Green energy efficient scheduling management

Inès de Courchelle, Tom Guérout, Georges da Costa, Thierry Monteil, Yann Labit

Version of Record: https://www.sciencedirect.com/science/article/pii/S1569190X18301382
Manuscript_e0e2159d02f99f434a70022a32e4e6a0

Green Energy efficient scheduling management

Inès De Courchelle[a,b], Tom Guérout[a], Georges Da Costa[b], Thierry Monteil[a], Yann Labit[a]

[a] *LAAS-CNRS, Toulouse University, CNRS, INSA, UPS, Toulouse, France*
[b] *IRIT Toulouse University, 118 Route de Narbonne, F-31062 Toulouse Cedex 9, France*

Abstract

The analysis of the energy efficiency in Cloud Computing infrastructures has become an important research domain as the utilization rate of the various on-demand services is daily higher and higher and its management is now considered as a main objective. Today, to tackle this challenging issue, Cloud providers integrate renewable energy sources to feed their infrastructure. Energy saving is part often an integral many companies goal. Unlike the classic supply of grid energy, the production of green energy is unstable and depends on nature of the weather or wind. It introduces new challenges as pervasive jobs to reduce a server consumption. In this article, studies based on the use and the storage of photovoltaic energy are exposed. We detail our design of a scheduler which uses solar energy production to make an off-line decision. This enables us to schedule virtual machines into a datacenter via different algorithms which consumes the least amount of brown energy as possible. We based our analysis through an existing workload from Google. We describe and study this workload to create one corresponding to our need. We also proposed to evaluate the storage size of a smartgrid related to the solar panel size. It is an analysis of the reliance between both storage (battery) and renewable energy production (solar panel) components sizing.

Keywords: Simulation, Energy-efficiency, Renewable energy, Scheduler

1. Introduction

Currently, Cloud Computing paradigm is facing major challenges which have arisen with the continuous development of on-demand services. The actual variety of these Cloud services constantly evolved to satisfy the users' expectations. This evolution also leads to an expansion of these platforms with high-performance computing resources. From an energy point of view, this heavy use of computing resources raises many complex problematic. Indeed, due to the present ecoloogical consideration, actual Cloud provider have to pay attention on how to feed their infrastructures. Actions that go in this direction are developing more an more. Particularly with regard to the high use of brown energy sources that give way to other greener and less polluting energy production solutions for the environment. Many companies focus on renewable energy to reduce their electricity usage and their carbon footprint. From 2010 to 2014, the energy consumption by datacenters has increased by about 4 percent in the United States [1]. For example, Google has invested 2.5 billion dollars into renewable energy projects in order to reach their goal of 100% reliance on renewable energy in 2017 [2]. At present, Google has datacenters supplied by the photovoltaic plant in the Atacama Desert, the largest of its kind in Latin America since january 2017 [3]. In order to reduce their footprint carbon many works have used the cost price to decrease the purchase of brown energy [1]. However, the use of multiple energy sources introduces new problems related to the energy storage. Many works exist based on the different storage equipment of their load/unload models and their usage in a datacenter [2] [3]. This additional storage form may be considered a safety measure in case of a shortage. Another example of a problem is the difficulty in relying on solar panels as they are complex and unpredictable based on their dependency on the weather variations.

Email addresses: dacosta@irit.fr (Inès De Courchelle), tguerout@laas.fr (Tom Guérout), dacosta@irit.fr (Georges Da Costa), monteil@laas.fr (Thierry Monteil), ylabit@laas.fr (Yann Labit)

[1] https://eta.lbl.gov/publications/united-states-data-center-energy-usag

[2] https://environment.google/approach/#/intro/infographics-1

[3] http://www.acciona-energia.com/pressroom/news/2017/january/acciona-covers-googles-electricity-consumption-chile-renewable-energy/

In this article, a virtual machine (VM) assignment problem is studied, being aware about the global IT context (computing & electrical flow), taking into account several real input parameters (real workload, solar panel production and storage capacities), different scheduling algorithms and the computation of various metrics. Each approach proposed is based on the use of a decision matrix which allows various energy strategies aiming to make the best use of the green energy production.

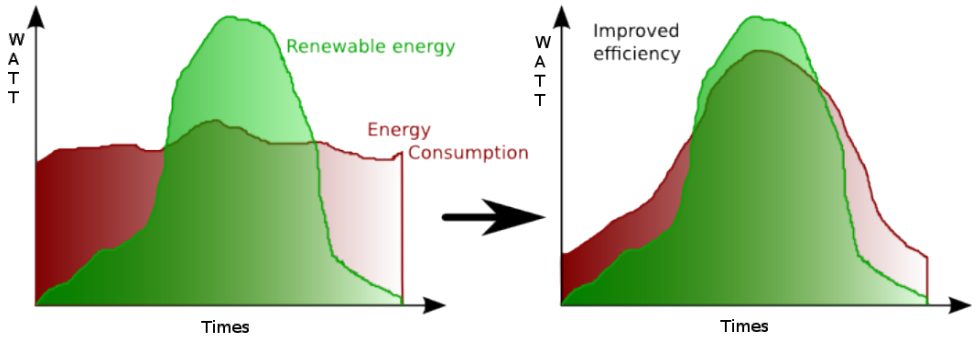


Figure 1: An example of agile workload which followed the solar panel production curve closely

For example, one strategy could be to have a correlation between the workload, that has be to executed by the VM, and the solar panel production aimed at reducing the usage of the electric grid and having a datacenter be almost fully self-sufficient (Figure 1).

The challenges we face include the intrinsic constraints for Virtual Machines (VM) scheduling and energy constraints. This, while satisfying the users' requests, and optimizing the servers to make them more profitable according to the time, the energy available and the potential load. Moreover, renewable energy sources can be used in a more productive way by using any storage capacities (batteries for example) and selling the substantial green overproduction. While taking into account those levers during the virtual scheduling process, leading to make the algorithms completion more complex, it also should allow to achieve smarter energy efficient goals like reducing the global consumption and thus-purchasing costs of brown energy sources.

The main contributions of this article are as follows:

- A green scheduler which aims to maximize the use of the solar panel production,
- An analysis on the battery and the solar panel sizing in order to better understand the behavior of the green energy usage,
- An input usage of a real workload, Google Traces [4], into a simulation tool, RenewSim,
- An impact analysis of different scheduling policies on a set of selected metrics.

The article is organized as follows: Section 2 overviews green datacenter supplied by renewable energy and schedulers using green energy in order to manage a workload and works on Google Traces dataset. Section 3 describes energy model and the scheduler strategy, Section 4 presents the methodology used and evaluates the scheduler strategy. Section 6 highlights our results for the algorithm adaptation. Section 7 is the analyze on the battery and the solar panel size, to understand the behavior of the green energy use. Section 8 is the balance sheet of this article and the future works.

2. Related work

Over the past ten years, multiple scientific articles have investigated multiples ways to reduce datacenter energy consumption. Different approaches have been studied to reduce the processor frequency in order to save energy like DVFS, or have tested using smart scheduling algorithm to allocate tasks on different servers. These attempts were made possible by using scheduler, genetic algorithm, or smartgrid feeds through several green energies systems including solar panels, wind turbines and storage components (Batteries). To tackle the main research domains related to this article, the following subsections describe some works on Cloud energy simulation, usage of renewable energies due to the localization and the use of green energy in workload scheduling and more importantly completing tasks using Google Traces dataset.

2.1. Cloud simulator for energy efficiency studies

Many works have been exploring the ability of scheduling distributed systems through simulator in order to reduce the datacenters consumption. Simgrid [5] is a simulator for distributed applications heterogeneous in distributed environments. It allows the execution of algorithms for scheduling a datacenter [6], the simulation of storage in a datacenter [7] and the use of the DVFS (Dynamic Voltage and Frequency Scaling) in a datacenter [8]. This simulator allows for the improvement of the actual application in the grid field. In CloudSim[9], which is based on GridSim[10], is a cloud computing simulator of a type of infrastructure as a Service (IaaS). The architecture of the datacenter is finely modeled and can be easily modified by the user. It integrates the virtualization (migration) of the addition of investment policies as well as the energy costs and operations that they incur. Another version proposed by Guerout et al [11], also integrates the DVFS. However, this simulator is purely dedicated to the internal management of the datacenter. This means Guerout et al aims at reducing the consumption of a datacenter but they do not take into account the type of energy used, whether it is brown or green energy.

2.2. Geographically distributed data center architectures to exploit the temporal and regional diversity of electricity price

Many works have focused on the shift in time and the migration of workloads use renewable energy in a better way such as [12]. Zhang et al. [13] proposed a middleware system called GreenWare. It aims at increasing the percentage of renewable energy used to power a network of distributed datacenters. The renewable energy used in this system are wind power and solar power. In this article, the cloud is composed as a set of datacenter distributed geographically. They studied services to minimize the carbon footprint of certain requests within a fixed budget cost by the Internet service operator. In [14], they used multiple geographically distributed mini datacenters which they could adjust the workload to where there were more available renewable sources. Liu et al. [15] proposed three algorithms to reach an optimal geographical load balancing, to increase the use of green energy and reduce the use of brown energy.

2.3. Green energy use to schedule workloads

The researches have been investigating how to best exploit renewable energy for scheduling a workload. Gori et al [16] proposed GreenSlot which allows the simulation of solar panels to schedule a batch set of tasks into a datacenter. It also allows the use of several types of energy green and brown. Although GreenSlot takes into consideration the predicted reliance on green energy in the near future, the use of grid energy assists in avoiding deadline violations. Gori et al proved that GreenSlot can increase green energy consumption while decreasing the energy costs by up to 39%. Gori et al [17] also proposed GreenHadoop which is an evolution of GreenSlot. It is a MapReduce framework for a datacenter powered by solar energy and similar to GreenSlot has the electrical grid as a backup. GreenHadoop contrary to GreenSlot does not require job behavior information and can manage data availability while keeping brown energy costs low. GreenHadoop incorporates the principle of dynamic replication of data to ensure the availability of the latter. A module of the prediction of energy production is present with how the charging of the batteries is managed and incorporates five levels of priorities of jobs (Very High, High, Normal, Low, and very low). GreenHadoop behaves differently as Hadoop. This green framework uses servers in three ways : by using as many servers as green energy can supply, by unsing fewer servers when brown energy is cheap, and even fewer when brown energy is expensive. The Algorithm scheduling works with two queues for jobs : Run and Wait. Both queues are used in FIFO order. Aksanli et al. [18] design an adaptive datacenter job scheduler supply by green energy. The scheduling process aims at using short term prediction of solar and wind energy production. Deng et al. [19] proposed an online control algorithm for a datacenter Power Supply System (DPSS) with multiple sources : MultiGreen. They use a smartgrid supply by green or renewable energy and the electricity grid. It allows the Cloud Service Providers to make online decisions on purchasing grid energy. They also use the UPS System as an energy storage. The online algorithm aims at finding a near optimal solution without a knowledge of power demand and renewable energy generation by reducing the operational cost in the datacenter's long-run operation. In [20], they provide a modular Energy-Aware computing Framework. EACOF brings a layer abstraction between sources of energy data and the applications that exploit them. It allows to measure and evaluate a software energy consumption. However, this work is more about the energy aware problematic in portable system.

2.4. Google workload

Several studies have previously focused on a better understanding of the Google Traces dataset and of the characteristics of the described tasks. In [21], authors described the heterogeneity of the hardware resources in the Google traces. They classified priorities in five categories Infrastructure(11), Monitoring(10), Normal production (9), other (2-8) and Gratis(0-1). In this analyze, they shown that 2% of tasks represent 80% of CPU, Memory and that 92% are long tasks with a *Free* priority. In [22], Reiss *et al.* analyzed the Google cluster performance. According to this article, many long jobs have stable resource utilization, which helps the adaptive resource schedulers. They concluded that machine configuration and workload composition are heterogeneous. This analyzed trace allows us to understand the Google dataset, the importance of an heterogeneous datacenter to adapt the resources to the demand. However, they did not provided information related on how to use this workload in another context. Some other articles [23, 24] have clustered jobs with a k-mean to classify the size of each each jobs in different categories. The article [25] evaluates the gap between the requested resources and the one consumed by tasks within the datacenter. The requested average load for a processor is 10% on the Google datacenter. This Google trace analysis shows that processors are overall under utilized which leads to an increase of the energy consumed (90% of available processor computing power is not used). Moreover, most of the workload has a low priority and is not sensitive to the latency. It shows that there is only a very few number of sensitive tasks (scheduling class: 2 or 3). In this article, authors focus on the lack of energy consideration in the Google trace. Di *et al.* [26] evaluated the Google trace compared to other Grid/HPC systems and in [27] they have observed that, for the Google workload, the resource utilization per application (*logic job name*) follows a typical Pareto principle.[22], [28] shown that the machines are almost homogeneous, 93% machines have the same CPU capacities and 86% of the machines have the same memories capacities.

