préparée dans le cadre d'une cotutelle entre l'Université de Grenoble et l'Universidade de São Paulo

Spécialité : **Informatique**

Arrêté ministériel : 6 janvier 2005 - 25 mai 2016

Présentée par

Miguel Felipe SILVA VASCONCELOS

Thèse dirigée par **Fanny DUFOSSÉ**
et codirigée par **Daniel CORDEIRO**

préparée au sein du **Laboratoire d'Informatique de Grenoble** et **Escola de Artes Ciências e Humaindades**
dans les Écoles Doctorales **MSTII** et **Programa de Pós-Graduação em Sistemas de Informação** et **Programa de Pós-Graduação do IME/USP**

Apprentissage sur Heuristiques Simples pour l'Ordonnancement *Online* de Tâches Parallèles

Carbon aware approaches for sizing and operating cloud data centers

Thèse soutenue publiquement le **27 novembre 2019**,
devant le jury composé de :

Martin SCHULZ
Professeur d'Université, Leibniz Supercomputing Centre, Technische Universität München, Allemagne, Rapporteur

Alfredo GOLDMAN VEL LEJBMAN
Professeur Associé, IME-USP, Universidade de São Paulo, Brésil, Président

Sarita BRUSCHI
Professeure Assistante, ICMC-USP, Universidade de São Paulo, Brésil, Examinatrice

Éric GAUSSIER
Directeur de Laboratoire, LIG, Univ. Grenoble Alpes, France, Examinateur

Raphael Y. DE CAMARGO
Professeur d'Université, CMCC, Universidade Federal do ABC, Brésil, Co-Directeur de thèse

Denis TRYSTRAM
Professeur d'Université, LIG, Univ. Grenoble Alpes, France, Directeur de thèse

Aos meus pais, Elena e Claudio.

> *Absque sudore et labore nullum opus perfectum est.*
>
> — **Schrevelius**

Remerciements (Acknowledgments)

Agradecer aos meus pais e irmao

meus amigos do lab e da each

meus orientadores

colaboradores do proejto datazero

tb a each e a univ de grenoble, fapesp, labex, pessoal do datazero e td mais

Abstract / Résumé

Abstract

Falar do consumo de energia

impacto ambiental dos dcs

emissao de carbono

tecnicas carbon aware

carbon responsive

sizing e contention

follow the renewables

Résumé

Falar do consumo de energia

impacto ambiental dos dcs

emissao de carbono

tecnicas carbon aware

carbon responsive

sizing e contention

follow the renewables

List of Figures

List of Tables

Contents

1 Introduction

- consumo de energia aumentando
- workload e evolucao de hardware (masanet)
- carbon aware
- not-so-urgent-computing
- inclusao de energias renovaveis nso dcs
- follow the renwables
- sizing
- intermitencia
- impacto ambiental de fabricar

1.1 Contributions

In this thesis we explore scheduling, sizing. More specifically, this thesis proposes the following broad sets of contributions:

1. algorithms for network aware and impact of live migrations without this consideration;
2. sizing;
3. long term sizing?

1.2 Content

The remainder of this thesis is organized as follows: In Chapter 2 we provide background knowledge, notably about scheduling and machine learning, to introduce the reader to the contributions of the thesis, as well as we present the closely related works. Chapter **??** we present the aforementioned procedure of learning scheduling heuristics, with its respective experimental results and discussions. In Chapter 4 we present the aforementioned experimental campaign to provide insight on possible expectations and performance gains if one replaces EASY Backfilling, with its respective experimental results and discussions. At last but not least, in Chapter 5 we present a general discussion about the achieved contributions of this thesis, with the closing remarks and future research directions.

Background and Related Work

2

In this chapter we provide background knowledge to the reader to understand the work performed at the remaining chapters of this thesis, notably Chapters and. This chapter contains background knowledge about scheduling problems and algorithms, as well as a brief introduction to machine learning, with emphasis on regression problems. Furthermore, to increase the presentation flow and quality of this chapter, many recent and related works are also presented in a intertwined manner along the text of this chapter.

- scheduling
- follow the renewables
- consolidation
- sizing / capacity planning

Related work from the previous papers:

Cloud infrastructures became a critical component of society in the last decade, from private life to big company development. The energy efficiency of these platforms has been widely studied and improved by academics and Cloud providers [21]. This progress, however, did not lead to a reduction of global Cloud energy consumption. In [20], authors estimate the growth of Data Centers (DCs) needs between 2010 and 2018 to a 10-fold increase in IP traffic, a 25-fold increase in storage capacity, and a 6-fold increase of DCs workload. The impact of this explosion of usages has thus been limited by efficiency improvement of platforms to an energy increase of only 6%. Projections over the following years are, however, quite pessimistic. In [19],

authors consider different scenarios for the period 2016–2030, with predictions ranging between a wavering balance and a significant increase in electricity needs.

These predictions consider big trends in IT, but they do not embrace unpredictable events, such as the COVID pandemic, and particularly the lockdown periods, that overturned the global Information Technologies (IT) usages [10].

Another approach to reducing the environmental impact of cloud computing energy needs consists of studying DCs' energy sources. Most Cloud providers have made commitments to renewable energy usage in recent years. According to a Greenpeace report, [13], many DCs were already fully supplied by renewable energy in 2017. However, they are not the majority of cases. A typical example is the IT infrastructures of Virginia, which are named the "Ground zero for the Dirty Internet", with 2% of renewable energy power plants against 37% of coal. They are known to host 70% of US internet traffic. Green computing is still chimerical.

A critical question on renewable energy production facilities is their intermittency. Hydroelectric dams and, to a certain extent, offshore windmills can provide constant energy. However, they are not appropriate for on-site electricity production for a DC. Onshore windmills and solar farms are more likely to be deployed with minimal constraints. The on-site electricity production is thus determined by local weather. In contrast to wind speed, which can be difficult to predict, solar irradiance follows daily and yearly patterns. Photovoltaic (PV) panels are thus more appropriate for predictable on-site renewable energy production facilities.

Most sizing research focuses on a single DC. There are two approaches, either to consider that the DC can use the electrical grid as a fallback or to consider how to size a DC only with on-site renewable sources.

Most sizing approaches consider the capability to use the electrical grid. [23] use a Particle Swarm Optimization approach for sizing a smart microgrid to supply fog DCs located in a rural area in India. The objective of the optimization is to

reduce the capital cost of buying solar panels, wind turbines, diesel generators, and batteries. Power from the regular electrical grid can be used when there is no green power production. The authors also propose a scheduling algorithm to maximize green energy usage. Niaz et al. [22] evaluates using curtailed renewable energy to power DCs and provide hydrogen to hydrogen refueling stations. The authors model their problem as a MILP (Mixed Integer Linear Programming) with the objective of minimizing the costs. System components included natural-gas–powered combined cooling, heating, and power systems, electrolyzers, hydrogen fuel cells, heat pumps, hydrogen tanks, and battery energy storage systems. The results were that using only power from the electrical grid was the worse in both economic and environmental terms. Using a mix of curtailed renewable energy and electricity from the grid was the most economical. Using only renewable energy was the best for the environment; however, it had the highest costs.

In some cases, the approach considers how also to size on-site energy production, removing the need to access the electrical grid. Richter et al. [27] proposes a planning methodology for net-zero energy systems, and performed a qualitative study to evaluate a net-zero energy DC located in Germany. The conclusion is that by selecting appropriate technologies for energy generation, increasing energy efficiency, and optimal sizing Energy Storage Systems, the DC showed large potential to operate as a net-zero energy system. A DC as a net-zero energy system can increase the marketing image and add economic value to the related company. Haddad et al. [16] proposes to size a DC using only on-site renewable energy and energy storage systems (batteries and hydrogen). This work focuses on a single DC and discusses the impact of its location, its workload, and its context on the resulting sizing (number of servers, renewable sources, and storage). Benaissa et al. [5] proposes to reduce the usual oversizing of renewable-powered DCs. Classical sizing approaches based on traces are defined by a few days with unusually high workloads and/or low renewable availability. In this work, the authors propose to reduce such sizing and evaluate the impact on the Quality of Service and on the

sizing itself. Contrary to the previous studies, they use a binary search approach to find the best relevant sizing instead of MILP formulation.

Some research focuses on the sizing of particular elements, such as the electrical infrastructure. Sheme et al. [29] studies the impact of the battery size to reach a specific green coverage of 50% (half of the energy consumption of the DC needs to be green). They develop a simulation tool that uses as input the area of PVs and capacity of the batteries. Experiments comparing countries (Finland, Crete, and Nigeria) show that the number of solar panels needed in Crete and Finland is slightly higher than in Nigeria, 17% and 45%, respectively. However, although Finland provides only 15% less annual solar energy than Nigeria, it requires a battery size of 39 times bigger to achieve wasted energy at level 0. While in Crete, a battery capacity of only 27% greater than in Nigeria is needed.

