tutorials* 17.4 (2015), pp. 2347–2376.

[Anj+16] Ashiq Anjum, Tariq Abdullah, M Tariq, Yusuf Baltaci, and Nick Antonopoulos. « Video Stream Analysis in Clouds: An Object Detection and Classification Framework for High Performance Video Analytics ». In: *IEEE Transactions on Cloud Computing* (2016).

[Aze+10] Dan Azevedo, Symantec Michael Patterson, J Pouchet, and R Tipley. « Carbon usage effectiveness (CUE): a green grid data center sustainability metric ». In: *White Paper* 32 (2010).

[Bac+16] Enzo Baccarelli, Nicola Cordeschi, Alessandro Mei, Massimo Panella, Mohammad Shojafar, and Julinda Stefa. « Energy-efficient dynamic traffic offloading and reconfiguration of networked data centers for big data stream mobile computing: review, challenges, and a case study ». In: *IEEE Network* 30.2 (2016), pp. 54–61.

[Bal+13] Daniel Balouek, Alexandra Carpen Amarie, Ghislain Charrier, Frédéric Desprez, Emmanuel Jeannot, Emmanuel Jeanvoine, Adrien Lèbre, David Margery, Nicolas Niclausse, Lucas Nussbaum, Olivier Richard, Christian Pérez, Flavien Quesnel, Cyril Rohr, and Luc Sarzyniec. « Adding Virtualization Capabilities to the Grid'5000 Testbed ». In: *Cloud Computing and Services Science*. Vol. 367. Springer, 2013, pp. 3–20.

[BB10a] Anton Beloglazov and Rajkumar Buyya. « Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers. » In: *MGC@ Middleware*. 2010, p. 4.

[BB10b] Anton Beloglazov and Rajkumar Buyya. « Energy efficient resource management in virtualized cloud data centers ». In: *IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*. 2010, pp. 826–831.

[BBDM99] Luca Benini, Alessandro Bogliolo, and Giovanni De Micheli. « System-level dynamic power management ». In: *Low-Power Design, 1999. Proceedings. IEEE Alessandro Volta Memorial Workshop on*. IEEE. 1999, pp. 23–31.

[BCT16] Roi Blanco, Matteo Catena, and Nicola Tonellotto. « Exploiting Green Energy to Reduce the Operational Costs of Multi-Center Web Search Engines ». In: *International Conference on World Wide Web (WWW)*. 2016, pp. 1237–1247.

[Bel+08] Christian Belady, Andy Rawson, John Pfleuger, and Tahir Cader. *Green grid data center power efficiency metrics: PUE and DCIE*. Tech. rep. Technical report, Green Grid, 2008.

[Bel+11] Anton Beloglazov, Rajkumar Buyya, Young Choon Lee, Albert Zomaya, et al. « A taxonomy and survey of energy-efficient data centers and cloud computing systems ». In: *Advances in computers* 82.2 (2011), pp. 47–111.

[Bel+16] Nicolas Beldiceanu, Bárbara Dumas Feris, Philippe Gravey, Sabbir Hasan, Claude Jard, Thomas Ledoux, Yunbo Li, Didier Lime, Gilles Madi-Wamba, Jean-Marc Menaud, Pascal Morel, Michel Morvan, Marie-Laure Moulinard, Anne-Cécile Orgerie, Jean-Louis Pazat, Olivier H. Roux, and Ammar Sharaiha. « Towards energy-proportional Clouds partially powered by renewable energy ». In: *Computing* (2016), p. 20.

[Ber+14] Marin Bertier, Frédéric Desprez, Gilles Fedak, Adrien Lebre, Anne-Cécile Orgerie, Jonathan Pastor, Flavien Quesnel, Jonathan Rouzaud-Cornabas, and Cédric Tedeschi. « Beyond the Clouds: How Should Next Generation Utility Computing Infrastructures Be Designed? » In: *Cloud Computing*. Computer Communications and Networks. Springer, 2014, pp. 325–345.

[BEY05] Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis*. cambridge university press, 2005.

[BH07] Luiz André Barroso and Urs Hölzle. « The case for energy-proportional computing ». In: *Computer* 40.12 (2007).

[Bia+12] Ricardo Bianchini, Íngo Goiri, Kien Le, and Thu D Nguyen. « Parasol: A Solar-Powered μDatacenter ». In: (2012).

[BJO13] Frédéric Benhamou, Narendra Jussien, and Barry A O'Sullivan. *Trends in constraint programming*. John Wiley & Sons, 2013.