2.5. Our differences

To compare the approach proposed in this article to those described above, we go though specific and major characteristics of this work, in terms of Electric & IT (Information Technologies) domains, management and usage of green energy source and comparison of few virtual machines scheduling algorithms, which are the following:

- Both Electricity and IT problems are tackled through different proposition about the management, the storage and the utilization of green energy production. To do so, few pure scheduling approaches of IT workloads, as basic as they are, are investigated in this complex context.
- Studies and simulations are proposed using RenewSim which is based on an architecture mixing both complex domains
- Categorization of the different elements of our architecture: Source, Destination, Backup and Overflow. A definition of each of these terms and how and why we are going to use them, is also clearly noted.
- Studies proposed are based on a central decision system able to take online decision to schedule virtual machines on servers.
- The virtual machines scheduling focused on the best possible use of green energy, leading to a smart management of the green energy flow given from the photovoltaic panels.
- A section is dedicated to a detailed study of the Google Traces dataset in order to understand and characterize this Cloud input. This part of the work then allow us to proposed experimental phases based on a clear analysis of this workload.
- The workload used came from the Google Traces dataset, including more precise parameters which have been analyzed and used to compare the different virtual machines scheduling approaches.

The next section focuses on the context and objectives of the scheduling approach proposed.

3. Scheduling specification

In this article, one of the main goals is to outline different solutions. Firstly, it is about how to manage the use of both energy production and computing resources. Secondly, it is to analyze their impact on various parameters. It is assumed that the computing resources (datacenter) are simultaneously fed by renewable and non-renewable energy sources. This section summarizes the used methodology.

3.1. Context

The energy efficient scheduling issue for such a datacenter comes from the use of green energy as the production depends on variable nature of the weather and wind. The figure 2 presents an architecture example of several sources and destinations of the considered smartgrid. It is possible to plug in several sources and destinations. The main source comes from the solar panels production and the main destination is the datacenter. Each element plugged to the bus is connected to a converter. Those are modules for the conversion of DC/DC (Direct Current to Direct Current) or DC/AC (Direct Current to Alternative Current). This grid is then supervised by a control module. Its role is to find the best possible configuration for both to manage the energy flow (feeding the datacenter) and to schedule tasks over the computing resources. This involves managing the power: the power sold from the solar panels to the suppliers, the power bought or sold to the supplier based on its cost and also the loading/unloading of the battery. The allocation of tasks depends on their characteristics (CPU and memory capacity, priority, preemptive or not) and the total amount of available energy.

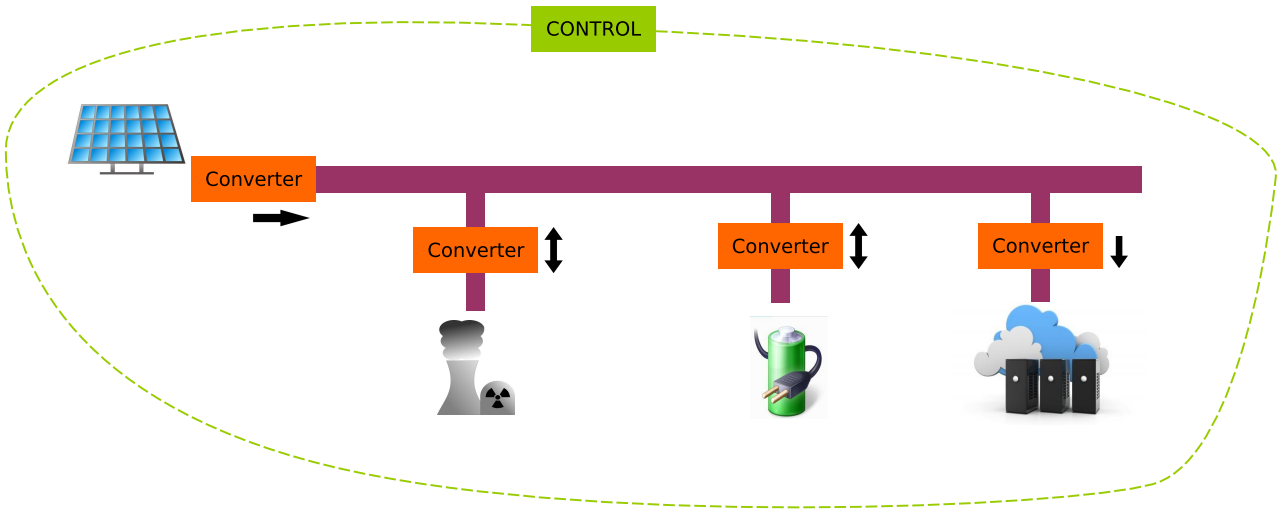


Figure 2: Global considered smartgrid architecture: The feeding of the datacenter via several sources is managed through a dedicated control module

3.2. Objectives

Our main objective is to provide a more efficient datacenter with green energy through a scheduler based on a production prediction.

This datacenter is powered simultaneously by renewable and non-renewable energy sources. The scheduling problem to supply a datacenter is the use of green energy via solar panels, whose production depends on nature, which varies according to meteorology.

The main source comes from the production of solar panels and the main destination is the datacenter. This intelligent network is supervised by a control module. The role is to find the best possible configuration to manage the energy flow and the workload requested by the computer center. This involves managing:

- The energy sold from the solar panels to the suppliers
- The energy purchased from the supplier based on its cost
- The load and unload of the battery

The goal is to equip the control module with a scheduler in order to place the workload over time. This workload consists of a set of VMs. The VMs assignment depends on their characteristics (CPU and memory capacity, priority, preemptive or not) and on the total amount of energy available.

The scheduling algorithms should allow for the datacenter:

- To be the least dependent on the electricity supplier
- To decrease the ecological footprint
- To reduce electricity costs

3.3. Functionalities

The algorithms must respect several elements of methods or constraints.

- The available VCPU capacity of the different servers composing the datacenter must be respected. This is a maximum acceptable number of VM per host.
- We must respect the prediction of solar production. A VM is allocated on a server when it can be powered via solar energy.
- Those algorithms must be able to propose results and simulations using RenewSim [29]. This simulator is based on a smartgrid architecture that mixes the two complex domains, energy flows and IT flows.
- The algorithms must be measurable via metrics evaluating the QoS and the ecological footprint. This metrics will help us compare and evaluate each algorithm.

4. Our scheduling approach

The algorithms presented in this section are taken from the literature. They have been adapted to our needs and their objectives are to be implemented in the RenewSim simulator [29]. More particularly, it is considered as a scheduler in the control module. This scheduler has access to the various information and flows that circulate in the intelligent network. It does not only make decisions based on the datacenter related activities, but also on available energy supplies. It takes into account the production of renewable energy, and the quantity available in the storage spaces.

4.1. The energy flow control

To take a faster decision, each element of the grid (figure 2) except the decision module, is described by a model [29]. This model aims at having an energy strategy. It is a data modeling to handle the power flow in a smartgrid.

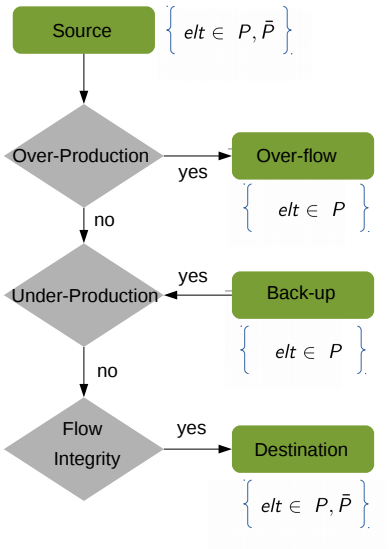


Figure 3: System states and contexts

The figure 3 presents all the states and contexts of the smartgrid. This model allows to characterize each element (the battery, the provider, the solar panel, the datacenter, the wind turbines ...) of the smartgrid in four states:

- *Source*: The main energy suppliers to feed the *Destinations*
- *Destination*: The main destination feeding by the mains energy suppliers (*Sources*)
- *Backup*: The backup energy when the (*Sources*) cannot feed the *Destinations*
- *Overflow*: The elements which are allowed to receive energy when the *Destinations* cannot handled all the energy from the *Sources*

There is two types of element :

- *Not manageable* $\rightarrow \bar{P}$: These elements are the sources of renewable energy and the workload requested by the computer center. For example, solar production does not depend on the system. The production of solar panels depends on the weather and the time of day. In addition, the datacenter must respond to a request that does not depend on the system.
- *Manageable* $\rightarrow P$: These elements are the storage space and the electricity supplier. For example, the price of the supplier depends on the time: during the night electricity is cheaper than during the day. This is why we can manage it and apply day or night policies.

The details of the three contexts defined as follow:
Under-production happens when the production of *sources* elements is less than the request of *destinations*. This situation is presented by the equation 1, with $\sum S_t$ the sum of all *sources* available at the moment t, and $\sum D_t$ the sum of all *destinations* demand at the moment t.

$$\sum S_t < \sum D_t \tag{1}$$

The *backup* elements allow to balance the equation 1, and to offer to the *destinations* elements the desired energy. This is represented via the equation 2, with $\sum backup_t$ the sum of all *backup* at a time t.

$$\sum S_t + \sum backup_t = \sum D_t \tag{2}$$

Over-production happens when the production of *sources* elements is above the request of *destinations*. This situation is presented by the equation 3, with $\sum S_t$ the sum of all *sources* available at the moment t, and $\sum D_t$ the sum of all *destinations* demand at the moment t.

$$\sum S_t > \sum D_t \tag{3}$$

The *overflow* elements allow to balance the equation 3, and to store the overflow of energy. This is represented via the equation 4, with $\sum backup_t$ the sum of all *overflow* at a time t.

$$\sum S_t = \sum D_t + \sum over\_flow_t \tag{4}$$

Flow integrity is reached out when the *destinations* and *sources* elements follow the equation 5. To guarantee the equation 5 in case of over-and under-production, the equation 6 adds $\sum backup_t$, the sum of all the *backup* available at the moment t (equation 2), and $\sum backup_t$ the sum of the *overflow* available (equation 4).

$$\sum S_t = \sum D_t \tag{5}$$

$$\sum S_t + \sum backup_t = \sum D_t + \sum over\_flow_t \tag{6}$$

The table 1 presents all the strategic possibilities.

In this model of flow representation, renewable energies cannot be *destinations*, because they cannot store energy. For the computer center, it can only be *destination* because it cannot produce energy or store energy. The other elements, storage spaces (1 & 2) and provider can be of any type because they are manageable. However, we must succeed in organizing the different priorities to avoid conflicts between them. There are no priorities for *sources* and *destinations*, because it is the sum of these which defines the cases of overproduction and underproduction.