Overall, most studies focus on sizing individual DCs. This is similar in the context of scheduling renewable-powered DCs: Song et al. [30] reviews recent publications on the field of DCs powered by Renewable Energy mix. It shows that among more than 100 publications, only a quarter focuses on geographically distributed data centers partially powered by renewable energy mix. It also shows that most research in this field focuses on workload scheduling, while few articles focus on the adaptation or sizing of the infrastructure.

The Carbon Explorer framework is an example of a study that explores sizing multiple DCs [1]. The framework explores three solutions to achieve 100% renewable operation of DCs distributed over the United States of America: i) only use renewable energy; ii) use renewable energy and energy storage; and iii) use renewable energy and schedule the workload. These DCs already have access to local solar power, wind power, or both. The carbon emissions from manufacturing PVs, wind turbines, batteries, and servers are considered. An exhaustive search is used to find the solutions. The work concludes that 100% renewable operation may not be the optimal solution when considering the geographic location of the DC, and the

carbon emissions from the manufacturing phase. Furthermore, the authors say that choosing the optimal solution is still an open research question for future work.

Our approach focuses on optimally sizing of geo-distributed DCs across the globe, which has not yet been studied to the best of our knowledge. Furthermore, in contrast to the Carbon Explorer framework, our solution allows using the regular electrical grid when opportune, given it may be supplied by a low-carbon intensive source. Finally, contrary to most studies using a MILP, our model uses a linear program formulation.

2.1 Summary

In this chapter we presented a brief overview of the scheduling theory in the context of High Performance Computing, with emphasis on parallel job scheduling problems, list scheduling and backfilling algorithms. We also made a short introduction to machine learning, with emphasis on regression problems. Finding optimal solutions for parallel job scheduling problems is notoriously hard and, although there are many approximation algorithms with proven performance guarantees, many practitioners end up adopting simple heuristics, such as EASY Backfilling, to schedule parallel jobs. The content present in this chapter must be sufficient for the reader to follow the remaining chapters of this thesis. In the remaining chapters we present the main contributions of the thesis, starting by presenting the ways we explored simulation and machine learning to learn simple parallel job scheduling heuristics.

3 Impact of the follow-the-renewables approaches in energy consumption[1]

3.1 Introduction

The “follow-the-renewables” approach (the read may consult Chapter 2 for more details) is an interesting strategy for mitigating the intermittency of renewable energy availability without the need of using energy storage devices, for example, when is night in the location where a DC A resides, the workload could be migrated to another DC B where the sun is still shining to use solar power. However, this strategy also has some limitations. First, the process of migrating a job between different DCs consumes energy itself: it uses network devices (as switches and routers) and a computational task for the live-migration process. The scheduling algorithm must consider this energy consumption before deciding if the migration is advantageous. Second, the network communications links that connects the servers inside the DC and the different DCs can suffer from congestion, which may increase migration duration. This results in unnecessary computation and energy consumption on both the origin and the target server: the server that is sending the job needs to wait for the migration process to finish to free its resources and receive additional jobs, or it could be turned off as well to save energy; and the server that is receiving the job will only start executing the new job after the end of the migration

process. An efficient scheduling algorithm must consider those factors to decrease the carbon footprint of the DCs operations.

The work by [7, 6] studied the scheduling of the cloud workload, in the form of VMS, on geographically distributed DCs to minimize the non-renewable energy consumption. It proposed different stochastic models to estimate renewable energy production and greedy heuristic algorithms to allocate tasks to servers. VMs are allowed to be migrated during the execution to a computer within the same DC (intra-DC server consolidation) or to a computer in DC located on another geographic location ("follow-the-renewables"). The scheduling algorithm considers the network cost of the migration.

In this chapter, the reader will find an extension of this work with an analysis of the indirect impact of using the follow-the-renewables approach in the energy consumption. This impact can be divided in 2: direct and indirect. For the direct impact, it is caused by the energy consumption of the network devices themselves, and the power consumption is considered static since it does not vary significantly based on the device usage [17]. For the indirect impact, it is generated by the live-migration of the jobs, that uses the network to transfer all the data related to the task to the destination machine, as well needs a computational tasks to perform this live-migration. More specifically, this chapter presents the following contributions:

- an analysis of the impact of not considering the network (bandwidth, latency and topology) in both the energy consumption and network congestion
- a accurate estimation algorithm for the time it takes to migrate a virtual machine that takes into account the network topology, links bandwidth and latency
- a scheduling algorithm for the live-migration of the jobs that uses the estimation algorithm and has the same or lower brown energy consumption than the baselines, with no network congestion

This chapter is organized as follows. In Section 3.2 the model of [6], the fundation of this work is summarized. Section 3.3 details the new scheduling method for the migrations, while Section 3.4 is devoted to simulations parameters. Results are detailed in Section 3.5. Finally, Section 3.6 summarizes the chapter.

3.2 The NEMESIS modeling

The resource management framework NEMESIS [6] manages a multi-cloud composed of DCs geographically distributed across a country. The DCs power demand is supplied from the regular electrical grid and locally installed PVs. Given the intermittency of renewable energy, the NEMESIS algorithm uses the stochastic modeling of SAGITTA [7] for obtaining the expected value of the renewable energy available at a given time to be used as input for the scheduling algorithms.

NEMESIS has a central controller and the workload execution is scheduled at time slots of 5 minutes. The workload consists of heterogeneous VMs in terms of the number of CPU cores, RAM size, and requested execution time. The scheduling decision uses greedy heuristics inspired by the Best-Fit algorithm. While it may not result in the optimal solution, greedy heuristics can provide an acceptable result in a reasonable amount of time, and these characteristics are relevant for the scenario of cloud computing with millions of jobs submitted per hour and a significant portion of these tasks have strict quality of service requirements and cannot be delayed. The scheduling algorithm has four main steps detailed as follows.

In the first step of the algorithm (pre-allocation of the incoming VMs), the controller will search for servers to allocate the VMS that were received during the time slot. There are two restrictions for this scheduling algorithm: i) the server has available computational resources to host the VM; and ii) executing the VM in this server would result in the minimum increase in the expected brown energy consumption.

The algorithm first sorts the VMs by its volume (a product of the number of CPU cores and the amount of RAM) in decreasing order and then performs the search for each of the VMs. If a server is found that respects the two restrictions, the algorithm makes a reservation (pre-allocation) for the VM being analysed and goes to the next VM. On the other hand, if no server is found, the VM will be delayed to be processed in the next time slot.

The second step of the algorithm (revision of the pre-allocations) performs a revision of the reservations made by the previous step, given that the greedy heuristic used may not provide the optimal solution. The strategy is to move the reservation from the DCs expected to consume more brown energy to the DCs expected to have the most availability of green energy. There are two constraints for this algorithm: i) it exists a server in the DC being evaluated that can host the VM; and ii) the expected brown energy is reduced.

The availability of green energy may change during the execution of the VMs, and the third step of the algorithm (migration of the running VMs) aims to reduce the brown energy consumption by migrating the running VMs. It uses the same strategy as the second step of the algorithm (moves the workload from DCs using more brown power to DCs that have more green power available) and with the following restrictions: i) the migration needs to finish before the beginning of the next time slot; ii) the remaining execution time of the VM needs to be greater than the duration of the migration process; iii) one DC can only migrate to another 2 DCs during a time slot; and iv) the migrations from one DC are planned to execute one after another, that is, they cannot happen simultaneously in parallel. Restrictions (iii) and (iv) are simple heuristics to avoid network congestion with the load generated by the VM live migrations.

Finally, the last step of the algorithm (server consolidation) tries to minimize the number of servers that are turned on, since the power consumption of the servers represents about 50% of the total energy consumption of a DC [11]. For each DC,

the algorithm evaluates whether it can redistribute the running VMs (by performing live migrations among the servers of the DC — intra-DC migrations) in a way that reduces the number of servers being used.

3.2.1 Cloud Modeling

NEMESIS cloud modeling is based on a real cloud infrastructure: the Grid'5000 testbed[2]. It is considered a subset of the original platform with 1035 servers distributed among 9 data centers in France and Luxembourg: 116 servers in Grenoble; 74 servers in Lille; 38 servers in Luxembourg; 103 servers in Lyon; 240 servers in Nancy; 44 servers in Reims; 129 servers in Rennes; 151 servers in Sophia Antipolis; and 140 servers in Toulouse. The servers are considered homogeneous in terms of memory, CPU, and energy consumption, and are based in the Taurus node of the Grid'5000, equipped with two Intel Xeon E5-2630 CPUs (6 cores per processor), and 32 GB RAM.

Regarding the network, it is modeled both the connection of the servers inside the DCs (network links with 1 Gbps of bandwidth) as well as the connection between the different DCs (network links with 10 Gbps bandwidth). Figure 3.1 presents the network topology of the cloud platform (the placement of the DCs was not based on their geographic location, but to better visualize the network links). One can observe that some network links are shared by multiple DCs, thus the migration planning needs to consider this information to avoid generating network congestion and the resulting waste of resources.