[Bro+10] A Brooks, E Lu, D Reicher, C Spirakis, and B Weihl. « Demand dispatch: using real-time control of demand to help balance generation and load. IEEE Power & Energy Magazine ». In: (2010).

[BSS06] Martin Bichler, Thomas Setzer, and Benjamin Speitkamp. « Capacity planning for virtualized servers ». In: (2006).

[Cer+16] Davide Cerotti, Marco Gribaudo, Riccardo Pinciroli, and Giuseppe Serazzi. « Stochastic analysis of energy consumption in pool depletion systems ». In: *International GI/ITG Conference on Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance*. Springer. 2016, pp. 25–39.

[CGH08] Kenneth Church, Albert G Greenberg, and James R Hamilton. « On Delivering Embarrassingly Distributed Cloud Services. » In: *HotNets*. Citeseer. 2008, pp. 55–60.

[Che+07] Feng Chen, Song Jiang, Weisong Shi, and Weikuan Yu. « Flexfetch: A history-aware scheme for i/o energy saving in mobile computing ». In: *Parallel Processing, 2007. ICPP 2007. International Conference on*. IEEE. 2007, pp. 10–10.

[Che+09] Haisheng Chen, Thang Ngoc Cong, Wei Yang, Chunqing Tan, Yongliang Li, and Yulong Ding. « Progress in electrical energy storage system: A critical review ». In: *Progress in Natural Science* 19.3 (2009), pp. 291–312.

[CHT12] Changbing Chen, Bingsheng He, and Xueyan Tang. « Green-aware workload scheduling in geographically distributed data centers ». In: *IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*. 2012, pp. 82–89.

[CK99] Chandra Chekuri and Sanjeev Khanna. « On Multi-Dimensional Packing Problems ». In: *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 1999, pp. 185–194.

[Cla+05] Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt, and Andrew Warfield. « Live migration of virtual machines ». In: *USENIX Symposium on Networked Systems Design & Implementation (NSDI)*. 2005, pp. 273–286.

[CLN09] Sivadon Chaisiri, Bu-Sung Lee, and Dusit Niyato. « Optimal virtual machine placement across multiple cloud providers ». In: *Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific*. IEEE. 2009, pp. 103–110.

[Cor] Intel Corp. *Intel Math Kernel Library 10.0 - LINPACK.*

[Cup+15] Leandro Cupertino, Georges Da Costa, Ariel Oleksiak, Wojciech Pia, Jean-Marc Pierson, Jaume Salom, Laura Siso, Patricia Stolf, Hongyang Sun, Thomas Zilio, et al. « Energy-efficient, thermal-aware modeling and simulation of data centers: the CoolEmAll approach and evaluation results ». In: *Ad Hoc Networks* 25 (2015), pp. 535–553.

[Dab+15a] Mehiar Dabbagh, Bechir Hamdaoui, Mohsen Guizani, and Ammar Rayes. « Efficient datacenter resource utilization through cloud resource overcommitment ». In: *IEEE International Conference On Computer Communications Workshops (INFOCOM)*. 2015, pp. 330–335.

[Dab+15b] Mehiar Dabbagh, Bechir Hamdaoui, Mohsen Guizani, and Ammar Rayes. « Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment ». In: *IEEE Network* 29.2 (2015), pp. 56–61.

[DHS12] Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification*. John Wiley & Sons, 2012.

[DKG12] Umesh Deshpande, Unmesh Kulkarni, and Kartik Gopalan. « Inter-rack live migration of multiple virtual machines ». In: *Proceedings of the 6th international workshop on Virtualization Technologies in Distributed Computing Date*. ACM. 2012, pp. 19–26.

[DM+13] Javier Diaz-Montes, Mengsong Zou, Ivan Rodero, and Manish Parashar. « Enabling autonomic computing on federated advanced cyberinfrastructures ». In: *Proceedings of the 2013 ACM Cloud and Autonomic Computing Conference*. ACM. 2013, p. 20.

[DØ09] KC Divya and Jacob Østergaard. « Battery energy storage technology for power systems?An overview ». In: *Electric Power Systems Research* 79.4 (2009), pp. 511–520.

[Fer+16] B. Dumas Feris, P. Gravey, P. Morel, M. L. Moulinard, M. Morvan, and A. Sharaiha. « 112-Gbit/s Passive Optical Pod Interconnect for small data centers using Pulse Amplitude Modulation ». In: *2016 International Conference on Optical Network Design and Modeling (ONDM)*. 2016, pp. 1–6.

[Ffm] *https://www.ffmpeg.org*. https://www.ffmpeg.org.