Table 1: Set available values for each element & category

Types / Element	Source	Destination	Backup	Overflow
Renewable energy 1	$\{\times, 1\}$	$\times$	$\{\times, 1, 2, ...\}$	$\times$
Renewable energy 2	$\{\times, 1\}$	$\times$	$\{\times, 1, 2, ...\}$	$\times$
datacenter	$\times$	$\{\times, 1\}$	$\times$	$\times$
Storage 1	$\{\times, 1\}$	$\{\times, 1\}$	$\{\times, 1, 2, ...\}$	$\{\times, 1, 2, ...\}$
Storage 2	$\{\times, 1\}$	$\{\times, 1\}$	$\{\times, 1, 2, ...\}$	$\{\times, 1, 2, ...\}$
Supplier	$\{\times, 1\}$	$\{\times, 1\}$	$\{\times, 1, 2, ...\}$	$\{\times, 1, 2, ...\}$
...	$\{\times, 1\}$	$\{\times, 1\}$	$\{\times, 1, 2, ...\}$	$\{\times, 1, 2, ...\}$
Element n	$\{\times, 1\}$	$\{\times, 1\}$	$\{\times, 1, 2, ...\}$	$\{\times, 1, 2, ...\}$

4.2. The scheduling strategy

Our approach consists in scheduling the VMs over time in terms of the energy production curve of the solar panels. To have an energy strategy, the approach uses the model presented in the table 1. Our scheduling strategy use the Table 2 as data model. The demand and the renewable energy generation are random variables. In order not to make the constraints too complex at first, we are interested in the use of energy as in Figure 4. Our scheduling strategy uses the table 2 as the data model. This table 2 shows the order of use of the different energy sources. Solar panels provide the datacenter as sources. In case of underproduction, the battery (Backup = 1) supplies the datacenter. If the battery is empty, the electrical supplier supplies the datacenter (Backup = 2). In case of overproduction, the energy is stored in the battery (Overflow = 1). If the battery is full, the energy is sold to the electricity supplier (overflow = 2). This model aims to buy the least amount of energy from the supplier. Our strategy can be represented by the figure 4.

Table 2: Energy model for scheduling strategy

	Source	Destination	Backup	Overflow
Solar panel	1	$\times$	0	$\times$
Datacenter	$\times$	1	$\times$	$\times$
Storage	0	0	1	1
Supplier	0	0	2	2

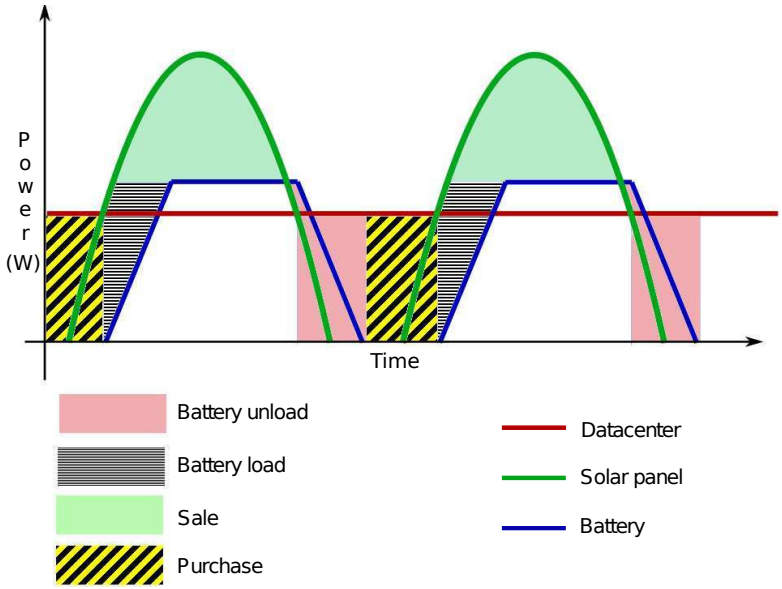


Figure 4: The different use of energy sources during 48 hours

The figure 5 is an example in an almost perfect situation. The VMs are allocated according to the time, the energy available in the solar production and the space available in the servers. The storage is not a constraint of scheduling algorithms but it is part of the scheduling strategy of energy flow management. This is a backup solution, only if the production of solar panels is not enough to power the datacenter.

The scheduling constraints of the VMs are:

- The use of heterogeneous servers in terms of VCPU
- The placement of heterogeneous VMs in terms of execution time
- The number of VCPUs used by VM
- The energy available from the production of solar panels

The advantage of this method is the maximum use of the production of solar panels and the reduction of the purchase of energy from the electricity supplier. A VM is only scheduled if there is sufficient energy from the solar panels. However, the VM continue to operate even if the solar energy is null. For example, in the figure 5, the *Vm 10* starts running when there is enough solar energy on the *S3* server. However, this VM continues to run when solar production decreases.

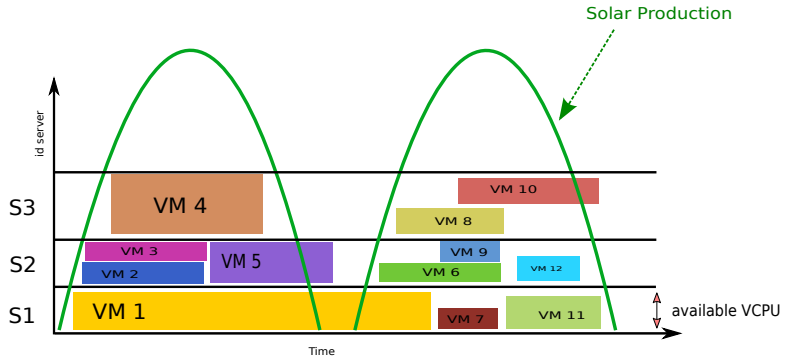


Figure 5: Example of VM scheduling regarding the PV production

4.3. The use of greedy algorithms

We describe a set of greedy algorithms [30, 31]. The advantages of greedy algorithms are the creation of simple and fast heuristic. The decisions made by the algorithm are taken step by step without ever reverting to the previous decision. In the algorithms, decision-making is rapid. The choice of use was based on greedy algorithms because a decision must emerge quickly, in order to satisfy as quickly as possible the user's request. The use of green energy is not sustainable over time. We insist on the advantage of the greedy algorithms. They take a step-by-step decision without ever going back on it.

4.4. Green scheduler algorithms

In this section, several scheduling solutions are presented and analyzed. For each of the following algorithms, the resource constraints of different servers and the solar energy available over time are always taken into account. Each of the algorithms own a set of independent VMs, stored in a queuing list. The figure 6 presents the different inputs and outputs of the algorithms described in this part.

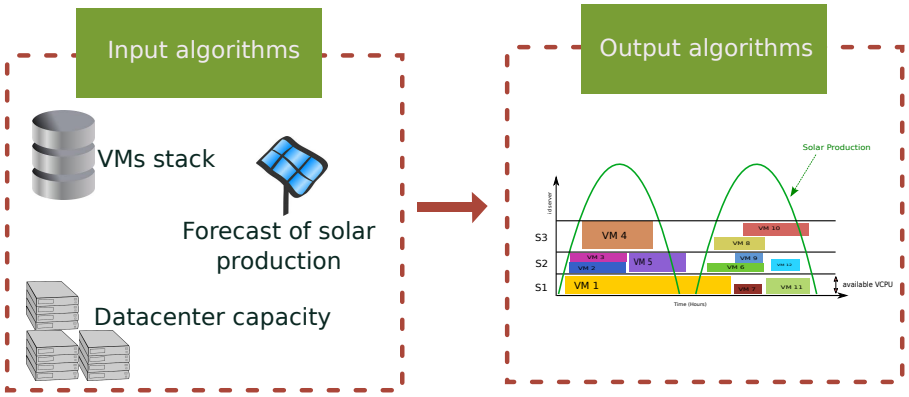


Figure 6: Inputs and outputs of the different scheduling algorithms

4.4.1. Round Robin (RR)

The round robin algorithm (Algorithm RR) allows to place a VM on a server and then once the VM is placed, the next VM will be placed on the next server and so on. In this case, if a VM can not be placed on a server, the next one is selected. If it can not be placed on one of the datacenter servers, it is ejected and the next VM is selected. The load of the servers is more or less balanced over time. The disadvantage is that there is no optimization of the computer park in terms of servers, there is an under-utilization of the resources. The purpose of this algorithm is to serve as a reference.

4.4.2. First Fit (FF)

The first fit algorithm (Algorithm FF) is to place the different VMs on a server. Once it is full, the operation is repeated on the next server. The disadvantage is the load of the different servers. The first servers will have a higher load than the last servers because the available solar energy will have greatly decrease by the scheduling already made. The purpose of this algorithm is to serve as a reference.

4.4.3. Round Robin with server classes (RR_CLASSES)

The algorithm Round Robin with server classes (Algorithm RR_CLASSES) is the algorithm (RR) except that it contains server classes. A set of servers is dedicated to schedule VMs based on executing time (short, medium or long). For example, 10 % of servers schedule long VMs, 40 % medium VMs, and 50 % short VMs. This division into classes, allows to distribute a set of VMs, and to schedule more VMs in time. This allows the long VMs to more easily obtain VCPU resources that can be used by VMs of shorter duration and hence prevent the scheduling of some more demanding VMs. The disadvantage of this algorithm is the optimization of the various resources of the datacenter. For example, a long VM is ejected because there is no longer Hardware resources on the servers reserved for the long one. This situation appears even if there is enough hardware resources on a server to schedule short or medium VMs. The server classes must be implemented according to the number of different VMs to be executed, in order to find the best distribution of server classes.

Algorithm 1 RR_CLASSES

```
while VM < nbVmsScheduled do
  for time=0;time<schedulingWindow;time++ do
    if solarNrj[time] == true and servers[s][classe][time] == true then
      VM ⇒ server[s][time]
      servers[classe][s + 1]
    end if
  end for
end while
```

4.4.4. Longest Processing Time with Round Robin (RR_LPT)

The algorithm Longest Processing Time with Round Robin (Algorithm RR_LPT) consists of scheduling a VM on a server, then once the VM is placed, the next VM will be placed on the next server and so on. This algorithm differs from the algorithm RR because the VMs are scheduling according to their size. The longest VMs are scheduled first.

Algorithm 2 RR_LPT

```
Require: list vms order per completion time (longest to shortest)
while VM < nbVmsScheduled do
  for time=0;time<schedulingWindow;time++ do
    if solarNrj[time] == true and servers[s][time] == true then
      VM ⇒ server[s][time]
      servers[s + 1]
    end if
  end for
end while
```

4.4.5. Longest Processing Time with First Fit (FF_LPT)

The algorithm Longest Processing Time with First Fit (Algorithm FF_LPT) consists of placing the different VMs on the next server. Once it is full, the operation is repeated on the server. This algorithm differs from the algorithm FF because the VMs are placed according to their size. The longest VMs are placed first.

Algorithm 3 FF_LPT

```
Require: list vms order per completion time (longest to shortest)
while VM < nbVmsScheduled do
  for time=0;time<schedulingWindow;time++ do
    if solar_nrj[time] == true and servers[s][time] == true then
      VM ⇒ server[s][time]
      if serverFull == true then
        servers[s + 1]
      end if
    end if
  end for
end while
```

These different algorithms have the same types of inputs in order to make them comparable and measurable between them.

4.5. Virtual Machine model based on Google Traces Attributes

To evaluate all the algorithm, we used the google traces [4]. Google provided a dataset of its servers usage in 2011. The monitored datacenter is composed by more than 12,000 servers with heterogeneous characteristics and 25 millions tasks. The different statistics were collected for a period of 29 days. In order to use the traces

proposed by Google in the simulator, a cleaning was imposed. We were interested in the data stored in the "task _events" directory. During the period of 29 days, they record 100 millions events which describe each task and their life cycle. An event indicates a transition between states (7). In [32] [33], they show that 95% of the task has only three events. In order to clean the traces, we keep only the tasks with three events, an initial one "Submit", a "Schedule" one and an ending event ("Finish","Fail","Kill", "Lost" or "Evict").