Regarding the energy consumption of the servers, it is considered a linear model based on CPU usage and that the VMs always uses 100% of the requested number of CPU cores. The server presents a fixed consumption for its IDLE state (97 W), and the power consumption based on core usage is as follows: 128 W for 1 core;

[2]https://www.grid5000.fr

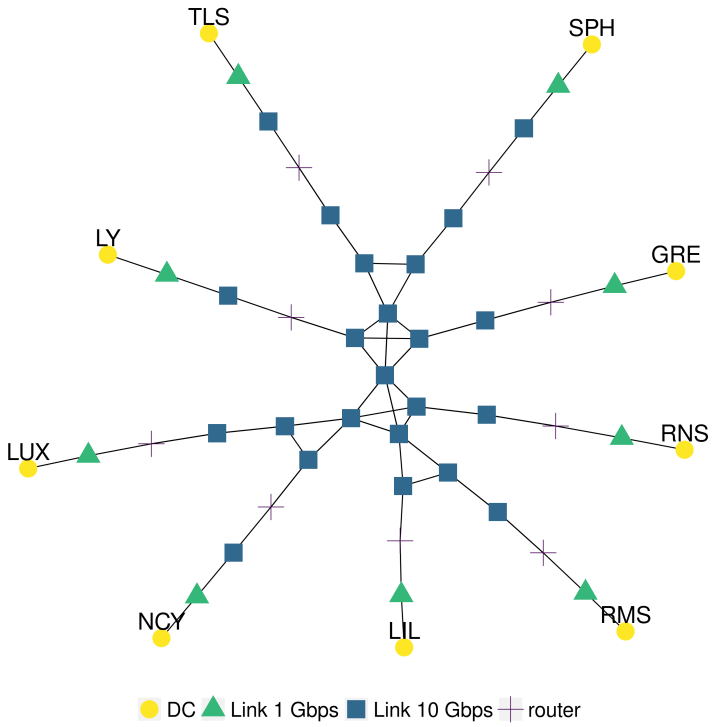

Figure 3.1 Topology of the Cloud platform, where "GRE" is Grenoble, "LIL" is Lille, "LUX" is Luxembourg,"LY" is Lyon, "NCY" is Nancy, "RMS" is Reims, "RNS" is Rennes, "SPH" is Sophia, and "TLS" is Toulouse.

146.4 W for 2 cores; 164.8 W for 3 cores; 183.2 W for 4 cores; 201.6 W for 5 cores; and 220 W with 6 cores. Furthermore, the energy consumption of turning on a server (127 W during 150 s), turning off a server (200 W for 10 s), and when the server is off (8 W) are modeled as well. Finally, only the power consumption of the servers is considered to model the power consumption of the DCs (Power Usage Effectiveness, or P.U.E., equals to 1), as we are focusing on the energy consumption of the computing part — the major consumer. Choosing a P.U.E. different than 1 would not affect the scheduling decisions made by the migration planning algorithm, since only the total power consumption would be increased by a constant factor, and the ordering of the DCs in terms of green energy availability would be the same. Furthermore, we consider a homogeneous P.U.E for all the DCs, given that the DCs are inside the same country.

3.2.2 VM live migration model

The VM live migration model of NEMESIS has 3 phases. In the first phase, all the memory pages of the VM are transfered to the destination server. In the second, after completing the copy of all the memory pages, a message is sent to notify the end of the stop-and-copy step. Finally, in the last step the commitment message is sent to the new host to inform the end of the migration process, and the VM will be destroyed in the server of origin and resumed in the destination host.

The duration of the migration is an essential information for the scheduling decision, as it can help avoiding oversaturating the network links and generating network congestion. Algorithm 1 is an extension of the NEMESIS algorithm, and it executes this estimation, where: *linkLatency* is the latency of the link; *routeSize* is the number of links that interconnects the host where the VM is originally running to the target host of the migration; *windowSize* = 4,194,304 Bytes, is the TCP maximum window size; *bandWidthRatio* = 0.97, represents the additional load caused by the headers of TCP/IP; *bandwidth* = the minimum bandwidth among the links that interconnect the host of origin with the target host; $\alpha = 13.01$, simulates the TCP slow start factor, that is, not all the bandwidth is instantly available for the communication; and γ is used to represent the Bottleneck effect of the TCP protocol. The parameters used in Algorithm 1 were based on [31]. The difference between Algorithm 1 and NEMESIS is the *routeSize* variable: NEMESIS used fixed values (5 for live migrations intra-DC, and 11 for live migrations inter-DC), and now is assumed that the cloud operator has information about the network topology of his DCs, thus the real number of links that interconnect the two hosts is used.

Algorithm 1 Estimation of the migration duration.

$theLatency \leftarrow linkLatency \cdot routeSize$
$transferLat \leftarrow theLatency + \frac{\gamma}{bandwidth}$
$throughputL \leftarrow \frac{windowSize}{2 \cdot transferLatency}$
$throughputB \leftarrow bandwidth$
$throughput \leftarrow min(throughputL, throughputB)$
$throughput \leftarrow throughput \cdot bandWidthRatio$
$timeToMigrate \leftarrow 3 \cdot \alpha \cdot transferLat + \frac{vmRamSize}{throughput}$

3.3 PLANNING THE MIGRATIONS

Algorithms 2 and 3 plan the VM migrations considering the network topology and the bandwidth usage. They are inspired in the migration planning of NEMESIS, with the following modifications: i) the available bandwidth of the links is considered (it changes based on the number of migrations that uses the same network link); ii) migrations can be performed in parallel between different DCs respecting the links bandwidth constraint; iii) the intra-DC migrations (for the server consolidation) are sequentially distributed in time (do not execute simultaneously) ; iv) the real number of links that interconnects the origin and the target server is used as input for the estimation algorithm of the migration duration.

The algorithms work as follows. Initially, the DCs are sorted by increasing order of expected remaining green energy (ERGE). The DC with the most available energy (initially at the beginning of the list) is denoted by M, and N is the DC with the least green energy (initially at the end of the list). The man idea of the algorithms is to migrate VMs from the DCs that are expected to use brown energy to the DCs that are expected to have green energy. The algorithm receives as input the information of the running VMs (grouped by the servers) of the DC M. The planning starts at the beginning of the time slot. For each VM that can be migrated, the algorithm tries to find a server in the destination DC N with the following restrictions: i) it has available computational resources to host the VM (CPU and memory); ii) the links that interconnect the server of origin (where the VM is running) and the

Algorithm 2 General planning of the migrations.

DCs ▷ Sorted by increasing ERGE
$plannedTime \leftarrow 0$
$timeSlotDuration \leftarrow 300$
$linksHistory \leftarrow \emptyset$
while $plannedTime < timeSlotDuration$ **do**
  $dc\_tx \leftarrow$ first item of DCs
  while $dc\_tx \neq$ last item of DCs **do**
    $dc\_rx \leftarrow$ last item of DCs
    if dc_tx has VMs that can be migrated **then**
      while $dc\_tx \neq dc\_rx$ **do**
        evaluate if it is worth to migrate VMs from dc_tx to dc_rx using Alg. 3
        $dc\_rx \leftarrow$ previous DC of DCs
      end while
    end if
    $dc\_tx \leftarrow$ next DC of DCs
  end while
  if no VM migration was planed and no migration is in execution **then**
    exit
  end if
  $plannedTime \leftarrow$ instant after the expected end of next migration
end while

Algorithm 3 Detailed migration planning between two DCs.

Require: dc_tx, dc_rx
$VMs \leftarrow$ list of VMs of dc_tx
for vm in VMs **do**
  $origin \leftarrow$ server where the vm is running
  $target \leftarrow$ server from dc_rx being evaluated
  $worth \leftarrow brownMig < brownNotMig$
  $e\_time \leftarrow$ estimate the migration time using Alg. 1
  $band\_ok \leftarrow$ links between $source$ and $target$ can receive the additional load of the migration
  if $worth$ and $band\_ok$ **then**
    registers the planning of the migration and updates the link history
  end if
end for

target server can receive the additional load of the migration without violating their bandwidth constraint; iii) the VM migration finishes during the current time slot (if the destination server is off, the time to turn the server on is taken into account); and iv) performing the migration reduces the expected brown energy consumption. If all these restrictions are respected, the VM is planned to migrate, and the algorithm registers in the *linkHistory* that the links that connect these two servers will be used until the instant when the VM migration is expected to finish in order to compute the available bandwidth of the network links. If there are still remaining VMs that could be migrated from DC M, the algorithm will try to migrate to the DC N-1, and so on until all the VMs from DC M are planned to migrate, or all the DCs were processed. After finishing processing for the DC M, the algorithm repeats the same process for the DC M+1 until all the DCs are processed.

After evaluating all the possible migrations for the instant at the beginning of the time slot, the algorithm will use the link usage history to obtain information about when there will be available links in term of bandwidth, that is, at what instant of time `t` the next migration is expected to finish. If there are still VMs that could be migrated, the migration planning algorithm will be executed again using the availability of the bandwidth at instant *t*. Then, the process repeats until there are no more VMs to migrate or the evaluation time reaches the end of the time slot.

Regarding the server consolidation, there are two differences to the original NEMESIS algorithm: i) it will only be applied to the DCs that didn't have inter-DC migrations planned to avoid generating network congestion — given that the intra-DC migrations could use the same network links as the inter-DC migrations planned in the previous step; and ii) the migrations are distributed in time using the estimation computed with Algorithm 1 to avoid overlapping them.