[Fig+09] S. Figuerola, M. Lemay, V. Reijs, M. Savoie, and B. St. Arnaud. « Converged Optical Network Infrastructures in Support of Future Internet and Grid Services Using IaaS to Reduce GHG Emissions ». In: *Journal of Lightwave Technology* 27.12 (2009), pp. 1941–1946.

[FRM11] Eugen Feller, Louis Rilling, and Christine Morin. « Energy-aware ant colony based workload placement in clouds ». In: *Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing*. IEEE Computer Society. 2011, pp. 26–33.

[FSR10] Will Fisher, Martin Suchara, and Jennifer Rexford. « Greening backbone networks: reducing energy consumption by shutting off cables in bundled links ». In: *ACM SIGCOMM workshop on Green networking*. 2010, pp. 29–34.

[FWB07] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso. « Power provisioning for a warehouse-sized computer ». In: *ACM SIGARCH Computer Architecture News*. Vol. 35. 2. 2007, pp. 13–23.

[GAL16] Misikir Eyob Gebrehiwot, Samuli Aalto, and Pasi Lassila. « Energy-Aware Server with SRPT Scheduling: Analysis and Optimization ». In: *International Conference on Quantitative Evaluation of Systems*. Springer. 2016, pp. 107–122.

[Gan+09] Anshul Gandhi, Mor Harchol-Balter, Rajarshi Das, and Charles Lefurgy. « Optimal power allocation in server farms ». In: *ACM SIGMETRICS Performance Evaluation Review*. Vol. 37. 1. ACM. 2009, pp. 157–168.

[GFKR15] Yashar Ghiassi-Farrokhfal, Srinivasan Keshav, and Catherine Rosenberg. « Toward a realistic performance analysis of storage systems in smart grids ». In: *IEEE Transactions on Smart Grid* 6.1 (2015), pp. 402–410.

[Gla12] James Glanz. *Power, Pollution and the Internet*. The New York Times. 2012.

[GN12] Rahul Ghosh and Vijay K Naik. « Biting off safely more than you can chew: Predictive analytics for resource over-commit in iaas cloud ». In: *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*. IEEE. 2012, pp. 25–32.

[Goi+11] Íñigo Goiri, Kien Le, Md E Haque, Ryan Beauchea, Thu D Nguyen, Jordi Guitart, Jordi Torres, and Ricardo Bianchini. « Greenslot: scheduling energy consumption in green datacenters ». In: *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM. 2011, p. 20.

[Goi+12] Íñigo Goiri, Kien Le, Thu D Nguyen, Jordi Guitart, Jordi Torres, and Ricardo Bianchini. « GreenHadoop: leveraging green energy in data-processing frameworks ». In: *Proceedings of the 7th ACM european conference on Computer Systems*. ACM. 2012, pp. 57–70.

[Goi+13] Íñigo Goiri, William Katsak, Kien Le, Thu D Nguyen, and Ricardo Bianchini. « Parasol and GreenSwitch: managing datacenters powered by renewable energy ». In: *ACM SIGARCH Computer Architecture News*. Vol. 41. 1. ACM. 2013, pp. 51–64.

[GOM17] David Guyon, Anne-Cécile Orgerie, and Chrisine Morin. « Energy-Aware Server with SRPT Scheduling: Analysis and Optimization ». In: *6th International Workshop on Energy-Efficient Data Centres*. ACM. 2017.

[Gor+11] Abel Gordon, Michael Hines, Dilma Da Silva, Muli Ben-Yehuda, Marcio Silva, and Gabriel Lizarraga. « Ginkgo: Automated, application-driven memory overcommitment for cloud computing ». In: *Proc. RESoLVE* (2011).

[Gre] *How dirty is your data?* Greenpeace report. 2011.

[Guo+11] Yuanxiong Guo, Zongrui Ding, Yuguang Fang, and Dapeng Wu. « Cutting down electricity cost in internet data centers by using energy storage ». In: *IEEE Global Telecommunications Conference (GLOBECOM)*. 2011, pp. 1–5.

[Haq+13] Md E Haque, Kien Le, Íñigo Goiri, Ricardo Bianchini, and Thu D Nguyen. « Providing green slas in high performance computing clouds ». In: *Green Computing Conference (IGCC), 2013 International*. IEEE. 2013, pp. 1–11.

[Her+09] Fabien Hermenier, Xavier Lorca, Jean-Marc Menaud, Gilles Muller, and Julia Lawall. « Entropy: a consolidation manager for clusters ». In: *Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*. ACM. 2009, pp. 41–50.