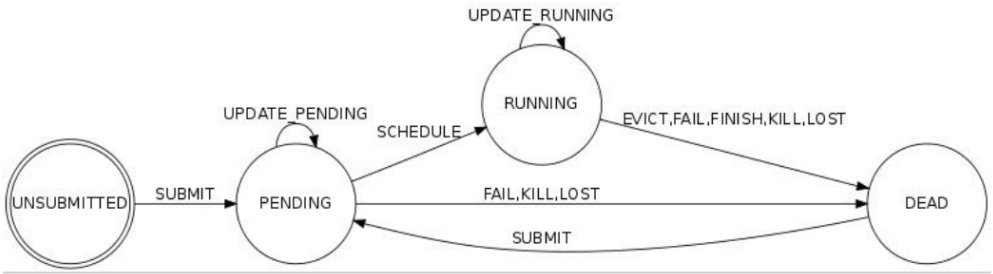


Figure 7: Google task transition states [4]

Table 3: Distribution of ending event for each tasks with three events[32]

Events type	**Finish**	**Kill**	**Fail**	**Other**
Number of tasks	17,775,284	6,381,906	86,348	15,369
Percentage	73.31%	26.31%	0.35%	0.03%

The Table 3 shows that more than 73.31% of the tasks end up with a "Finish" against approximately 26.31% by a "Kill" event and 0.35% by an "Fail". The VMs used are the Google tasks with the following ending events: "Finish", "Kill" and "Fail". We keep only the task finishing with this three events because they are more presented in the Google Tasks. It allows us to have VMs with heterogeneous characteristics in terms of execution time because most of the "kill" tasks have an execution time above the average. The CSV files allow us to follow the lifetime of a task at a given time t. There is no data structure for each tasks during the collected period. In order to replay the Google data in our simulator, we have used some features of Google Tasks and created a data structure. We need to places VMs on physical machines. A VM is defined as follows:

Table 4: The VMs model: description based on the Google dataset

name	Description
ide	A unique ID which allows to follow the VM evolution
submit_time	The arrival time of a VM into the VM stack to execute, this time allows to know when a VM is eligible to be scheduled
schedule_time	The time when the VM is scheduled and is executed
ending_time	The time when the VM is terminated
event	The ending event: "fail" or "finish" or "kill"
priority	The priority (between 0 and 11)
latency	Its sensitivity to latency (between 0 and 3)
vCPU	The vCPU demands between (1 and 4)

In Table 4, the *schedule_time* and *ending_time* attributes allow us to calculate the "execution time". The *ide* attribute is a unique identifier obtained using the concatenation of two values of Google attributes [4].

Then, the *submit_time*, *schedule_time* and *ending_time* correspond to the time where the event is changing. The *event* is the ending event of the VMs which corresponds to an event from the Google dataset. This event could be a "fail" or a "finish" or a "kill". We keep only the Google task with Three events, in [32] because they analyzed that 95% tasks have only three events. Then, the *priority*, and the *Latency* correspond respectively to the characteristics of Google (presented in [4]). We add the characteristic "nb_vcpu" in order to calculate the consumption of a server.

5. Evaluation metrics

In order to evaluate and compare the different algorithms, this section introduces a set of metrics which will be used in the following. In a first part metrics are introduced to evaluate the green consumption. In a second part, metrics are presented to evaluate our QoS of the scheduling.

5.1. Green metrics

There are several green measurements to measure how green a datacenter is. The Power Utilization Efficiency (PUE) [34] is a measure for determining the energy efficiency of a datacenter. This is the most popular method for calculating energy efficiency. This involves assessing the amount of total energy consumed by a site over one year, relative to the amount of energy required to operate the datacenter equipment. However, the work presented in this article does not focus on cooling technologies in a datacenter. The green energy coefficient (GEC) [35] is used to quantify the share of renewable energy consumed by a datacenter. Like the PUE, it is introduced by the Green Grid Organization in November 2012 [36]. In order to overcome the drawbacks of the PUE, we decided to use the GEC, because the PUE does not take into account certain elements for the datacenter: such as the location or the load rate of each individual components. the GEC is the ratio of the amount of electricity consumed from renewable sources by the total consumption of the datacenter. A GEC formulation is proposed in equation 7 in our case, where $Prod_{solar}$ all the green energy generated in the datacenter, $Prod_{sell}$ all green energy not used and sold to the electricity supplier, and $Supplier_{purchase}$ all energy purchased from the supplier.

Proposition 1 The GEC :

$$0 \leqslant GEC = \frac{P_{solar} - P_{sell}}{S_{purchase} + (P_{solar} - P_{purchase})} \leqslant 1 \tag{7}$$

We choose to withdraw the sale of the green energy ($Prod_{sell}$) because it never provides the datacenter. This is not part of the amount of energy required for the operation of the computer hardware. If the GEC result is closer to 1, it means a greater use of green energy and closer to 0, the architecture uses more non-renewable energy than green energy.

5.2. QoS metrics

Several works have addressed the definition of QoS in a distributed environment [37] as well as QoS related to energy [11, 38]. A Datacenter must be capable of delivering a level of performance for users. A set of metrics can be used to evaluate quality in a cloud environment. Some metrics, such as those presented in [39], are used to evaluate the elasticity of a cloud, through the resources used, the provisioning cost and under-procurement cost. Rossi *et al.* [40] introduces *Orchestrator (e-eco)* to improve energy savings and application performance in a cloud. This indicator is also based on a set of metrics used to evaluate resources, consolidations, or VM virtualization. For example, through their simulation, SLA violations/saturation indicators are much lower ($\simeq$ 30 pts) compared to other methods such as Alvarruiz *et al.* [41].

To evaluate the QoS of the different algorithms presented, we choose to compare the rate of excluded and scheduled VMs. This aims to show the ability of the datacenter to meet user demand and the ability of the datacenter to adapt to the workload. These values are strongly linked to the quality felt by the customers. Furthermore, it could be related to the benefits that can be realized by the operator of the datacenter. Moreover, to compare the algorithms, we introduced a completion time indicator (Equation 8). It corresponds to the sum of the execution time (from its scheduling to its completion) for each scheduled VM and each excluded VM.

Proposition 2: The hourly quantity of ejected-scheduled VMs is written as:

$$\sum_{i=1}^{n}(comp_t^{vm_i}) \tag{8}$$

where n the number of excluded or scheduled VMs and $comp_t$ the end time of the VM vm_i.

This indicator helps understand the excluded VMs, to find out if there are many long or short VMs and its impact on the scheduler.

5.3. Example of metrics uses

The equation 9 is an example the GEC when it gets a good rate. If the solar panels produce 100 Joules in 48 hours and the sale to the supplier is equal to 10 Joules and the purchase is equal to 20 Joules, the GEC will be equal to 0.82.

$$\frac{100-10}{20+(100-10)} = \frac{90}{110} = 0.82 \tag{9}$$

The equation 10 is another example of the GEC when it gets a bad rate. If the solar panels produce 80 joules in 48 hours and the sale to the supplier is equal to 40 and the purchase is equal to 40 joules, the GEC will be equal to 0.50. In this case, there was more sale than the previous example 9. However, the GEC gets a lower rate because the purchase is more important. This can be explained by the period of energy use. If the VMs are scheduled when the solar production declines, they will continue to run when there is no more energy. Then, the battery unload will begin and once the battery is completely unloaded, the purchase at supplier will be realized. So the purpose of the algorithms is not to sell to the supplier and load as much as possible the battery. The main goal is to make the best use of the renewable energy to power the datacenter.

$$\frac{80-40}{40+(80-40)} = \frac{40}{80} = 0.50 \tag{10}$$

6. Algorithm experiments

In these experiments is evaluated the consumption of the datacenter, the use of renewable energies and the behavior of the VMs with the different algorithms described above in section 4.4. The indicators used are presented in the section 5. In these experiments, we highlight the correlation between the energy flows and the IT flows proposed in the section 4.2.

6.1. Experiments set up

The following experiments are the implementation of different algorithms. They are introduced in the control module in the RenewSim simulator [29] for purposes of obtaining results on energy efficiency and on QoS via a set of indicators. In order to correlate IT and energy flows, these experiments highlight the decision-making on heterogeneous objectives.
These algorithms schedule a set of VMs on servers and over time. Here, a set of VMs is given as input to the simulator. These correspond to synthetic traces generated from the Google data [4] . These synthetic traces follow the model proposed in the section 4.5.

For the following experiments, 1000 Vms (600 short time execution, 200 medium time execution and 200 long time execution) were used. There are more VMs short because Google has more short tasks than long. The short VMs have a working time of 3 minutes and the medium ones have a working time from 45 minutes to 2 hours. This allows us to keep a heterogeneity in the workload. The number of VMs was set to 1000 because this workload had to be adapted to the servers capacities. The datacenter is composed of several servers with heterogeneous characteristics.

The datacenter has five servers (table 5) with different characteristics. In order to analyze the behavior of rs algorithms adaptation, we decided to use a small datacenter composed with only 5 servers.

The server s4 has more vCPU (20 vCPU) than the others because we needed a server with more resources for long VMs. We chose a small and heterogeneous datacenter to understand the IT and energy flows. In addition, the number of five servers was chosen in order to obtain a correlation between the number of VMs and solar production for 48 hours. At the initial state of the experiment, all servers are off. They turn on for the first time as soon as a VM is scheduled. For the algorithm RR_CLASSES, we have used the following classes:

- s0 ⇒ short VMs
- s1 ⇒ medium VMs
- s2, s3, s4 ⇒ long VMs

There is more server for long VMs because they use the server vCPU for a longer period. Therefore, long VMs use more vCPU over time than shorter VMs (medium, short).

Table 5: Number of vCPU per server

s0	**s1**	**s2**	**s3**	**s4**
8 vCPU	8 vCPU	16 vCPU	16 vCPU	20 vCPU

$$C_s(t) = base + (Pmax_s - Pmin_s) \times (\frac{vcpu_{vm}}{vcpu\_total_s}) \tag{11}$$

The consumption is modelled by the equation 11. The equation represents the consumption at a given time t of a VM on a server, where:

- $base$ the power min ($Pmin_{ser}$) of the server if it is off and zero if it is on
- $vcpu_{vm}$ is the number of VCPU asked by the VM
- $vcpu\_total_{ser}$ the total of VCPU owned by the server
- $Pmax_{ser}$ the power max of the server

The idle power of each server is 80W and the maximum power is 169. This data is extracted from actual data and measured on DELL R630 servers with two CPUs (XEON).

The following experiments:

- Analyze the scheduling result of the algorithms (Section 6.2)

- Knowledge of the algorithm that consumes the least (Section 6.3)
- Identify the algorithms that eject the most VMs (Section 6.4)
- Pick out the algorithms using the least renewable energy thanks to the GEC indicator

All these experiments are related to one another and allowed us to:

- Understand the relationship between the number of VMs ejected and the total energy consumption
- Compare the share of green energy to total energy consumption

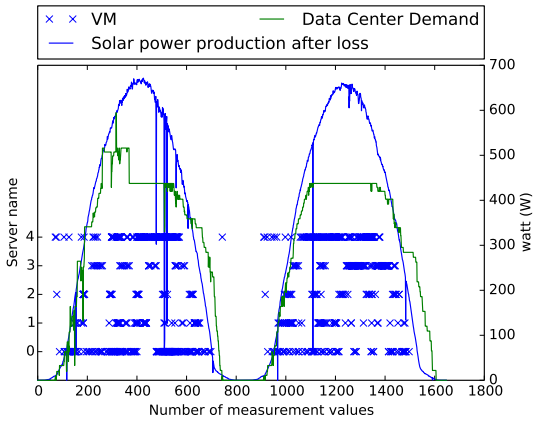
6.2. Scheduling results

In this section, we are presented the scheduling results, in terms of placement on the servers for each algorithm.

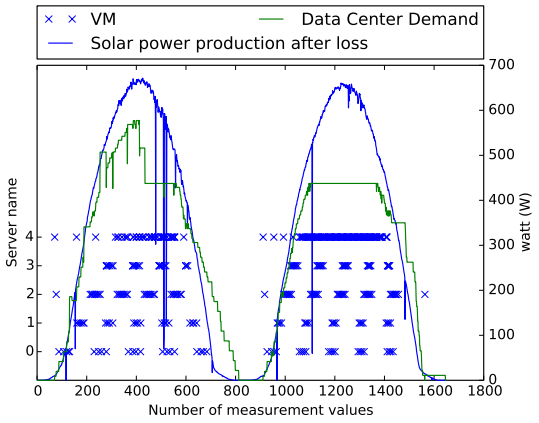
6.2.1. Analyze of the First Fit algorithms

The figures 8a and 8b have quite similar scheduling results. The difference is visible in the servers. The figures 9a and 9b expose the scheduling result for the *S0* server. Each yellow box is a VM.

In the FF_LPT algorithm, the longest VMs are placed first on the first server and then on the next server until the last one (*S4*). The scheduling gap of the VMs is greater in the first servers then decreases. However, we observe a gap for the server *S4*. The figures 10a and 10b highlight this phenomenon. This server *S4* is the last one of the algorithm where the VMs are scheduled. The available solar energy is, therefore less important because many other VMs are already running on other servers. Moreover, this scheduling gap, between the different VMs, describes the time when the solar panels produce the least energy at the beginning of the day, or at the end of the day. The first consumption peak, on each result is due to the ignition of the servers (equation 11).