The new algorithm is called c-NEMESIS, where the "c" stands for congestion and its full name is "Congestion and Network-aware Energy-efficient Management framework for distributEd cloudS Infrastructures with on-Site photovoltaic production",

an extension of the NEMESIS algorithm with modifications in the steps “migration of the running VMs” and “server consolidation”.

The computational complexity of the Algorithm 3 is $O(n_{VMS} \times n_{servers} \times n_{links})$, where n_{VMS} is the number of running VMs on the DC that is sending the VM, $n_{servers}$ is the number of candidate servers that have the least possible amount of free cores to run the VM in the destination host, and n_{links} represents the number of links that interconnects the VM's host of origin and the target host. For Algorithm 2, the computational complexity is given by $O(n_{DCs} \log n_{DCs} + {n_{DCs}}^2 \times n_{VMS} \times n_{servers} \times n_{links})$, where n_{DCs} is the number of DCs.

3.3.1 Energy cost of migrations

The energy consumption of a live VM migration is modeled in NEMESIS by a computation task that is executed in the target host. This task uses 100% of a single CPU core during the migration process. If the migration is impacted by network congestion, its duration will increase resulting in wastage of energy both from the server where the VM was initially running as in the target server.

The wasted energy is proportional to the extra time migrating, that is, the difference between the duration of the migration process compared to the duration that it would take if there were no network congestion. A lower bound for the wasted energy can be computed with Algorithm 4 using as input this extra time migrating. Given that the migration planning uses a “follow-the-renewables” approach to move the workload from the DCs that are using brown power at that instant to the DCs that have more available green energy, the algorithm computes individually the wasted energy in the server that is sending the VM and in the server that is receiving the VM. For the server that is sending the VM, this extra energy consumed is brown(er), increasing the overall brown energy consumption of the cloud, since this server could be turned off to save energy. For the server that will receive the VM, this

extra energy is green(er), and it is wasted because it was only available at that instant of time (no energy storage devices) and could have been used to execute the workload.

Algorithm 4 Extra energy consumption of migrating.

$pCore \leftarrow 20.5$
$wastedOrigin \leftarrow 0$
$wastedTarget \leftarrow 0$
for mig in $Migrations$ **do**
  $vmCores \leftarrow$ amount of cores of the VM being migrated
  $extraTime \leftarrow mig.Time - mig.TimeNoCong$
  if $extraTime > 0$ **then**
    $wastedOrigin += extraTime \cdot pCore \cdot vmCores$
    $wastedTarget += extraTime \cdot pCore$
  end if
end for

The value of *pCore* is an estimate for the additional cost of executing a core, and is obtained as follows: the server consumes $220\,\mathrm{W}$ at full load and subtracting the power consumption of the IDLE state ($97\,\mathrm{W}$) it results in $123\,\mathrm{W}$. Finally, this value is divided by the total amount of cores of the server (6), resulting in $20.5\,\mathrm{W}$. Notice that *pCore* is only multiplied by the number of cores of the VM for the server of origin, since it remains executing the VM until the end of the migration, and in the destination server the computational task uses only a single CPU core.

3.4 Experiments

The experiments were performed using computational simulations. The Simgrid [8] (version 3.28) framework was used to develop the simulations because it allows modeling distributed computing experiments, as cloud platforms, in particular: energy consumption of executing the tasks, turning on and off the servers, network usage by the live-migration process, network topology and the network congestion. Other relevant reason for choosing the Simgrid framework is the fact that it is well validated by the scientific community with over 20 years of usage. For the network,

it was used the default flow-level TCP modeling of Simgrid that produces precise results for large distributed computing scenarios (as in our case with thousands of servers) in a reasonable amount of time. It is possible to use packet-level simulation, however, despite being more precise, it would result in huge execution time for the simulations [31]. Regarding the cloud infrastructure, it is considered the adapted version of the Grid'5000 (same as NEMESIS, see Section 3.2.1). Finally, it was simulated one week of the multi-cloud operation.

3.4.1 Workload

Traces from real cloud providers were used to generate the workload for the experiments. The data extracted from the traces were: the number of CPU cores requested, the instant of time when the task was submitted, and its duration. Regarding the requested RAM per VM, it is considered that each VM will consume 2 GB per core requested, similar to the `t2.small` instance of Amazon EC2[3]. It is also considered that the VMs executes with a full CPU usage of the requested cores, the worst scenario for energy consumption. The workloads are scaled to use at a maximum of 80% of the computational resources of the cloud platform at any given simulated time. This decision ensures that the tasks will always be allocated to the servers, and that there will be some room for the servers to receive VMs from live-migrations, thus allows analyzing the different scheduling approaches. Furthermore, from the cloud traces it is possible to observe that the DCs are not used 100% of its computational capacity. Figure 3.2 illustrates the number of VMs submitted during the week (in yellow), and the cumulative demand of CPU cores requested at a given time (in purple), that is, the total number of CPU cores used by the VMs in execution at that instant of time.

[3] `https://aws.amazon.com/ec2/instance-types/`

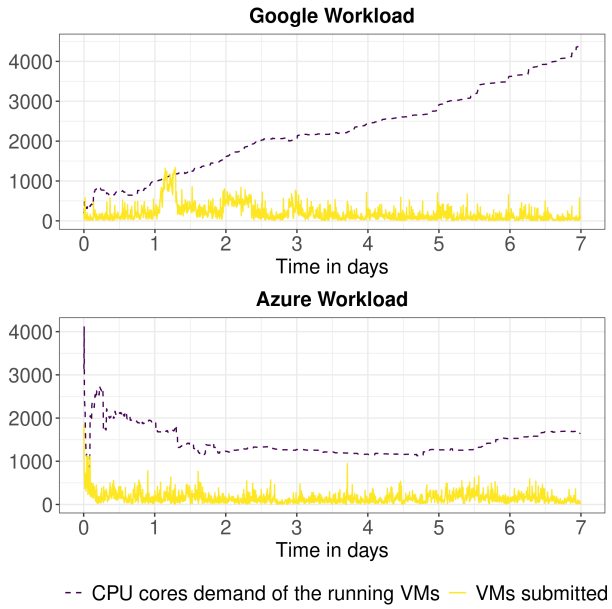

Figure 3.2 Workloads used as input for our simulations.

The first workload was generated using as base the 2011 Google Cluster Workload traces [26], and it consists of over 380,000 VMs. In this workload, the VMs have a long execution time, as can be seen in Figure 3.2 that the value of the running VMs' CPU cores demand keeps increasing throughout the week. The second workload was based on traces from Microsoft Azure [14], more specifically the Azure Trace for Packing 2020, and contains over 304,000 VMs. This generated workload has a different behavior than the previous one: there is a peak in the VM submissions at the beginning of the week, and the VMs have a shorter execution time, as seen that the CPU cores demand does not keep increasing during the week.

These traces do not provide data about the network usage of the tasks and it was not modeled in the simulations. The additional load in the network is generated only by the live migration of the VMs. Therefore this work can be seen as a lower bound for the real world scenario.

3.4.2 Green energy traces

The data for the energy produced by the solar panels were obtained from the Photovolta[4] project by *Université de Nantes*. The data represents the power production of two arrays of 4 PVs each (Sanio HIP-240-HDE4) with a total nominal power production of 1920 Wc at intervals of 5 minutes. In order to simulate the different production of the geographically distributed DCs, a different week of the production trace was used for each DC. Furthermore, each DC had 3 PV panels per server. The PVs installed in the DCs generated the following amount of energy during the simulated week: i) Grenoble: 1.58 MWh; ii) Lille: 0.07 MWh; iii) Luxembourg: 0.15 MWh; iv) Lyon: 1.19 MWh; v) Nancy: 2.16 MWh; vi) Reims: 0.38 MWh; vii) Rennes: 1.63 MWh; viii) Sophia: 1.75 MWh; and ix) Toulouse: 1.53 MWh. In total, around 10.5 MWh of green power was produced in the simulated week. Figure 3.3 shows the green power production per DC during the simulated week.