[HQ03] Shaoxiong Hua and Gang Qu. « Approaching the maximum energy saving on embedded systems with multiple voltages ». In: *Proceedings of the 2003 IEEE/ACM international conference on Computer-aided design*. IEEE Computer Society. 2003, p. 26.

[Hu+16] Wenlu Hu, Ying Gao, Kiryong Ha, Junjue Wang, Brandon Amos, Zhuo Chen, Padmanabhan Pillai, and Mahadev Satyanarayanan. « Quantifying the Impact of Edge Computing on Mobile Applications ». In: *ACM SIGOPS Asia-Pacific Workshop on Systems*. 2016, 5:1–5:8.

[Hua+11] Qiang Huang, Fengqian Gao, Rui Wang, and Zhengwei Qi. « Power consumption of virtual machine live migration in clouds ». In: *International Conference on Communications and Mobile Computing (CMC)*. 2011, pp. 122–125.

[Ind] *ACPI-Advanced Configuration and Power Interface Specification*. http://www.acpi.info/DOWNLOADS/ACPIspec50.pdf. 2011.

[Int08] SRI Consulting Business Intelligence. *Six Technologies with Potential Impacts on US Interests out to 2025*. Tech. rep. National Intelligent Concil, 2008.

[IS11] Atsushi Ishii and Toyotaro Suzumura. « Elastic stream computing with clouds ». In: *Cloud Computing (CLOUD), 2011 IEEE International Conference on*. IEEE. 2011, pp. 195–202.

[IY98] Tohru Ishihara and Hiroto Yasuura. « Voltage scheduling problem for dynamically variable voltage processors ». In: *Low Power Electronics and Design, 1998. Proceedings. 1998 International Symposium on*. IEEE. 1998, pp. 197–202.

[Jal+16] Fatemeh Jalali, Kerry Hinton, Robert Ayre, Tansu Alpcan, and Rodney S Tucker. « Fog Computing May Help to Save Energy in Cloud Computing ». In: *IEEE Journal on Selected Areas in Communications* 34.5 (2016), pp. 1728–1739.

[Kat+11] Randy H Katz, David E Culler, Seth Sanders, Sara Alspaugh, Yanpei Chen, Stephen Dawson-Haggerty, Prabal Dutta, Mike He, Xiaofan Jiang, Laura Keys, et al. « An information-centric energy infrastructure: The berkeley view ». In: *Sustainable Computing: Informatics and Systems* 1.1 (2011), pp. 7–22.

[KGB13] Atefeh Khosravi, Saurabh Kumar Garg, and Rajkumar Buyya. « Energy and carbon-efficient placement of virtual machines in distributed cloud data centers ». In: *European Conference on Parallel Processing*. Springer. 2013, pp. 317–328.

[Kim+08] Wonyoung Kim, Meeta S Gupta, Gu-Yeon Wei, and David Brooks. « System level analysis of fast, per-core DVFS using on-chip switching regulators ». In: *High Performance Computer Architecture, 2008. HPCA 2008. IEEE 14th International Symposium on*. IEEE. 2008, pp. 123–134.

[Koo11] Jonathan Koomey. « Growth in data center electricity use 2005 to 2010 ». In: *A report by Analytical Press, completed at the request of The New York Times* 9 (2011).

[KR07] Jihong Kim and Tajana Simunic Rosing. *Power-Aware Resource Management Techniques for Low-Power Embedded Systems.* 2007.

[Kri+11] Andrew Krioukov, Christoph Goebel, Sara Alspaugh, Yanpei Chen, David E Culler, and Randy H Katz. « Integrating Renewable Energy Using Data Analytics Systems: Challenges and Opportunities ». In: *IEEE Data Engineering Bulletin* 34.1 (2011), pp. 3–11.

[KT12] C. Kachris and I. Tomkos. « A Survey on Optical Interconnects for Data Centers ». In: *IEEE Communications Surveys Tutorials* 14.4 (2012), pp. 1021–1036.

[LH11] Antonio Luque and Steven Hegedus. *Handbook of photovoltaic science and engineering*. John Wiley & Sons, 2011.

[Li+17] Yunbo Li, Anne-Cécile Orgerie, Ivan Rodero, Manish Parashar, and Jean-Marc Menaud. « Leveraging Renewable Energy in Edge Clouds for Data Stream Analysis in IoT ». In: *IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*. Madrid, Spain, May 2017.

[Lin+13] Minghong Lin, Adam Wierman, Lachlan LH Andrew, and Eno Thereska. « Dynamic right-sizing for power-proportional data centers ». In: *IEEE/ACM Transactions on Networking (TON)* 21.5 (2013), pp. 1378–1391.