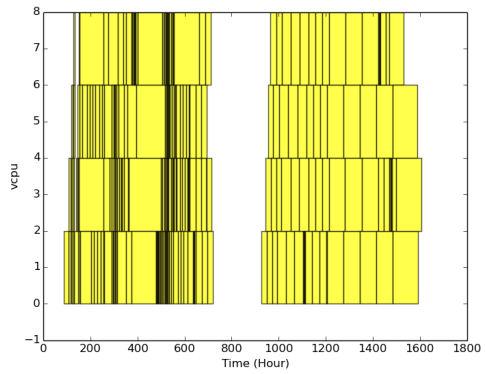


(a) VMs distribution with datacenter consumption and production of solar panels for algorithm FF

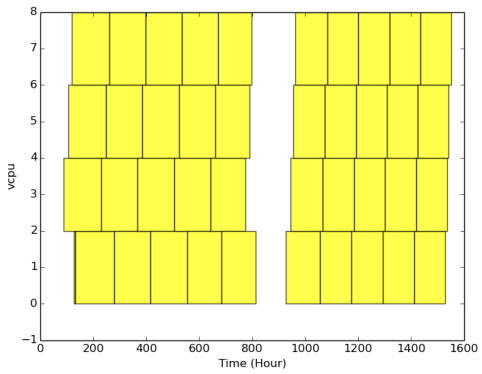


(b) VMs distribution with datacenter consumption and production of solar panels for algorithm FF_LPT

Figure 8: First Fit algorithms

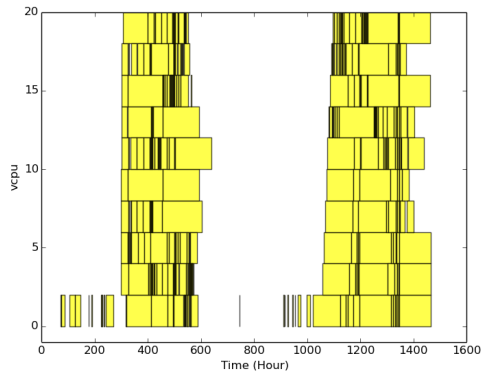


(a) Scheduling VMs on the server *S0* for the algorithm FF

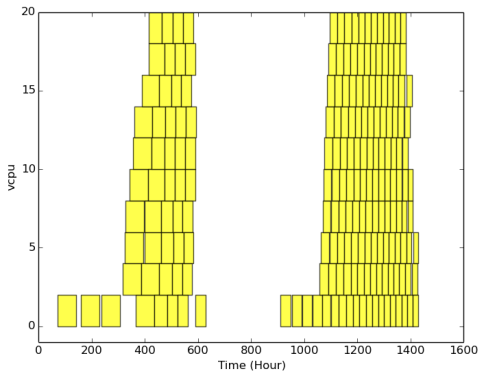


(b) Scheduling VMs on the server *S0* for the algorithm FF_LPT

Figure 9: Scheduling VMs on the server *S0* for the First Fit algorithm



(a) Scheduling VMs on the server *S4* for the algorithm FF

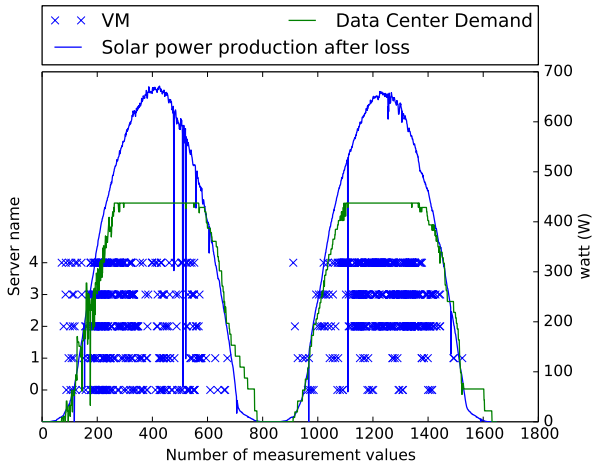


(b) Scheduling VMs on the server *S4* for the algorithm FF_LPT

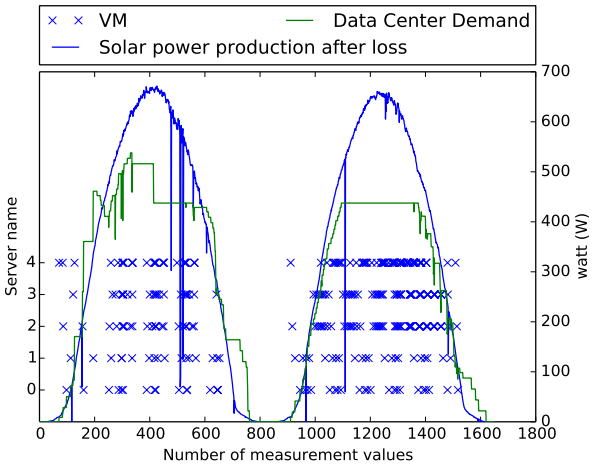
Figure 10: Scheduling VMs on the server *S4* for the First Fit algorithms

6.2.2. Analyze of the Round Robin algorithms

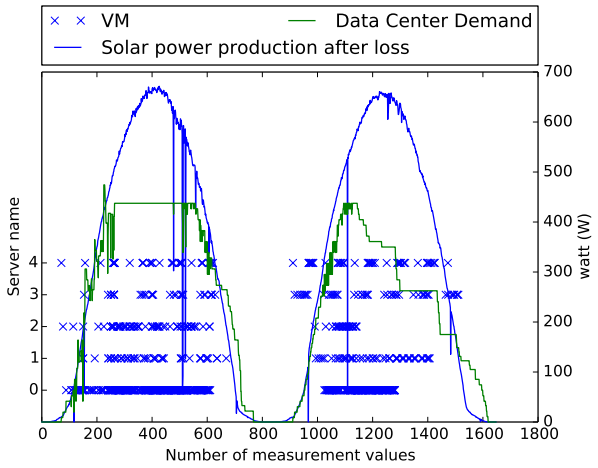
The figures 11a, 11b, and 11c, show the different results for the Round Robin algorithms. First, on the algorithm RR, the consumption peak (when the servers switch on) does not appear. This can be explained by the fact that the servers at the beginning of the experiment are not overloaded. Then, on the first morning, the algorithm RR_LPT (figure 11b), the consumption of the servers is higher than photovoltaic production. This is also true when photovoltaic production decreases. Each algorithm schedules the VMs when there is enough photovoltaic energy at a given time t. However, a VM continues to consume throughout its life cycle. Even if there is not enough photovoltaic energy at a given time t but VMs have started running at $t-1$, they will continue to consume even afterwards. For the algorithm RR_CLASSES, the servers *S0*, *S1* and *S2* are not busy until the end of the second day (time unit $\simeq$ 1300), these servers belong to the class of short VMs and long VMs.



(a) VMs distribution with datacenter consumption and solar panels production for the RR algorithm



(b) VMs distribution with datacenter consumption and solar panels production for the RR_LPT algorithm



(c) VMs distribution with datacenter consumption and solar panels production for the RR_CLASSES algorithm

Figure 11: Round Robin algorithms

Several assumptions can be made for this distribution of VMs:

- Either all short and long VMs are programmed
- Or there is no longer enough vCPU available on the servers over a long time to run the long VMs

To understand the different scheduling results obtained in the above algorithms, the following parts will analyze the energy result regarding VMs characteristics.

6.3. Energy consumption

6.3.1. The consumption

For each algorithm, consumption was compared to the scheduling and ejected VMs. The figure 12 presents these results. For each algorithm, the consumption value varies from 11,500Wh to 13,000Wh for 48 hours. The least consumptive algorithm is RR_CLASSES (11,500Wh). This result can be explained by the grouping of servers in classes. Each server runs a type of VM: long, medium, or short. So each server is fully optimized. In our case, there is only one server for the shortest VMs and three for the longest VMs because they use more resources over time. This algorithm also has the best result on the sale and purchase. It sells more energy than other algorithms and buys less energy.

6.3.2. The purchase

For each algorithm, the purchase value does not exceed 2000Wh for 48 hours. The algorithms FF and FF_LPT have the highest energy consumption and the highest purchases because they use the maximum resources of each server (at least the first). These results are explained by the fact that all these algorithms do not take into account the renewable energy during the execution of a VM. They only check if the VM has enough renewable energy at the instant t to be programmed at the instant t. If there is no more energy at time $t+1$, the algorithm continues to execute the VM and will obtain the battery energy (Backup = 1) or if there is no longer energy in the battery, the energy will be purchased from the electricity supplier. Servers must be permanently powered over time in contrast to a Round Robin algorithm. A Round Robin algorithm looks for server resources per server.

6.3.3. The sell

For each algorithm, the value of the sale varies from 1500Wh to 3000Wh for 48 hours. The algorithm RR_CLASSES always has the best result ($\approx$ 3000 Wh). It sells more energy than the other algorithms because the classes of some servers are full and others underused. To understand these results, it is necessary to follow the rates of ejected and scheduled VMs.

6.3.4. Scheduled and ejected VMs

The figure 13 shows the consumption in relation to the scheduled and ejected VMs. The algorithms FF, RR and RR_CLASSES have the best results for scheduled VMs, the percentage of VM ejected is very low (between 0.5 and 2 %) but the algorithm RR_CLASSES consumes less than the others. This is due to the size of the ejected VMs, because if they are longer, they will produce a higher power consumption.

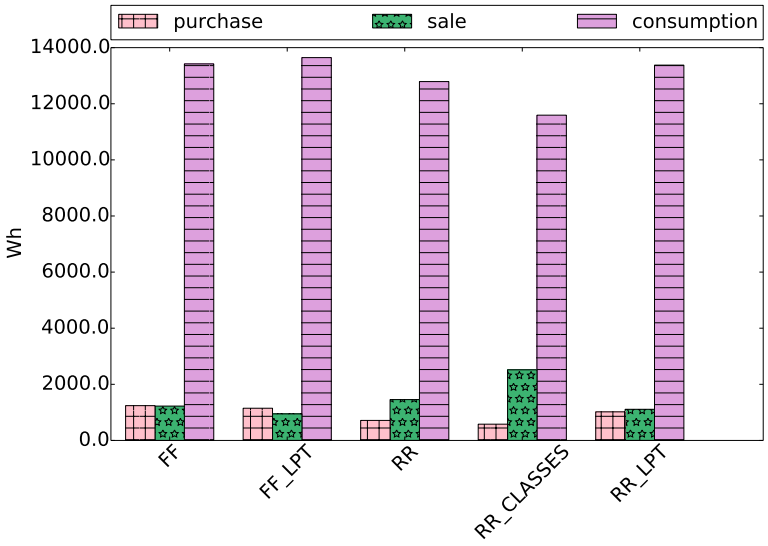


Figure 12: The datacenter consumption compared to the energy sale and purchase

6.4. Number of Virtual Machines scheduled and not scheduled

We will study the characteristics of the VMs for each algorithm, relative to their completion time indicator (Equation 8) and their execution time.

6.4.1. Scheduling results and characteristics of ejected - scheduled VMs

The figure 14 shows the percentage of VMs ejected and scheduled. The algorithm FF has the least number of VMs ejected with the algorithmsRR_CLASSES and RR. However, the figure 15 shows the size of the VMs ejected and the RR_CLASSES algorithm only ejects long VMs. This type of VM consumes more energy than the others, which explains the best consumption results compared to other algorithms (section 6.3).

The figure 15 shows that the algorithm FF_LPT has many VMs ejected for the shortest, due to the priority scheduling of long VMs.

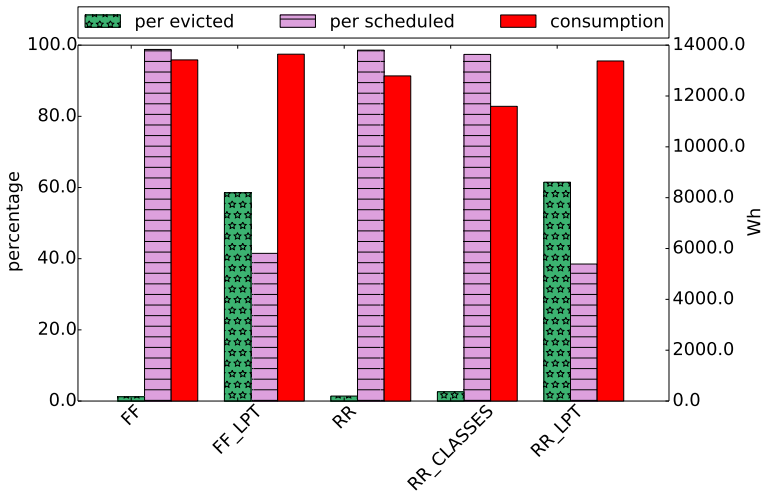


Figure 13: The datacenter consumption compared to the number of scheduled and ejected VMS

The algorithm RR_LPT has many ejected VMs and an enormous amount of time ejected (92.5 hours) since short VMs never have access to resources, long VMs take the priority over the short and medium VMs (figure 15). The vCPU resource space is not available to schedule VMs, and the available vCPUs are too short to schedule medium VMs.