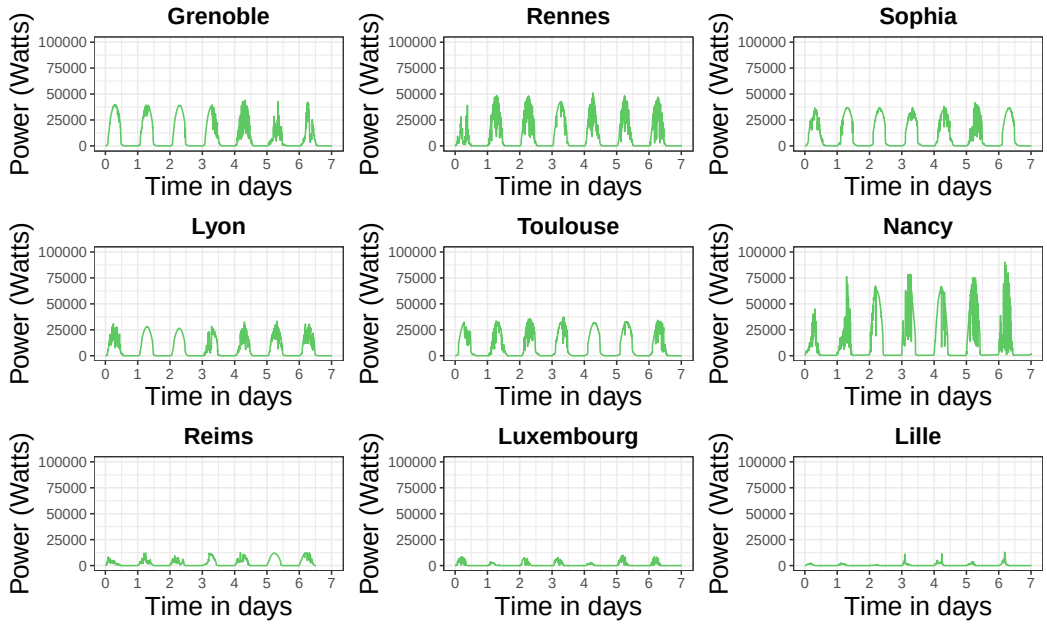

Figure 3.3 Green power production (in Watts) produced by DC during the simulated week.

[4] http://photovolta.univ-nantes.fr/

3.5 Results

In this section, the results of the simulations are presented. First, it is shown an analysis of the accuracy of an essential component of NEMESIS and c-NEMESIS algorithms: the algorithm for the duration of migration (Algorithm 1. Then, it is presented an analysis of the live migrations performed and their impact on network congestion, wasted energy, and the total and brown energy consumption. To further evaluate the effectiveness of using "follow-the-renewables approaches", results from two other state-of-the-art works (at the time the article was written, november of 2021) are presented that incorporate this approach in different ways.

The first is the Workload shifting non brownout (WSNB) algorithm [35]. The algorithm works as follows. Initially, the VM is allocated to the nearest DC (aiming to reduce the response time) and to the server that will increase the energy consumption by the least (a server that is already on and that its available computational resources are equal or slightly greater than the requirements of the requested VM). Then, if this initial DC does not have available renewable power to execute the VM, the algorithm will evaluate another DC (the DCs are sorted by available green energy) that match this demand and do not exceed a threshold for the response time. If another DC is found, the VM will be reallocated to it. The algorithm does not perform server consolidation with live-migrations, it tries to use the minimum number of servers during this allocation. To adapt this baseline for the experiments, the response time restriction was removed from the algorithm, that is, we consider that all the DCs are homogeneous in terms of latency for the VM request, since the used workloads do not have this data, and this modification does not changes the behavior of the algorithm.

The second work from state of the art is the FollowMe@Source (or FollowMe@S) algorithm [3] that has two variants: i) FollowMe@S Intra, that only perform VM migrations inside the same DC (intra-DC migrations); and ii) FollowMe@S Inter,

that only perform VM migrations between different DCs (inter-DC migrations). Both algorithms have the same general steps, described as follows.

The FollowMe@S algorithm has two main steps: allocation and migrations. The allocation step begins by sorting the DCs by the availability of green energy. Then the algorithm will search for a server that can host the VM respecting the sorted list of DCs. If no server was found after searching through all DCs, the VM will be processed in the next scheduling round. The process will be executed again for all the other VMs to be scheduled. In the migration step, either only intra-DC or inter-DC migrations are performed. First, the algorithm obtains the list of all the VMs that are in execution at that instant of time. Then it evaluates the usage of the servers to obtain the underutilized hosts (less than 20% of CPU usage) and mark them to be turned off. The running VMs of these underutilized hosts will be removed using ive migrations in order to turn them off and save energy. In the intra-DC case (FollowMe@S Intra), the algorithm will run in each DC separately. For each VM, it will try to find a server with available computational resources to host it, and the migration is performed if a server is found. In the inter-DC case (FollowMe@S Inter), for each VM that can be migrated, the algorithm will evaluate the DCs sorted by the availability of green energy, and migrate the VM to the server of the first DC that can host the VM. One important point to notice is that FollowMe@S does not consider the network to schedule its migrations. The following modifications were performed in the algorithm to adapt it for the experiments: i) it is not considered the network costs of a distributed algorithm, since NEMESIS, c-NEMESIS, and WSNB are centralized algorithms; ii) the workload degradation performance by migrating the VM — for example migrating to a server with a less efficient CPU — is not modeled as a homogeneous platform is used.

The baselines adopts the “follow-the-renewables” approaches in different ways. The algorithms WSNB and FollowMe@S Intra only use “follow-the-renewables” for the initial scheduling, and the VMs are not migrated to other DCs during their execution.

On the other hand, the algorithm FollowMe@S Inter performs VM migrations to other DCs — same strategy adopted by NEMESIS and c-NEMESIS.

A public Git repository hosts all the code for the simulations, the traces for the workloads and green power production, and the instructions to run and extract the results[5]. Finally, these algorithms and workloads are deterministic, therefore only results for a single execution of the simulations are presented.

3.5.1 Accuracy of the estimation algorithm

We define an underestimated migration as a VM live migration process whose duration was longer than what was estimated. Table 3.1 presents the number of live migrations that were underestimated by NEMESIS and c-NEMESIS, and the percentage value that is based on the total number of migrations (that can be found in Table 3.3). For both workloads, c-NEMESIS presented almost no underestimation for the migrations in comparison to NEMESIS for both workloads. Furthermore, it is interesting to notice that virtually all the intra-DC migrations were underestimated in NEMESIS because they start simultaneously, resulting in network congestion.

Table 3.1 Number of underestimated live migrations and the ratio of the overestimation, where "W" stands for "Workload", "G" for Google, and "A" for Azure.

Algorithm	W	Inter	Intra
NEMESIS	G	245 (4.4%)	3393 (95.7%)
c-NEMESIS	G	0 (0%)	61 (2.9%)
NEMESIS	A	140 (3.6%)	3324 (94.1%)
c-NEMESIS	A	24 (0.1%)	49 (3.8%)

In order to evaluate the estimation algorithm for the duration of the migrations, two metrics are used to assess its accuracy: the Mean Absolute Percentage Error (M.A.P.E.) and the Root Mean Square Error (R.M.S.E.). The M.A.P.E. is defined by: $\frac{1}{n}\sum_{i=1}^{n}\frac{|R_i-F_i|}{R_i}$, where n represents the amount of values being considered, i the index of the value being considered, R_i the real duration of migration, and

[5] https://gitlab.com/migvasc/c-knemesis

F_i the estimated duration. The results of the M.A.P.E. is a percentage value, and it represents the relative value of the estimation errors compared to the original value, thus it is a metric easy to understand metric. The R.M.S.E. is defined by: $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(R_i - F_i)^2}$. The R.M.S.E. is a metric similar to the standard deviation, and it allows to validate how far from the original value was the estimation.

Table 3.2 Accuracy measurements, where "W" stands for "Workload", "G" for Google, and "A" for Azure. The M.A.P.E. value is in percentage (%), and the R.M.S.E. in seconds.

Algorithm	**W**	**M.A.P.E.**	**R.M.S.E.**
c-NEMESIS	G	0.70	0.395
NEMESIS	G	32.895	18.56
c-NEMESIS	A	0.649	0.432
NEMESIS	A	34.01	20.03

Table 3.2 presents the results for the accuracy measurements. It is possible to observe that c-NEMESIS is accurate with low error values. The improvement in the precision of the estimation algorithm is justified by two points: i) the new algorithm has full information of the network topology and uses the actual number of links that interconnect the servers involved in the migration process; ii) since it is more precise, the migration planning results in less network congestion, which affects the duration of the migration.

3.5.2 Analysis of the VM live migrations performed

Table 3.3 presents the number of live migrations performed during the simulated week. NEMESIS performed the lowest number of inter-DC migrations for both workloads, since its migration planning does not allow for migrations in parallel leaving or arriving at the same DC. The c-NEMESIS algorithm executed more inter-DC migrations than NEMESIS, as it takes into account the network topology and allows for migrations in parallel. c-NEMESIS had the lowest number of intra-DC migrations for two factors: i) the server consolidation step is only executed for the DCs that do not have inter-DC migrations planned in time slot; and ii) intra-DC

migrations are distributed in time. The FollowME@S algorithm had the highest number of inter and intra-DC migrations because the planning does not consider network usage.

Table 3.3 Number of VM live migrations performed, where "W" stands for "Workload", "G" for Google, and "A" for Azure.