[Liu+15] Zhenhua Liu, Minghong Lin, Adam Wierman, Steven Low, and Lachlan LH Andrew. « Greening geographical load balancing ». In: *IEEE/ACM Transactions on Networking (TON)* 23.2 (2015), pp. 657–671.

[LOM15] Yunbo Li, Anne-Cécile Orgerie, and Jean-Marc Menaud. « Opportunistic Scheduling in Clouds Partially Powered by Green Energy ». In: *GreenCom: IEEE International Conference on Green Computing and Communications*. 2015, pp. 448–455.

[LOM17] Yunbo Li, Anne-Cécile Orgerie, and Jean-Marc Menaud. « Balancing the use of batteries and opportunistic scheduling policies for maximizing renewable energy consumption in a Cloud data center ». In: *PDP 2017-25th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*. 2017.

[LTC14] Yusen Li, Xueyan Tang, and Wentong Cai. « On dynamic bin packing for resource allocation in the cloud ». In: *Proceedings of the 26th ACM Symposium on Parallelism in Algorithms and Architectures*. ACM. 2014, pp. 2–11.

[LTC16] Yusen Li, Xueyan Tang, and Wentong Cai. « Dynamic bin packing for on-demand cloud resource allocation ». In: *IEEE Transactions on Parallel and Distributed Systems* 27.1 (2016), pp. 157–170.

[LWW07] Charles Lefurgy, Xiaorui Wang, and Malcolm Ware. « Server-level power control ». In: *Autonomic Computing, 2007. ICAC'07. Fourth International Conference on*. IEEE. 2007, pp. 4–4.

[Men+10] Xiaoqiao Meng, Canturk Isci, Jeffrey Kephart, Li Zhang, Eric Bouillet, and Dimitrios Pendarakis. « Efficient resource provisioning in compute clouds via vm multiplexing ». In: *Proceedings of the 7th international conference on Autonomic computing*. ACM. 2010, pp. 11–20.

[MGW09] David Meisner, Brian T Gold, and Thomas F Wenisch. « PowerNap: eliminating server idle power ». In: *ACM Sigplan Notices*. Vol. 44. 3. ACM. 2009, pp. 205–216.

[MSJ14] Ishai Menache, Ohad Shamir, and Navendu Jain. « On-demand, Spot, or Both: Dynamic Resource Allocation for Executing Batch Jobs in the Cloud ». In: *USENIX International Conference on Autonomic Computing (ICAC)*. June 2014, pp. 177–187.

[MZL16] Yuyi Mao, Jun Zhang, and Khaled B Letaief. « Dynamic computation offloading for mobile-edge computing with energy harvesting devices ». In: *arXiv preprint arXiv:1605.05488* (2016).

[NCS09] Dusit Niyato, Sivadon Chaisiri, and Lee Bu Sung. « Optimal power management for server farm to support green computing ». In: *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*. IEEE Computer Society. 2009, pp. 84–91.

[Ned+08] Sergiu Nedevschi, Lucian Popa, Gianluca Iannaccone, Sylvia Ratnasamy, and David Wetherall. « Reducing Network Energy Consumption via Sleeping and Rate-Adaptation ». In: *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. Vol. 8. 2008, pp. 323–336.

[OAaL14] Anne-Cécile Orgerie, Marcos Dias de Assunção, and Laurent Lefèvre. « A survey on techniques for improving the energy efficiency of large-scale distributed systems ». In: *ACM Computing Surveys (CSUR)* 46.4 (2014), p. 47.

[OLG08] Anne-Cécile Orgerie, Laurent Lefèvre, and Jean-Patrick Gelas. « Save watts in your grid: Green strategies for energy-aware framework in large scale distributed systems ». In: *Parallel and Distributed Systems, 2008. ICPADS'08. 14th IEEE International Conference on*. IEEE. 2008, pp. 171–178.

[Pat+10] Mike Patterson, Bill Tschudi, Otto Vangeet, Jud Cooley, and Dan Azevedo. « ERE: A metric for measuring the benefit of reuse energy from a data center ». In: *White Paper* 29 (2010).

[Pet+14] I. Petri, J. Diaz-Montes, M. Zou, O. F. Rana, T. Beach, H. Li, and Y. Rezgui. « In-Transit Data Analysis and Distribution in a Multi-cloud Environment Using CometCloud ». In: *International Conference on Future Internet of Things and Cloud (FiCloud)*. 2014, pp. 471–476.