6.4.2. Completion time indicator of ejected - scheduled VMs

The figure 16 shows the hourly amount of VMs ejected and the ejection percentage of each algorithm. This illustration highlights the highest hourly amount of ejected VMs (65 hours) held by the algorithm RR_CLASSES. However, this algorithm has a relatively low percentage of VMs ejected.
The algorithm FF consumes more energy than the other algorithms (figure 12), this analysis is justified by the very high hourly quantity of VMs (figure 17) and the number of scheduling VMs. The algorithm FF has better results concerning the hourly quantity of the VMs ejected and the percentage of expulsion. Its hourly amount of ejected VMs is 25 hours, since all ejected VMs are medium or short VMs and they are only 12.
The algorithms RR_LPT and FF_LPT have scheduled more than 70 % of the VMs and do not have the lowest scheduled time quantity this is explained by the scheduling prioritization of the long VMs.

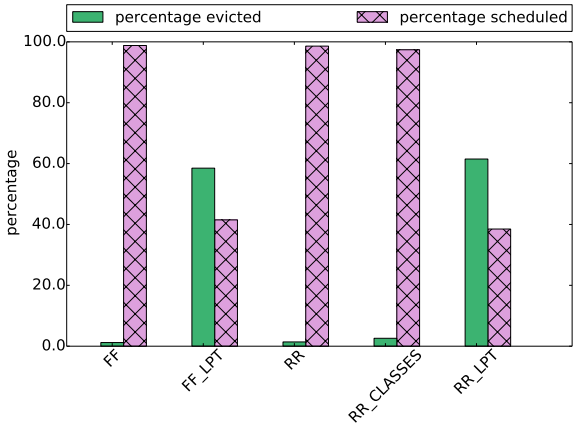


Figure 14: The scheduling result compare to the percentage of scheduled and ejected VMs for each algorithm

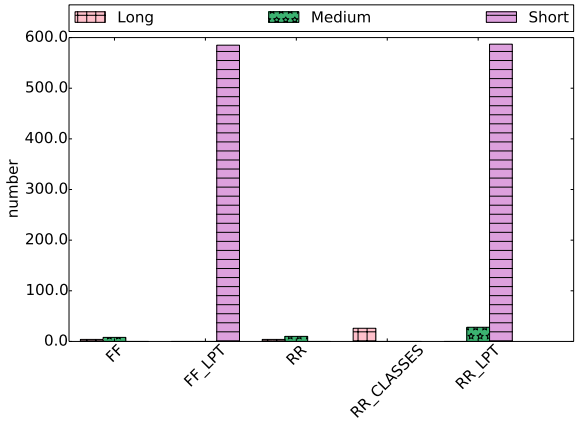


Figure 15: The scheduling result for the size of ejected VMs

6.5. Energy impact

As a reminder, the equation 7 proposes to calculate the GEC in our case. This equation allows us to quantify the share of renewable energy consumed by datacenter. The greater the sale will be and the lower the purchase,

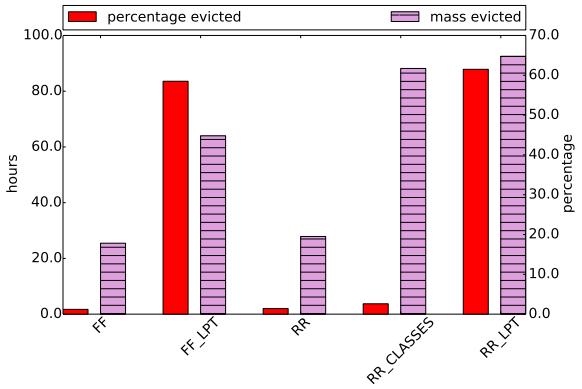


Figure 16: Scheduling result for the completion time indicator for the ejected VMs

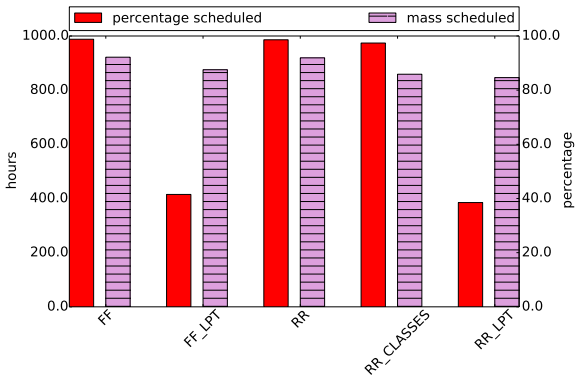


Figure 17: Scheduling result for the completion time indicator for the scheduled VMs

the higher the CEG will be. The figure 18 shows the GEC results for each algorithm. To confirm these results, the table 6 presents the results of the different metrics used for each algorithm in order to evaluate the scheduling strategy.

The algorithm FF_LPT consumes the most, sells the least and buys the most energy, which is why its GEC is the weakest. However, its ejected hourly quantity is the best, given that it will order the longest VMs first.

The best GEC is held by algorithms with a very high ejected VMs(RR_CLASSES et RR_LPT) but the two Round Robin algorithms do not optimize the resources of the different servers. This is why there are many VMs ejected. However, there are fewer VMs ejected for the RR_CLASSES this is due to the server classes. This algorithm can be optimized by redefining the server classes. In our case, there are not enough resources to run certain sizes of VMs. According to figure 15, the most ejected VMs belong to the average type (from 45 minutes to 2 hours).

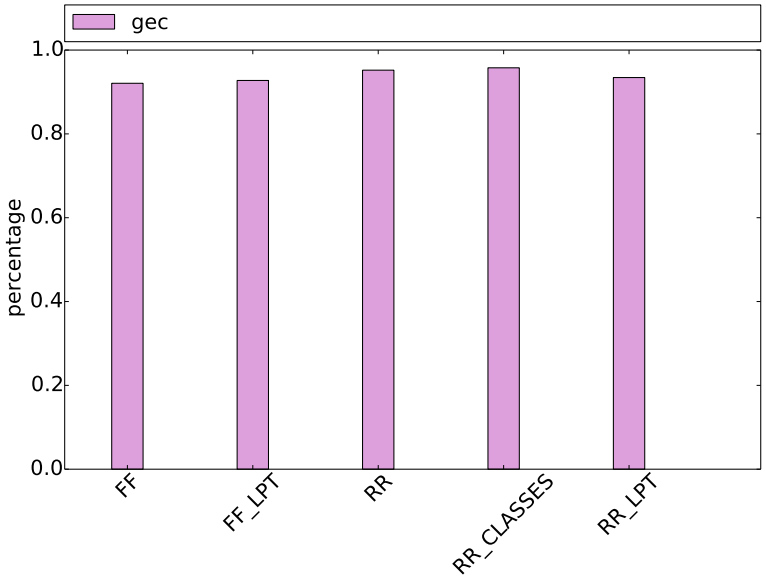


Figure 18: Gec results for each algorithm

6.6. Algorithm statement

In this section, an ecological scheduling approach was introduced. Thanks to these experiments, the various algorithms make it possible to understand the agile workload. All these analyzes are recurrent because we are experimenting on a specific study. The proposed energy scheduling strategy can improve the use of green energy. The algorithm RR_CLASSES has the best results, but we need to set the classes and we need to know

the characteristics of the workload. The algorithms FF et FF_LPT seem appropriate for scheduling an agile workload because they allow the datacenter to use the appropriate number of servers and use each server. One perspective might be to add a QoS rate as the main constraint to get a better QoS. To achieve this result, we can use the priorities and latency offered by Google.

Through all these experiments, the use of algorithms and models helps us to meet specifications expected by the datacenter administrator. The figure 19 is an illustration of the algorithms manipulation. Each algorithm certifies a goal:

- *QoS*: ensure QoS to schedule VMs (number of scheduled VMs, or promote long VMs)
- *Environment*: promote the use of green energy
- *Lucrative*: make the datacenter lucrative (sell green energy)

The table 7 gives an appreciation to each algorithm according to the expected specification:

- *+*: acceptable
- *++*: fine
- *+++*: good

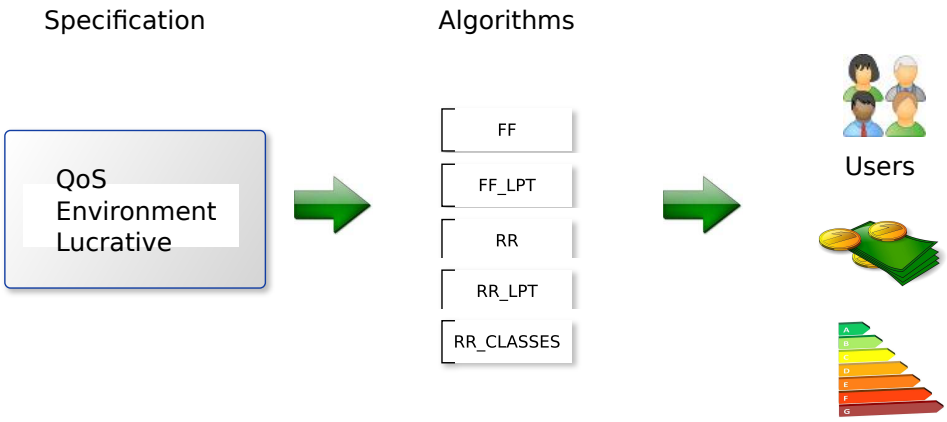


Figure 19: Illustration of decision-making in relation to a specification

The following experimentation is a study on the correlation between solar production and the capacity of the battery according to a workload.

7. Sizing experiments

7.1. Experiments set up

Through the RenewSim simulator [29], the following study highlighted the influence of battery size and solar production on a workload. A set of simulations was carried out on the same workload by varying the input data of the simulator, in this case the battery storage capacity and the solar production.

To evaluate the size of the battery relative to the production of the solar panels, it was chosen to analyze the GEC. The scheduler uses the algorithm FF because it presents the best results in terms of compromise between the various indicators (table 6). The same workload is maintained, but the battery size and solar production are modulated to evaluate the behaviour of the GEC.

The figures 20a and 20b are an example of an experience where solar production evolves from one scheduling to another with the same battery capacity of 10,000 Joules. At each start of the experiment the battery is initialised to 0 joules. When solar production decreases, the consumption of the datacenter exceeds production. In the figure 20a, the workload consumption at 600 to 900W and 1500 to 1600W is still almost higher than the solar production curve. This measure corresponds to the afternoon when the sun goes down. The reason why the workload curve is not equal to solar production, is the scheduling method. The algorithm checks at the instant t if there is enough energy to schedule a VM, but the algorithm does not check at time $t+n$, if the energy will always be available. Some VMs were scheduled just before the end of the day, when solar production began to decline.