Algorithm	W	Inter	Intra
NEMESIS	G	5617	3545
c-NEMESIS	G	18056	2074
FollowMe@S Intra	G	0	560862
FollowMe@S Inter	G	96464	0
NEMESIS	A	3863	3532
c-NEMESIS	A	17479	1300
FollowMe@S Intra	A	0	177086
FollowMe@S Inter	A	93388	0

3.5.2.1 Total and brown energy consumption of the cloud platform

Table 3.4 presents the total and brown energy consumption of the cloud platform for the simulated week. The c-NEMESIS performed the better in terms of brown energy usage, having the lowest consumption among the other evaluated algorithms — with exception for NEMESIS using the Azure workload, in which the consumption was the same. Regarding the total energy, c-NEMESIS consumed more than NEMESIS because it performed more migrations. On the other hand, more green energy was harnessed, since the brown energy consumed was the same or lower. Regarding the FollowME@S algorithm, both inter and intra versions had similar brown energy consumption, but the inter-DC approach had a marginally lower brown energy consumption than the intra-DC (around 0.3% for the Google workload and 0.04% for the Azure workload). The WSNB algorithm had the highest consumption of both total and brown energy.

The difference between the total and brown energy consumed with the green energy generated in the simulated week (10.5 MWh) is compared to obtain the green energy usage of the algorithms. For the Google workload, the usage of green energy was:

Table 3.4 Comparison of energy consumption (MWh), where "W" stands for "Workload", "G" for Google, and "A" for Azure.

Algorithm	W	Total	Brown
NEMESIS	G	25.46	17.23
c-NEMESIS	G	25.56	17.18
FollowMe@S Intra	G	27.56	19.13
FollowMe@S Inter	G	27.59	19.07
WSNB	G	29.49	20.89
NEMESIS	A	30.43	21.21
c-NEMESIS	A	30.55	21.20
FollowMe@S Intra	A	31.69	22.41
FollowMe@S Inter	A	31.69	22.40
WSNB	A	33.56	24.23

c-NEMESIS = 80%, NEMESIS = 79%, FollowMe@S Intra = 80%, FollowMe@S Inter = 81%, and WSNB = 82%. Regarding the Azure workload, the usage was: c-NEMESIS = 89%, NEMESIS = 88%, FollowMe@S Intra = 89%, FollowMe@S Inter = 89%, and WSNB = 89%.

The scheduling policies and how the algorithms adopt the "follow-the-renewables" strategy justify the difference between the total and brown energy consumption, and the relative value of renewable energy used. The algorithms WSNB and FollowMe@S Intra that presented the highest brown energy consumption applied "follow-the-renewables" at the initial scheduling of the workload, and didn't migrate the VMs in execution to other DCs as the green energy availability changed over time — as did by the algorithms FollowMe@S Inter, NEMESIS and c-NEMESIS. To further understand the difference in these results, the next section will analyze the live migrations' impact on the total and brown energy consumption.

3.5.2.2 Wasted resources in the migrations

To evaluate wastage of resources in terms of network and energy, all the live migrations performed in the algorithms are compared with a perfect scenario: all

the migrations are performed individually and isolated, having full access to the network.

Table 3.5 presents statistics about the extra time (in seconds) it took to migrate the VMs in the simulations in comparison with the perfect scenario. The absolute value is the absolute difference in seconds. For example, on average, the live migrations performed by the NEMESIS algorithm was 10.11 seconds longer compared to the perfect scenario for the Google workload. The relative value is the ratio of the difference. For example, in the FollowMe@S Inter with the Google workload, the migration duration was more than 10 times longer in comparison with the perfect scenario.

Table 3.5 Extra seconds during migrations compared to the case when there is no congestion, where "W" stands for "Workload", "G" for Google and, "A" for Azure. "avg." for the average of the observations, "max." for the maximum value, "abs." for the absolute value, and "rel." for the relative value.

Algorithm	W	avg. abs.	max. abs.	avg. rel.	max. rel.	Total seconds
NEMESIS	G	10.11	56.98	1.53	3.98	92915.7
c-NEMESIS	G	0.22	0.88	1.0	1.05	4331.93
FollowMe@S Intra	G	113.94	1028.51	6.37	40.82	64525098.3
FollowMe@S Inter	G	215.18	4875.8	10.67	155.5	22058949.9
NEMESIS	A	11.62	62.96	1.59	3.98	86235.46
c-NEMESIS	A	0.23	6.14	1.0	1.32	4224.43
FollowMe@S Intra	A	91.08	938.48	4.39	25.56	16384188.8
FollowMe@S Inter	A	186.56	8047.89	7.8	157.24	18531893.3

The c-NEMESIS algorithm had the best performance in terms of extra seconds spent migrating for both workloads, with values close to the perfect scenario: the average relative difference (avg. rel.) was approximately 1. The NEMESIS algorithm had a good performance as well with low extra seconds values, and the duration of the migrations took, on average, about 1.5 more than in the perfect scenario. The migrations performed by FollowME@S, both intra and inter-DC cases, had the worst performance: the duration took, on average, from 4 to 10 times more than the perfect scenario. These results highlight the importance of considering the network

for the migration scheduling, since c-NEMESIS and NEMESIS presented the lowest wastage of resources.

Using the extra seconds spend in the migrations, a lower bound for the wasted energy is computed using Algorithm 4 for both the servers that sent the VM (origin) — that represents the brown(er) energy consumption — and the servers that received the VMs (target) — that represents the green(er) energy waste. Table 3.6 presents the results. The algorithm FollowMe@S (both inter and intra-DC versions) was the algorithm that most wasted energy. The c-NEMESIS algorithm had the lowest wastage of energy overall, despite performing more migrations than NEMESIS (that was the second better in terms of wastage of energy). These values are justified by the fact that wasted energy is directly proportional to the duration of the migrations, thus bad planning will congestion the network and as consequence increase the total energy consumption. Finally, it is important to notice that the green wasted energy is not negligible and can be used to power the cloud, for example, in the FollowMe@S intra-DC with the Google workload, the wasted green energy could have powered the Luxembourg DC for approximately 44 hours (at 100% computational performance).

Table 3.6 Wasted energy in the migrations (Wh) in comparison to the perfect scenario where the migration process have full access to the network links. In the table "W" stands for "Workload", "G" for Google, and "A" for Azure.

Algorithm	W	Origin	Target
NEMESIS	G	545.1	529.1
c-NEMESIS	G	35.42	24.67
FollowMe@S Intra	G	473971.6	367434.6
FollowMe@S Inter	G	153829.96	125613.5
NEMESIS	A	539.59	491.06
c-NEMESIS	A	39.31	24.06
FollowMe@S Intra	A	163128.14	93298.9
FollowMe@S Inter	A	175086.3	105528.8

3.6 SUMMARY

Reducing the environmental impact of the operation of cloud computing platforms, in particular the carbon footprint generated from the country-like the energy consumption, is a complex and challenging problem currently being addressed from multiple angles. In this chapter, we focus on the strategy “follow-the-renewables”, and studied the indirect impact on the energy consumption caused by the additional load generated in the network. The experiments were based on real-world data for the cloud infrastructure, workloads, and photovoltaic power production.

This chapter demonstrates that the indirect network impact on the energy consumption in multi-clouds for “follow-the-renewables” approaches is generated by bad scheduling policies for the migrations, which results in network congestion — as the live migrations compete for the available bandwidth of the network links, affecting not only the applications that are running inside the VMs, but as well the duration of the migration process. Given that migrating a VM also consumes energy — proportional to its duration — and that the workload will be sent to the DCs using green(er) energy (“folllow-the-renewables”), the extra energy consumption is in reality wasted energy that could be used to power the cloud platform. Furthermore, the adoption of the “follow-the-renewables” needs to consider the whole execution of the workload: the state-of-the-art algorithms that only used the green energy information for the initial scheduling and didn’t migrated the workload as the availability of renewable energy changed had the highest brown energy consumption.

The proposed estimation algorithm for the duration of the live migrations that uses as input information about the communication network of the data centers (network links latency, bandwidth and topology) is accurate. By using this estimation algorithm and using as input the network characteristics of the cloud, the migration planning of c-NEMESIS was able to increase the number of migrations by a least

3-fold without network congestion, while maintaining or reducing the brown energy consumption in comparison to other state-of-the-art works.

As future research directions, the network usage by the workload and how it will compete for network resources with the live migrations needs further investigation. It is also possible to extend the proposed solutions for other virtualization techniques as containers by updating the estimation algorithm with a model for container live migration. Finally, there are approaches that explore turning off the network devices to save energy. This technique reduces the available network links in the cloud platform and the network traffic will be re-routed, and it is necessary to analyze if the energy savings are more significant than the impacts caused by the network congestion.

4 Sizing the renewable infrastructure to greener the DC operation[1]

4.1 Introduction

On the previous chapter, we saw the adoption of the carbon-aware technique follow-the-renewables applied to a multi-cloud distributed over a country, and the drawbacks it presents if not used properly, as network congestion and extra energy consumption. Now, the reader will be presented to another strategy: sizing (or dimensioning) the renewable infrastructure to reduce the carbon footprint of the cloud operation. This strategy consists in defining how much investments needs to be made, for example defining the area of solar panels (when considering solar power), the number of wind turbines, and the capacity of the energy storage devices (as lithium-ion batteries) that needs to be manufactured.