[PMGG14] Elina Pacini, Cristian Mateos, and Carlos García Garino. « Dynamic scheduling based on particle swarm optimization for cloud-based scientific experiments ». In: *CLEI Electronic Journal* 17.1 (2014), pp. 3–3.

[Pol+14] Marco Polverini, Antonio Cianfrani, Shaolei Ren, and Athanasios V Vasilakos. « Thermal-aware scheduling of batch jobs in geographically distributed data centers ». In: *IEEE Transactions on Cloud Computing* 2.1 (2014), pp. 71–84.

[QH01] Gang Quan and Xiaobo Hu. « Energy efficient fixed-priority scheduling for real-time systems on variable voltage processors ». In: *Proceedings of the 38th annual Design Automation Conference*. ACM. 2001, pp. 828–833.

[QKMM13] Flavien Quesnel, Hemant Kumar Mehta, and Jean-Marc Menaud. « Estimating the Power Consumption of an Idle Virtual Machine ». In: *IEEE International Conference on Green Computing and Communications (GreenCom)*. Beijing, China, Aug. 2013.

[Rao+12] Lei Rao, Xue Liu, Le Xie, and Wenyu Liu. « Coordinated energy cost management of distributed internet data centers in smart grid ». In: *IEEE Transactions on Smart Grid* 3.1 (2012), pp. 50–58.

[Ren+16] Runtian Ren, Xueyan Tang, Yusen Li, and Wentong Cai. « Competitiveness of Dynamic Bin Packing for Online Cloud Server Allocation ». In: *IEEE/ACM Transactions on Networking* (2016).

[Res16] Lux Research. *Coal Computing: How Companies Misunderstand Their Dirty Data Centers*. White paper. 2016.

[Riz+10] Nikzad Babaii Rizvandi, Javid Taheri, Albert Y Zomaya, and Young Choon Lee. « Linear combinations of dvfs-enabled processor frequencies to modify the energy-aware scheduling algorithms ». In: *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*. IEEE. 2010, pp. 388–397.

[RTZ11] Nikzad Babaii Rizvandi, Javid Taheri, and Albert Y Zomaya. « Some observations on optimal frequency selection in DVFS-based energy consumption minimization ». In: *Journal of Parallel and Distributed Computing* 71.8 (2011), pp. 1154–1164.

[RVBW06] Francesca Rossi, Peter Van Beek, and Toby Walsh. *Handbook of constraint programming*. Elsevier, 2006.

[Sat17] Mahadev Satyanarayanan. « The Emergence of Edge Computing ». In: *Computer* 50.1 (2017), pp. 30–39.

[SB10] Benjamin Speitkamp and Martin Bichler. « A mathematical programming approach for server consolidation problems in virtualized data centers ». In: *IEEE Transactions on services computing* 3.4 (2010), pp. 266–278.

[SBD07] R. Schmidt, D. Beaty, and J. Dietrich. « Increasing Energy Efficiency In Data Centers ». In: *ASHRAE Journal* (2007), pp. 18–24.

[Sha+10] Navin Sharma, Jeremy Gummeson, David Irwin, and Prashant Shenoy. « Cloudy computing: Leveraging weather forecasts in energy harvesting sensor systems ». In: *Sensor Mesh and Ad Hoc Communications and Networks (SECON), 2010 7th Annual IEEE Communications Society Conference on*. IEEE. 2010, pp. 1–9.

[She+16] A Shehabi, SJ Smith, N Horner, I Azevedo, R Brown, J Koomey, E Masanet, D Sartor, M Herrlin, and W Lintner. « United States data center energy usage report ». In: *Lawrence Berkeley National Laboratory, Berkeley, California. LBNL-1005775 Page* 4 (2016).

[Sim+13] Pieter Simoens, Yu Xiao, Padmanabhan Pillai, Zhuo Chen, Kiryong Ha, and Mahadev Satyanarayanan. « Scalable crowd-sourcing of video from mobile devices ». In: *ACM International conference on Mobile systems, applications, and services*. 2013, pp. 139–152.

[SLM16] Bianca Schroeder, Raghav Lagisetty, and Arif Merchant. « Flash reliability in production: The expected and the unexpected ». In: *14th USENIX Conference on File and Storage Technologies (FAST 16)*. 2016, pp. 67–80.

[Sny10] Bill Snyder. « Server virtualization has stalled, despite the hype ». In: *InfoWorld* (2010).

[SS09] Chistopher Stewart and Kai Shen. « Some Joules Are More Precious Than Others: Managing Renewable Energy in the Datacenter ». In: *Workshop on Power Aware Computing and Systems (HotPower)*. 2009.