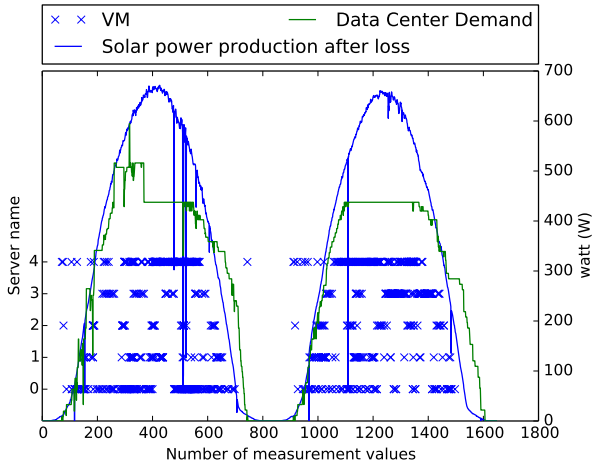
Table 6: The metrics results observed for each algorithm

Metric	**RR**	**FF**	**RR_CLASSES**	**RR_LPT**	**FF_LPT**
Number of ejected VMs	14 (1.4%)	12 (1.2%)	49 (2.6%)	615 (61.5%)	585 (58.5%)
Number of scheduled VMs	986 (98.6%)	988 (98.8%)	981 (97.4%)	385 (38.5%)	415 (41.5%)
Completion time scheduled (≈ hours)	920	922	858	846	875
Completion time ejected (≈ hours)	28	25	88	92.5	63
GEC	0.95	0.92	0.96	0.93	0.93

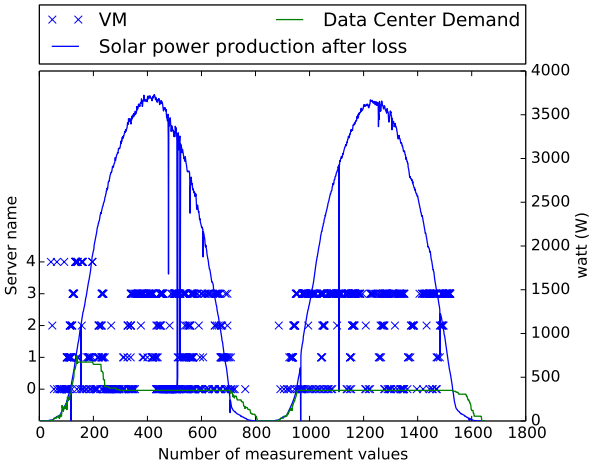
Table 7: Decision-making policies according to algorithms

Objectives		RR	FF	RR_CLASSES	RR_LPT	FF_LPT
QoS	Number of scheduled VMs	++	+++	++	+	+
	Longs VMs	+	+	++	+++	+++
Environment	Renewable energy	++	++	+++	+	+
Lucrative	Renewable energy sale	++	++	+++	++	+

The figure 20b shows the distribution of the VMs with the same workload but with a higher solar panel output. The algorithm used is still the FF. The difference with the previous scheduling is the use of servers. There is enough solar energy and vCPU resources to run the different VMs. As a result, the server *S4* is a little bit used.



(a) VMs distribution with datacenter consumption and production of solar panels for algorithm FF



(b) VMs distribution for algorithm FF with an higher solar panel production

Figure 20: Scheduling example for the same workload with a different solar production

The following two experiments are the result of a series of simulations for which the size of the battery and solar production have evolved. These experiments help to:

- Know the best correlation between the battery capacities and the solar production through the GEC
- Identify a stage where the battery capacities no longer has any influence on the GEC.

The performance objectives of each sizing is the use:

- *Sizing 1*: of a conscious environmental scheduling algorithm (with an agile workload)
- *Sizing 2*: of a less conscientious scheduling algorithm using much more battery (without an agile workload)

7.2. Sizing 1 with an agile workload

In this part, we present a set of simulation realised with RenewSim. Each simulation result is stored in a history, allowing us to analyze and compare each result.

7.2.1. Experiment set up

To realize, all the simulations we:

- used the algorithm FF presented in the section 4.4
- Initialized the battery to 0 joule
- Increased solar production from ≈ 2kWh to an other simulation
- Modulated battery size from 40,000 joules to 200,000 joules
- Took the workload from Section 6 (600 shorts, 200 mediums, 200 longs) ;
- Integrated the same datacenter from Section 6
- Applied the same energy strategy from Section 6.

7.2.2. Analyze

The figure 21 shows the evolution of the GEC when the battery capacities and solar production are changed. This makes it possible to understand the optimal configuration, where these two elements obtain the best result in terms of GEC. When the size of the battery and the production of solar panels increases (from 0 to 18kWh for the production of the solar panels and from 0 to 200 000 joules for the battery), this increases the GEC. However, the battery does not allow the smart grid to improve the GEC, because in the algorithm, the battery is a backup for the simulation and not a constraint. It does not appear in the GEC equation. This is not the main factor in its outcome.

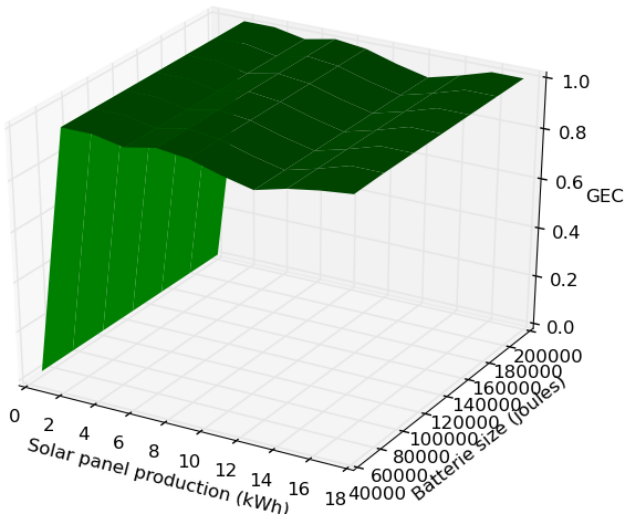


Figure 21: GEC evolution compared to the solar panel and battery size (FF algorithm)

The figure 22 is an example of how the battery can increase the GEC. When the execution of a VM has started, and when the solar energy is enough, the VM continues even if there is no more green energy. If there is enough energy in the battery, it is not necessary to buy energy from the supplier, this purchase is included in the GEC equation, so that the GEC result will be better.

The GEC is growing faster due to its equation. If there are no scheduled VMs and a little solar panel, the GEC will be equal to 1 because it does not buy electricity. The control module will only sell and store energy.

In this case, the battery has little influence on the calculation of the GEC, since it is only for a few hours, when the datacenter is in the case of the figure 22 and the control module still buys electricity because of the battery maximum power. It is not enough to provide t all VMs at the moment to get a better GEC.

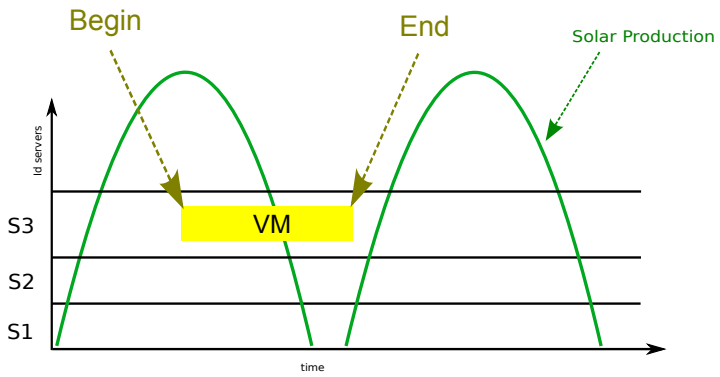


Figure 22: Example of the battery leverage for the GEC

7.3. Sizing 2 with a constant workload

The first dimension does not allow to see the evolution of the GEC because even if the majority of the VMs are ejected and the energy purchase is weak, its value can be close or equal to 1. To overcome this, another scheduling algorithm was used.

7.3.1. Experiment set up

The algorithm FF2 presented below is based on the algorithm FF whose energy constraint has been suppressed. The only constraint is the servers availability.

Algorithm 4 FF2

```
while VM < nbVmsScheduled do
  for time=0;time<schedulingWindow;time++ do
    if servers[s][time] == true then
      VM ⇒ server[s][time]
      if serverFull == true then
        servers[s + 1]
      end if
    end if
  end for
end while
```

The figure 23 is an example of scheduling for the algorithm FF2. This illustration shows the VMs scheduled over time, with a peak production of solar panels of nearly 700W and a battery capacity of 10,000 joules. The number of servers used for the scheduling VMs is less than the FF algorithm because the VMs have enough vCPUs for 48 hours on the different machines. The workload decreases over time because the number of scheduled VMs is lower in time. The server *S3* only has scheduled VMs on the first day.

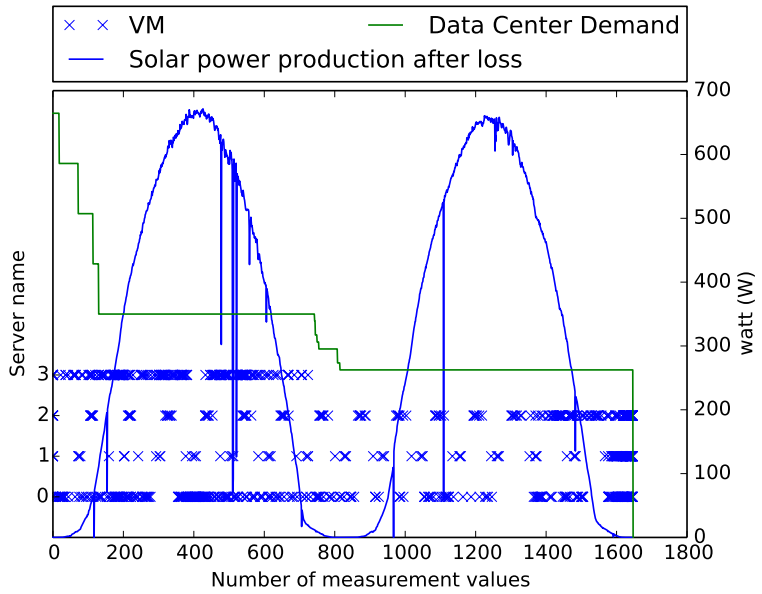


Figure 23: VMs distribution with datacenter consumption and production of solar panels for algorithm FF2

The experiment set up are identical to the previous experiment (section 7.2.1) except on the use of the algorithm.

7.3.2. Analyze

The figure 24 shows the evolution of the GEC. It is better when solar production increases. The influence of the battery size appears at the moment when the solar production begins to reach 10kWh. The bigger the battery capacities is the more the GEC increases. This makes it possible to buy less and less from the supplier. When the solar production is 18kWh, the battery has a pronounced influence on the GEC.

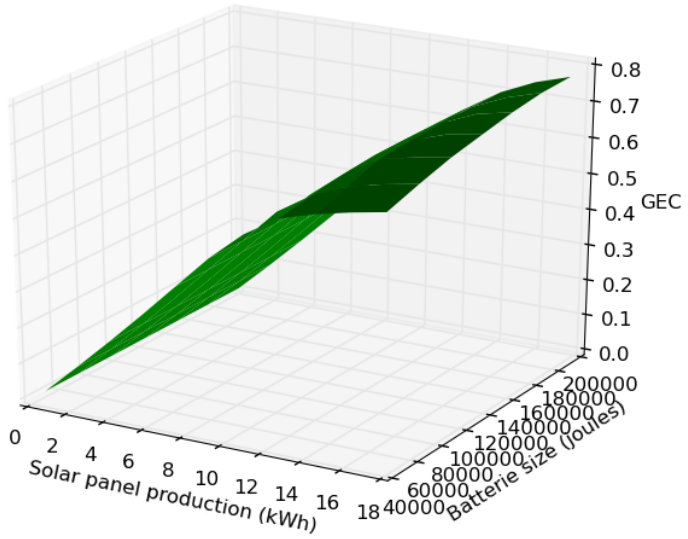


Figure 24: GEC evolution compared to the solar panel and battery size (algorithm 4)

The table 8 present some GEC values:

- *For a solar production of 0kWh*: the GEC is at 0 this is explained by the only use of non-green energy

and the battery load strategy. At the beginning of each simulation, it is empty and can only be loaded for the duration of the simulation. In this case, there is no solar production, so no loading.

- *For a solar production of 8kWh*: the GEC is the same for any battery size. In this case, the battery has no influence on the GEC.
- *For a solar production of 10kWh and 12kWh*: the GEC increases slightly thanks to the battery capacities.
- *For a solar production of 16kwh and 18kWh*: the GEC increases significantly with the battery capacities. In these cases, the battery influences the indicator.

Table 8: GEC values compared to the solar panel and battery size (algorithm 4)

Battery \ Solar	0kWh	8kWh	10kWh
50 000 joules	0	0.45	0.53
100 000 joules	0	0.45	0.55
200 000 joules	0	0.45	0.56
	12kWh	16kWh	18kWh
50 000 joules	0.59	0.64	0.65
100 000 joules	0.62	0.69	0.71
200 000 joules	0.63	0.73	0.76

7.4. Sizing statement

It was presented the influence of battery size and solar production and their correlation on the GEC. In our case, the indicator gets better results if the algorithm used has a goal of conscientious performance of the environment (section 7.2). However, this algorithm does not emphasise the battery role. In the second dimension (part 7.3), the battery optimizes the green energy use. This is explained by the increase of the GEC with a high battery capacities and solar production association.