The scenario considered has some differences to better represent modern cloud providers. First, it is considered that the DCs are geographically distributed over the world, which represents real cloud providers as Google, Microsoft, Facebook and so on, as can be seen in Figure X. Many reasons justifies the need for geographically distributed DCs: i) meet the demand for the ever increasing number of users; ii) redundancy, for example if there is a problem in some region the workload can be shifted to another location; iii) reduce the latency or response time for the users. This geographic distribution also allows to better harvest the renewables sources, since there is a more variable climate conditions, solar irradiation levels is different

between the countries, as well as the wind received. Second, many locations in the world have the presence of low carbon intensive sources in their local electricity grid, therefore in reality there is not only green and brown classification (as seen in the previous chapter) but many shades of the colors. Classifying between brown and green made sense in the previous scenario since it was considered only a single country, and the renewable energy was less carbon intensive than the local electricity grid. Finally, energy storage devices can be used to store the overproduction of renewable energy and use it when opportune.

One point that cannot be neglected is that manufacturing the renewable infrastructure also presents a environmental impact: batteries have an ideal level of charge that can improve their lifetime, which can reduce the replacement frequency, but on the other hand, causes them to be oversized [4], and their recycles rates still need to improve [24]; and considering the current state-of-the-art PVs, if they produce 40% of the global electricity by 2050, they will consume about 5% of today's CO_2 budget [32].

In this chapter, we explore the adoption of both strategies, sizing the PVs and batteries and scheduling with follow-the-renewables to reduce the carbon footprint of operating existing cloud platforms. More specifically, this chapter presents the following contributions:

- the two sub-problems—PVs and batteries sizing, and workload scheduling—are modeled as a single problem, which allows evaluating scenarios such as: should the battery capacity or the PV area be increased, or should the workload be scheduled in a data center located in another part of the world?

- we propose a model that uses a linear programming approach (LP) with real variables, allowing us to optimally solve the problem we address in polynomial time using classical LP solvers. This allows a large number of scenarios to be evaluated over broad time horizons (i.e., one year) to take

the seasonal behavior of renewable energy production into account. This model can be extended to multiple scenarios, and it may help decision-makers evaluate which regions need more investment to reduce the cloud operation's environmental impact.

The remainder of the chapter is organized as follows: Section 4.2 defines the problem addressed, the assumptions, the models, and the objective function. Details about the problem constraints and how to optimally solve the problem are given in Section 4.3. Comprehensive experiments, presented in Section 4.4, are discussed in Section 4.5 before we conclude in Section 4.6.

4.2 Problem statement

In this section the reader will find a description of the addressed problem and hypothesis. The next two sections further details the modeling, notations, and the optimal approach to solve the decision problem that we tackle within the chapter.

4.2.1 Addressed problem

As mentioned earlier, the goal is to reduce the carbon footprint of an existing cloud platform both in its operation and sizing the renewable infrastructure. The considered cloud platform consists of several data centers spread worldwide, in both hemispheres and on all continents. The solution that will be proposed aims to design an additional solar-based power supply infrastructure to the classical power grid connection and to define an optimal way to operate the global IT cloud platform by scheduling its workload. To green the cloud infrastructure and reduce its carbon footprint, we have to reduce the usage of high-carbon intensive energy to operate

the data centers. Using renewable energy is a promising option, however the carbon footprint of manufacturing needs to be taken into account.

Another problem is that the location where the DC is installed determines how sustainable it can be. First, the carbon intensity of the energy consumed from the local electricity grid depends on how it is produced: natural gas, coal combustion, hydraulic or nuclear energy. Second, each location has different climate conditions and the renewable power production will also have different efficiency, for example, solar power will be higher in locations that receive more solar irradiation which depending if its near or far from the tropics. Therefore, powering the cloud federation is a balance or mix between using low-carbon electricity from the local grid and using its solar panels, with the understanding that batteries are mandatory to mitigate intrinsic solar power intermittency.

The current decision problem aims at defining the additional renewable power supply architecture from solar energy to reduce the carbon footprint of the global cloud infrastructure. It is assumed that: i) the data centers are already in operation and the sizing will only define the area of PV panels and the capacity of the batteries; ii) the DCs can be supplied from power of the local electricity grid, power from the PV panels or power stored in the batteries; iii) job submission is centralized; iv) 100% of the jobs must be completed in time (no delay); v) no migration of jobs; vi) jobs can be executed in any of the data centers; vii) cloud platform is homogeneous regarding the IT part (number of servers, CPU cores and model, network equipment), but the total power consumption of the DCs is different because of the power used for cooling the DCs at each geographic location. Finally, the following inputs are considered:

- specifications of the cloud infrastructure
  - number of servers, CPU cores per server, network switches

  - power consumption of the servers (static and dynamic) and network equipment
  - Power Usage Effectiveness (PUE) to represent cooling needs
- specifications of the renewable infrastructure
  - manufacturing carbon footprint
  - technical parameters: batteries charge/discharge ratio and Max Depth of Discharge, PV efficiency to convert solar irradiation to power
- weather conditions (solar irradiation) in areas where each data center operates for the federation (time series of 1 year with one value for every hour)
- carbon intensity of the local electricity grid in g CO2 eq per kWh for each DC
- the workload computing demand from clients (time series of 1 year with one value for every hour)

Now, the models and notations will be introduced before the objective function to optimize.

4.2.2 Models and notations

To propose a solution in terms of job operations, a decision horizon $\mathcal{H}$ is defined where job scheduling decisions can be taken. To do so, we propose to discretize $\mathcal{H}$ into K indivisible time slots whose duration is Δt such that $\mathcal{H} = K \times \Delta t$. To simplify the notations, we consider $\Delta t = 1 u.t.$ (unit of time). In our experiments, we will assume that $\Delta t = 1h$ such that $K = 8760\,h$ with $\mathcal{H} = 1$ year. Let k be the index of the time slot that addresses any time instant t such that $k\Delta t \leq t < (k+1)\Delta t$ with $0 \leq k < K$.

Table 4.1 Main notations for the IT model for each DC^d $(1 \leq d \leq D)$ during time slot k $(0 \leq k < K)$

Δt	time duration of each time slot in unit of time $[u.t]$
$\mathcal{H}$	decision horizon $\mathcal{H} = K\Delta t$
K	number of time slots $\Delta t = 1\,\text{h} = 1\,u.t.$
k	time slot between dates $k\Delta t$ and $(k+1)\Delta t$ excluded
DC^d	a specific data center d of the cloud federation
$\mathcal{DC}$	the set of all data centers $\{DC^d \mid d = 1,\ldots,D\}$
C^d	number of CPU cores within DC^d
$Pcore$	dynamic power consumption of one CPU core at 100% of utilisation
$Pidle^d$	static power consumption of all the servers of DC^d
$Pintranet^d$	power consumption of network devices of the DC^d
P_k^d	the power demand to perform tasks on DC^d during time slot k
PUE^d	Power Usage Effectiveness of data center d
$\mathcal{T}$	the workload to perform $(= \{T_i \mid 1 \leq i \leq N\})$
T_i	task i of the workload $\mathcal{T}$ $(1 \leq i \leq N)$
r_i	release date of tasks T_i
p_i	processing time of tasks T_i
c_i	number of cores needed to execute task T_i
w_k	number of cores needed during the kth time slot
w_k^d	number of cores needed during the kth time slot on DC^d

4.2.2.1 IT part model

Let $\mathcal{DC} = \{DC^d \mid d = 1, \ldots, D\}$ be the set of data centers in the cloud federation. Considering a given data center DC^d, let C^d be its number of CPU cores and $Pcore$ the energy used to power one core.

The power consumption of a cloud data center can be classified as static or dynamic [2]. For the static part, the current model considers the idle power consumption $Pidle^d$ of the servers, the Power Usage Effectiveness (PUE) to represent the power consumption used to cool the DC infrastructure, and the power consumption of the network switches $Pintranet^d$ that interconnect the servers in each data center DC^d. Regarding the latter, cloud data centers usually adopt the fat-tree topology to interconnect servers in the DC [2]. In this topology, one can compute the number of network switches needed to match the number of servers. The power consumption of the network switches is considered to be static based on actual measurements, which have shown that the consumption does not change significantly with the device usage [18]. Moreover, each geographic location has different cooling needs, therefore each DC has a specific PUE^d value.