[SS99] Robert E Schapire and Yoram Singer. « Improved boosting algorithms using confidence-rated predictions ». In: *Machine learning* 37.3 (1999), pp. 297–336.

[SSP17] Hongyang Sun, Patricia Stolf, and Jean-Marc Pierson. « Spatio-temporal thermal-aware scheduling for homogeneous high-performance computing datacenters ». In: *Future Generation Computer Systems* 71 (2017), pp. 157–170.

[TLC85] Marvin M Theimer, Keith A Lantz, and David R Cheriton. « Preemptable Remote Execution Facilities for the V-System ». In: (1985).

[TP15] Maolin Tang and Shenchen Pan. « A hybrid genetic algorithm for the energy-efficient virtual machine placement problem in data centers ». In: *Neural Processing Letters* 41.2 (2015), pp. 211–221.

[Var+16] Blesson Varghese, Nan Wang, Sakil Barbhuiya, Peter Kilpatrick, and Dimitrios S. Nikolopoulos. « Challenges and Opportunities in Edge Computing ». In: *CoRR* abs/1609.01967 (2016).

[Ver+07] Gary Verdun, Dan Azevedo, H Barrass, S Berard, M Bramfitt, T Cader, T Darby, C Long, N Gruendler, B Macarthur, et al. « The Green Grid metrics: data center infrastructure efficiency (DCiE) detailed analysis ». In: *The Green Grid, Tech. Rep* (2007).

[Ver+09] Akshat Verma, Gargi Dasgupta, Tapan Kumar Nayak, Pradipta De, and Ravi Kothari. « Server workload analysis for power minimization using consolidation ». In: *Proceedings of the 2009 conference on USENIX Annual technical conference*. USENIX Association. 2009, pp. 28–28.

[Vil+14] Violaine Villebonnet, Georges Da Costa, Laurent Lefevre, Jean-Marc Pierson, and Patricia Stolf. « Towards Generalizing" Big Little" for Energy Proportional HPC and Cloud Infrastructures ». In: *Big Data and Cloud Computing (BdCloud), 2014 IEEE Fourth International Conference on*. IEEE. 2014, pp. 703–710.

[Vil+15] Violaine Villebonnet, Georges Da Costa, Laurent Lefevre, Jean-Marc Pierson, and Patricia Stolf. « "Big, Medium, Little": Reaching Energy Proportionality with Heterogeneous Computing Scheduler ». In: *Parallel Processing Letters* 25.03 (2015), p. 1541006.

[VJ01] Paul Viola and Michael Jones. « Rapid object detection using a boosted cascade of simple features ». In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 1. 2001.

[VRM14] Luis M. Vaquero and Luis Rodero-Merino. « Finding Your Way in the Fog: Towards a Comprehensive Definition of Fog Computing ». In: *ACM SIGCOMM Computer Communication Review* 44.5 (Oct. 2014), pp. 27–32.

[Wan+12a] Di Wang, Chuangang Ren, Anand Sivasubramaniam, Bhuvan Urgaonkar, and Hosam Fathy. « Energy storage in datacenters: what, where, and how much? » In: *ACM SIGMETRICS Performance Evaluation Review*. Vol. 40. 1. ACM. 2012, pp. 187–198.

[Wan+12b] Kai Wang, Florin Ciucu, Chuang Lin, and Steven H Low. « A stochastic power network calculus for integrating renewable energy sources into the power grid ». In: *IEEE Journal on Selected Areas in Communications* 30.6 (2012), pp. 1037–1048.

[WCC14] Chia-Ming Wu, Ruay-Shiung Chang, and Hsin-Yu Chan. « A green energy-efficient scheduling algorithm using the DVFS technique for cloud datacenters ». In: *Future Generation Computer Systems* 37 (2014), pp. 141–147.

[WD14] Josh Whitney and Pierre Delforge. *Data Center Efficiency Assessment scaling up energy efficiency across the Data Center Industry: evaluating Key Drivers and Barriers*. NRDC white paper. 2014.

[WK12] J Whitney and J Kennedy. *The Carbon Emissions of Server Computing for Small-to Medium-sized Organization*. 2012.

[Wu+12] Grant Wu, Maolin Tang, Yu-Chu Tian, and Wei Li. « Energy-efficient virtual machine placement in data centers by genetic algorithm ». In: *International Conference on Neural Information Processing*. Springer. 2012, pp. 315–323.