The dimensioning results are only used on a given workload. A simulation perspective could be on workload, for example, by modulating the number of long, medium, short VMs to get the right number of servers.

8. Conclusion

The purpose of this article was to provide a green scheduling approach for a datacenter feeding by several energies:

- We modeled a system states which summarizes the different states (over-production, under-production and flow integrity)
- We used a virtual machine model based on the Google Traces workload
- We adapted a set of algorithms and a set of metrics

We modeled a system states which summarizes the different states (over-production, under-production and flow integrity) and the related denominations (source, over-flow, back-up and destination) of the *manageable* and *not-manageable* elements have been proposed. Then, these both states and denominations have been used in a generic matrix which allow to defined all possible optimization strategies.

We used a virtual machine model based on the Google Traces workload. The virtual machine model used in this article has been done based on the Google Traces workload [4]. The analyzes done on a previous work on this dataset have allowed to extract few relevant properties in order to set-up a quite accurate virtual machine model based on real data. Moreover, three metrics have been proposed to evaluate different kind of scheduling behaviour. The GEC metric has been adapted to the global electrical context of the proposed architecture while taking into account both energy purchase and sell from the supplier. The two others is more focused on the ability of having a suitable number of computing resources to schedule virtual machines.

We have adapted a set of algorithms to meet the following objectives:

- Create a scheduling based on green energy prediction

- Be as little dependent as possible on the electricity supplier

- Be able to evaluate scheduling using metrics

We chose greedy algorithms because the use of green energy is not sustainable in time and it allows us to get a quick decision making. These different algorithms have the same types of inputs to make them comparable and measurable between them. We have defined our QoS indicators that judge compliance of the computing center to allocate a sufficient number of resources to the VMs. We have adapted an indicator to evaluate the percentage of green energy used in the data centre. The constraints of the algorithms are the same, and correspond to the space available in terms of VCPU for each server and the available produced energy. These algorithms allow us to obtain an agile and flexible workload. However, these algorithms need the weather predictions in order to perform scheduling. Currently, scheduling does not adapt to actual production. It is based solely on the predictions of solar production achieved. Moreover, VMs continue to run even if there is no more solar production at this time t. One possible improvement would be to suspend the work of a VM and wait for the energy to return. A second solution is the real-time migration [42] to another server in order to reduce the total consumption of the computing center and in addition allow a gain of energy. Moreover, in these algorithms, there is no prediction of battery charge over time to optimize their use. This could allow it to be used more, not minimally, in order to buy less electricity from the supplier.

Future work concerns both application and physical areas. It includes the opportunities to temporally suspend a virtual machine execution for a while and also to allow the live migration between two computing resources. Improvement on the storage element modelling can be done to better predict its load and unload periods. The analyzes phase will be extending by including more parameters given in the Google dataset. In a scheduling optimization point of view, the use a bio-inspired heuristic could be a great solution to conduct a multi-objective simulation while maintaining a scalability. Finally, in-vivo experimentation thanks to real experimental platform will be the next main step in order to face unpredictable natural events.

9. Acknowledgment

This work was supported by the neOCampus operation funded by University Paul Sabatier, Toulouse, France.

References

[1] H. Lei, T. Zhang, Y. Liu, Y. Zha, X. Zhu, Sgeess: Smart green energy-efficient scheduling strategy with dynamic electricity price for data center, Journal of Systems and Software 108 (2015) 23–38.

[2] D. S. Palasamudram, R. K. Sitaraman, B. Urgaonkar, R. Urgaonkar, Using batteries to reduce the power costs of internet-scale distributed networks, in: Proceedings of the Third ACM Symposium on Cloud Computing, ACM, 2012, p. 11.

[3] V. Kontorinis, J. Sampson, L. E. Zhang, B. Aksanli, H. Homayoun, T. S. Rosing, D. M. Tullsen, Battery Provisioning and Associated Costs for Data Center Power Capping, Department of Computer Science and Engineering, University of California, San Diego, 2012.

[4] C. Reiss, J. Wilkes, J. L. Hellerstein, Google cluster-usage traces: format + schema, Technical report, Google Inc., Mountain View, CA, USA, revised 2012.03.20. Posted at `http://code.google.com/p/googleclusterdata/wiki/TraceVersion2` (Nov. 2011).

[5] H. Casanova, A. Legrand, M. Quinson, Simgrid: A generic framework for large-scale distributed experiments, in: Computer Modeling and Simulation, 2008. UKSIM 2008. Tenth International Conference on, 2008, pp. 126–131. doi:10.1109/UKSIM.2008.28.

[6] W. Liu, H. Li, W. Du, F. Shi, Energy-aware task clustering scheduling algorithm for heterogeneous clusters, in: Proceedings of the 2011 IEEE/ACM International Conference on Green Computing and Communications, IEEE Computer Society, 2011, pp. 34–37.

[7] A. Lebre, A. Legrand, F. Suter, P. Veyre, Adding storage simulation capacities to the simgrid toolkit: Concepts, models, and api, in: Cluster, Cloud and Grid Computing (CCGrid), 2015 15th IEEE/ACM International Symposium on, IEEE, 2015, pp. 251–260.

[8] J.-C. Charr, R. Couturier, A. Fanfakh, A. Giersch, Energy consumption reduction with dvfs for message passing iterative applications on heterogeneous architectures, in: Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International, IEEE, 2015, pp. 922–931.

[9] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, R. Buyya, Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Software: Practice and Experience 41 (1) (2011) 23–50.

[10] R. Buyya, M. Murshed, Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing, Concurrency and computation: practice and experience 14 (13-15) (2002) 1175–1220.

[11] T. Guerout, T. Monteil, G. D. Costa, R. N. Calheiros, R. Buyya, M. Alexandru, Energy-aware simulation with dvfs, Simulation Modelling Practice and Theory 39 (2013) 76 – 91, s.I.Energy efficiency in grids and clouds. doi:http://dx.doi.org/10.1016/j.simpat.2013.04.007.
URL http://www.sciencedirect.com/science/article/pii/S1569190X13000786

[12] N. Deng, C. Stewart, D. Gmach, M. Arlitt, J. Kelley, Adaptive green hosting, in: Proceedings of the 9th international conference on Autonomic computing, ACM, 2012, pp. 135–144.

[13] Y. Zhang, Y. Wang, X. 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.

[14] S. Bird, A. Achuthan, O. A. Maatallah, W. Hu, K. Janoyan, A. Kwasinski, J. Matthews, D. Mayhew, J. Owen, P. Marzocca, Distributed (green) data centers: A new concept for energy, computing, and telecommunications, Energy for Sustainable Development 19 (2014) 83–91.

[15] Z. Liu, M. Lin, A. Wierman, S. H. Low, L. L. Andrew, Greening geographical load balancing, in: Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems, ACM, 2011, pp. 233–244.

[16] Í. Goiri, K. Le, M. E. Haque, R. Beauchea, T. D. Nguyen, J. Guitart, J. Torres, R. 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.

[17] Í. Goiri, K. Le, T. D. Nguyen, J. Guitart, J. Torres, R. 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.

[18] B. Aksanli, J. Venkatesh, L. Zhang, T. Rosing, Utilizing green energy prediction to schedule mixed batch and service jobs in data centers, ACM SIGOPS Operating Systems Review 45 (3) (2012) 53–57.

[19] W. Deng, F. Liu, H. Jin, C. Wu, X. Liu, Multigreen: Cost-minimizing multi-source datacenter power supply with online control, in: Proceedings of the fourth international conference on Future energy systems, ACM, 2013, pp. 149–160.

[20] H. Field, G. Anderson, K. Eder, Eacof: A framework for providing energy transparency to enable energy-aware software development, in: Proceedings of the 29th Annual ACM Symposium on Applied Computing, ACM, 2014, pp. 1194–1199.

[21] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, M. A. Kozuch, Towards understanding heterogeneous clouds at scale: Google trace analysis, Intel Science and Technology Center for Cloud Computing, Tech. Rep (2012) 84.

[22] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, M. A. Kozuch, Heterogeneity and dynamicity of clouds at scale: Google trace analysis, in: Proceedings of the Third ACM Symposium on Cloud Computing, ACM, 2012, p. 7.

[23] M. Alam, K. A. Shakil, S. Sethi, Analysis and clustering of workload in google cluster trace based on resource usage, arXiv preprint arXiv:1501.01426, 2015.

[24] Y. Chen, A. S. Ganapathi, R. Griffith, R. H. Katz, Analysis and lessons from a publicly available google cluster trace, EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2010-95 94.

[25] F. Gbaguidi, S. Boumerdassi, É. Renault, E. Ezin, Characterizing servers workload in cloud datacenters, in: Future Internet of Things and Cloud (FiCloud), 2015 3rd International Conference on, IEEE, 2015, pp. 657–661.

[26] S. Di, D. Kondo, W. Cirne, Characterization and comparison of cloud versus grid workloads, in: 2012 IEEE International Conference on Cluster Computing, IEEE, 2012, pp. 230–238.

[27] S. Di, D. Kondo, F. Cappello, Characterizing cloud applications on a google data center, in: 2013 42nd International Conference on Parallel Processing, IEEE, 2013, pp. 468–473.

[28] Z. Liu, S. Cho, Characterizing machines and workloads on a google cluster, in: 2012 41st International Conference on Parallel Processing Workshops, IEEE, 2012, pp. 397–403.

[29] I. De Courchelle, T. Monteil, Y. Labit, T. Guerout, A data model for supplying a Data Center with several energy sources, in: workshop WSSC, Toulouse, France, 2016.

[30] L. Kleinrock, Analysis of a time-shared processor, Naval Research Logistics (NRL) 11 (1) (1964) 59–73.

[31] A. Khosravi, S. K. Garg, R. 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.

[32] G. Da Costa, L. Grange, I. De Courchelle, Modeling and generating large-scale google-like workload, in: Green and Sustainable Computing Conference (IGSC0 2016 Seventh International, IEEE, 2016, pp. 1–7.

[33] G. Da Costa, L. Grange, I. De Courchelle, Modeling, classifying and generating large-scale google-like workload, in: Elsevier, Sustainable Computing: Informatics and Systems, 2018.

[34] G. Grid, The green grid data center power efficiency metrics: Pue and dcie, Green Grid report, 2007.

[35] C. Garnier, M. Aggar, M. Banks, J. Dietrich, B. Shatten, M. Stutz, E. Tong-Viet, Data centre life cycle assessment guidelines, The Green Grid, white paper 45 (2012) v2.

[36] T. G. Grid, `https://www.thegreengrid.org/en/newsroom/news-releases/global-leaders-industry-and-government-reach-agreement-measuring-data-center` (2012).

[37] K. Kritikos, B. Pernici, P. Plebani, C. Cappiello, M. Comuzzi, S. Benrernou, I. Brandic, A. Kertész, M. Parkin, M. Carro, A survey on service quality description, ACM Computing Surveys (CSUR) 46 (1) (2013) 1.

[38] R. Buyya, A. Beloglazov, J. Abawajy, Energy-efficient management of data center resources for cloud computing: a vision, architectural elements, and open challenges, arXiv preprint arXiv:1006.0308, 2010.

[39] S. Islam, K. Lee, A. Fekete, A. Liu, How a consumer can measure elasticity for cloud platforms, in: Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering, ACM, 2012, pp. 85–96.

[40] F. D. Rossi, M. G. Xavier, C. A. De Rose, R. N. Calheiros, R. Buyya, E-eco: Performance-aware energy-efficient cloud data center orchestration, Journal of Network and Computer Applications 78 (2017) 83–96.

[41] F. Alvarruiz, C. de Alfonso, M. Caballer, V. Hern'ndez, An energy manager for high performance computer clusters, in: Parallel and Distributed Processing with Applications (ISPA), 2012 IEEE 10th International Symposium on, IEEE, 2012, pp. 231–238.

[42] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield, Live migration of virtual machines, in: Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation-Volume 2, USENIX Association, 2005, pp. 273–286.