[WWZ12] Xiaoli Wang, Yuping Wang, and Hai Zhu. « Energy-efficient multi-job scheduling model for cloud computing and its genetic algorithm ». In: *Mathematical Problems in Engineering* 2012 (2012).

[XL12] Dan Xu and Xin Liu. « Geographic trough filling for internet datacenters ». In: *IEEE International Conference On Computer Communications (INFOCOM)*. 2012, pp. 2881–2885.

[Yan+14] F. Yang, S. Wang, J. Li, Z. Liu, and Q. Sun. « An overview of Internet of Vehicles ». In: *China Communications* 11.10 (2014), pp. 1–15.

[Yao+12] Yuan Yao, Longbo Huang, Abhihshek Sharma, Leana Golubchik, and Michael Neely. « Data centers power reduction: A two time scale approach for delay tolerant workloads ». In: *IEEE International Conference On Computer Communications (INFOCOM)*. 2012, pp. 1431–1439.

[YDS95] Frances Yao, Alan Demers, and Scott Shenker. « A scheduling model for reduced CPU energy ». In: *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*. IEEE. 1995, pp. 374–382.

[Yue91] Minyi Yue. « A simple proof of the inequality $FFD(L) \leq \frac{11}{9}OPT(L) + 1, \forall L$ for the FFD bin-packing algorithm ». In: *Acta mathematicae applicatae sinica* 7.4 (1991), pp. 321–331.

[Zha+12] Xiangliang Zhang, Zon-Yin Shae, Shuai Zheng, and Hani Jamjoom. « Virtual machine migration in an over-committed cloud ». In: *IEEE Network Operations and Management Symposium (NOMS)*. 2012, pp. 196–203.

[Zhu+11] Wenwu Zhu, Chong Luo, Jianfeng Wang, and Shipeng Li. « Multimedia cloud computing ». In: *IEEE Signal Processing Magazine* 28.3 (2011), pp. 59–69.

[ZWW11] Yanwei Zhang, Yefu Wang, and Xiaorui Wang. « Greenware: Greening cloud-scale data centers to maximize the use of renewable energy ». In: *ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing*. Springer. 2011, pp. 143–164.

[Gre14] Green Grid. *Harmonizing Global Metrics for Data Center Energy Efficiency, Global Taskforce Reaches Agreement Regarding Data Center Productivity*. Tech. rep. The Green Grid Consortium, 2014.

UNIVERSITE BRETAGNE LOIRE

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Thèse de Doctorat

Yunbo Li

Allocation de ressources dans un Cloud partiellement alimenté par des sources d'énergie renouvelable

Resource allocation in a Cloud partially powered by renewable energy sources

Résumé
La plupart des infrastructures de cloud efficace en énergie proposées dans la littérature ne tiennent pas compte de la disponibilité électrique et des énergies renouvelables dans leurs modèles. L'intégration des énergies renouvelables dans les centres de données réduit considérablement leur consommation d'énergie et leur empreinte carbone. Étant donné que l'énergie renouvelable est intermittente et fluctue en fonction du temps, elle est habituellement sous-utilisée. Nous abordons le problème de l'amélioration de l'utilisation des énergies renouvelables dans un centre de données unique et étudions deux approches : la planification opportuniste et le stockage de l'énergie. Nos résultats démontrent que les deux approches permettent de réduire la consommation d'énergie non-renouvelable sous différentes configurations. Nous étendons ce travail au contexte des Edge Clouds et de l'Internet des Objets dans le cas de l'analyse de flux de données. Nous montrons comment rendre les Edge Clouds plus verts avec une production d'énergie renouvelable sur site combinée à un stockage d'énergie et à une dégradation de performance des applications des utilisateurs.

Mots clés
Cloud, énergie renouvelable, efficacité énergétique, allocation de ressources.

Abstract
Most of the energy-efficient Cloud frameworks proposed in literature do not consider electricity availability and renewable energy in their models. Integrating renewable energy into data centers significantly reduces the traditional energy consumption and carbon footprint of these energy-hungry infrastructures. As renewable energy is intermittent and fluctuates with time-varying, it is usually under-utilized. We address the problem of improving the utilization of renewable energy for a single data center and investigate two approaches: opportunistic scheduling and energy storage. Our results demonstrate that both approaches are able to reduce the brown energy consumption under different configurations. We extend this work to the context of Edge Clouds and Internet of Things on the use case of data stream analysis. We show how to make Edge Clouds greener with on-site renewable energy production combined with energy storage and performance degradation of the users' applications.

Key Words
Cloud, renewable energy, energy efficiency, resource allocation.