Finally, the dynamic part of the power consumption is represented by the additional power demand generated by using the CPU cores in each data center — to execute the workload w_k^d assigned to the DC d at time slot k . Equation (4.1) represents the power consumption of each DC for each step k $(0 \leq k < K)$:

$$P_k^d = PUE^d \times (Pidle^d + Pintranet^d + Pcore \times w_k^d) \tag{4.1}$$

4.2.2.2 Workload model

Considering the workload that needs to be executed, let $\mathcal{T} = \{T_i \mid i = 1, \ldots, N\}$ be the set of N tasks that have to finish in time during the time horizon $\mathcal{H}$. Each task

Table 4.2 Main notations for the Electrical part model of each DC^d $(1 \leq d \leq D)$ during each time step k $(0 \leq k < K)$

I_k^d	solar irradiation at time slot k $[Wh/m^2]$
Apv^d	surface area of photovoltaic panels of DC DC^d $[m^2]$
η_{pv}	PV efficiency (in %) in converting solar irradiation to power
$Pgrid_k^d$	power consumed from the grid at time slot k $[W]$
Pre_k^d	power generated from the PVs at time slot k $[W]$
BAT^d	battery capacity installed in DC^d $[Wh]$
B_k^k	battery level of energy at the time $k \times \Delta t$ $[Wh]$
Pch_k^d	power charged in the batteries during time slot k $[W]$
$Pdch_k^d$	power discharged from the batteries during time slot k $[W]$
η_{ch}	efficiency of the charging process
η_{dch}	efficiency of the discharging process

T_i has the following properties: i) release date r_i; ii) processing time p_i; and iii) needs c_i CPU cores at 100% of usage to be executed. Let w_k be the total number of CPU cores needed to compute tasks during the time slot k in order to complete the workload in time. At each time step, w_k is the sum over all cores required by the tasks executed in time slot k, and given that the tasks will be scheduled to the DCs, w_k^d is the sum of the number of cores used to execute the allocated workload in $\mathcal{DC}$ (with $1 \leq d \leq D$) and $0 \leq k < K$) as shown by Equation (4.2):

$$w_k = \sum_{T_i | r_i \leq k\Delta t < r_i + p_i} c_i = \sum_d w_k^d \tag{4.2}$$

4.2.2.3 Electrical part model

The power supply of the whole cloud platform $\mathcal{DC}$ has three sources: i) the renewable energy generated by the solar panels (PVs) installed on each DC^d site; ii) the classical electrical grid $Pgrid_k^d$ of each country on which $DC^d \in \mathcal{D}$ is hosted — used as backup when the power from the renewable infrastructure is not enough; and iii) energy discharged from the energy storage devices. The storage devices are mandatory to mitigate the intermittency of solar power by either storing the overproduction when the sun shines or to provide energy during the night period when the solar panels does not produce power. Lithium-Ion batteries have been chosen to play this role because of their good efficiency in terms of costs, power and energy density, charge and discharge rates, and self-discharge [34]. To model the fact that the cloud platform will not sell back energy to the grid, there is the constraint that the power from the grid ($Pgrid_k^d$) is always positive.

Regarding the renewable power production, it depends on the solar irradiation I_k^d received at the location of DC^d during the time slot k, on the surface area of the solar panels Apv^d and on the efficiency η_{pv} of the PVs. Equation (4.3) models the on-site renewable power production.

$$Pre_k^d = I_k^d \times Apv^d \times \eta_{pv} \tag{4.3}$$

Batteries are systematically installed next to the PVs for the reason mentioned above. Let B_k^d be the battery level of energy, that is, amount of energy (in Wh) at time kt stored in batteries of capacity BAT^d installed in DC^d ($B_0^d = Binit^d$ being the amount of energy at the beginning the time horizon $\mathcal{H}$). The variable Pch_k^d represents the power charged in the batteries of DC d during the time slot k (and $Pdch_k^d$ is the power discharged from the batteries). It is not possible to charge and discharge simultaneously: if Pch_k^d is greater than zero, $Pdch_k^d$ equals zero and vice versa. Regarding the batteries modeling, the charging and discharging process have

an efficiency η_{ch} and η_{dch} less than 1. Lithium-Ion batteries have a low value for daily self-discharge (0.5 % per day) [34], thus this property was not modeled. Finally, to increase the lifetime, the batteries cannot be discharged more than its Maximum Depth of Discharge.

Equation (4.4) models the battery in terms of the level of energy:

$$B_k^d = B_{k-1}^d + Pch_{k-1}^d \times \eta_{ch} \times \Delta t - \frac{Pdch_{k-1}^d}{\eta_{dch}} \times \Delta t \tag{4.4}$$

with $0.2 \times BAT^d \leq B_k^d \leq 0.8 \times BAT^d$ for any time slot k and DC^d ($0 \leq k < K$ and $1 \leq d \leq D$) — this last restriction models the Max Depth of Discharge property to increase the battery lifespan. The modeling of the batteries and PVs are based on [15].

4.2.2.4 Footprint model

In the current model, carbon emissions of operating the cloud platform originate from three sources: (i) consuming power from the regular electrical grid and manufacturing of both (ii) the photovoltaic panels, and (iii) the batteries. Equation (4.5) models the carbon footprint of the regular local electrical grid, defined by the carbon intensity of the power grid $gridCO2^d$ in the region of data center DC^d times the amount of energy used during the time slot k.

$$FPgrid_k^d = Pgrid_k^d \times \Delta t \times gridCO2^d \tag{4.5}$$

It is considered that the power from local electricity grid may originate from multiple sources, and the $gridCO2^d$ is an input that represents its average carbon intensity during the year: the value will be low if it is supplied by solar, wind power, hydroelectric or nuclear power. On the other hand, the value will be higher if it is supplied by coal, oil, biomass, or natural gas.

For the carbon footprint of the PVs, in order to account for the fact that the amount of solar irradiation received is not homogeneous for different geographic regions, one must also consider the expected power output that PVs can produce over their lifetime relative to the cost of manufacturing. Therefore, the carbon footprint of PVs is also related to the location of each data center. The carbon footprint thus defined is modeled by Equation (4.6):

$$pvCO2^{d} = \frac{FPpv_{1m2}}{expectedEpv^{d}} \tag{4.6}$$

where $FPpv_{1m2}$ is the emissions of manufacturing $1\,m^2$ PV in $g\,\mathrm{CO}_2 - eq$, and $expectedEpv^{d}$ is the expected energy production in Wh that $1\,m^2$ of the PV during its lifetime at the location of DC^{d}. As a result, the unit of this metric is expressed in $g\,\mathrm{CO}_2 - eq.Wh^{-1}$, and so, the total emissions from the PVs are related to its power production, as shown in Equation (4.7).

$$FPpv_{k}^{d} = pvCO2^{d} \times Pre_{k}^{d} \times \Delta t \tag{4.7}$$

Regarding the batteries' carbon footprint $FPbat^{d}$ of the DC DC^{d}, it is related to their capacity BAT^{d} in kWh and carbon emissions of the manufacturing process $batCOS$ in $g\,\mathrm{CO}_2 - eq.kWh^{-1}$, as seen in Equation (4.8). To be consistent with the calculation of $FPpv_{k}^{d}$, $batCO2$ is the share of the carbon footprint of the battery type chosen for a capacity of $1\,kWh$ over the time horizon of $\mathcal{H}$, assuming a battery has a lifetime of 10 years. Thus, given that we are considering 1 year of cloud operation, $batCO2$ is the tenth of the total carbon footprint of the considered battery.

$$FPbat^{d} = BAT^{d} \times batCO2 \tag{4.8}$$

These modifications regarding the lifetime of PVs and batteries were necessary because we are considering only one year of cloud operation. If we use the total carbon emissions for manufacturing the PVs and batteries, the solver will find a solution where there is few to no PV or batteries, because using the regular electrical grid would be less carbon-intensive.

4.2.3 Objective function

Now that the models have been introduced, the objective function can be defined (see Equation (4.9)). It consists of minimizing the carbon footprint of the globally distributed cloud federation in order to reduce as much as possible carbon emissions, which come from both the consumption of electricity from the power grid, as well as from the manufacturing of photovoltaic panels and batteries with k and d defined as follows: $0 \leq k < K$ and $1 \leq d \leq D$.

$$\text{minimize} \sum_{k=0}^{K-1} \sum_{d=1}^{D} (FPgrid_k^d + FPpv_k^d) + \sum_{d=1}^{D} FPbat^d \tag{4.9}$$

4.3 Optimal resolution

The models presented in the previous section consist of several linear equations. We show in this section that constraints governing the use of the globally distributed cloud platform can be expressed as linear expressions. New real variables are introduced to finalize the linear program that needs to be solved to achieve the targeted objective. The solution obtained after solving the linear program is optimal in nature and computed in polynomial time, as long as the variables are not integers. Polynomial time is mandatory if we consider the number of variables needed for a time horizon $\mathcal{H}$ as long as one year. We assume to choose real positive values for all variables even if variables denote discrete objects like cores. Indeed, w_k^d is the

number of cores needed to run tasks on DC^d during time slot k. Considering the size of the cloud with its thousands of cores, the decimal part of each w_k^d can be neglected. Having the solution with more or less than a core on a given DC does not change the order of magnitude for the PV and battery sizing process.

The globally distributed cloud platform that we plan to optimally size, as mentioned in the problem statement (Section **??**), has only one goal: completing a given amount of work during a given year and knowing the weather conditions during the same year. We present a set of constraints that must be respected to make this mission possible. Some constraints are explicit, and some are implicit to avoid the addition of integer variables which would transform this LP into a MILP whose solving process would not scale at all.

4.3.1 Constraints to address the workload

Since the distributed cloud federation configuration is defined a priori by a set of existing cloud DCs at each chosen location, the amount of work to be performed must respect each data center's computational capabilities DC^d. Equation (4.10) expresses that the number of cores that are switched on does not exceed the existing number of cores of DC^d:

$$w_k^d \leq C^d \tag{4.10}$$

4.3.2 Constraints to reach the power demand

The electric part of each DC has to supply the DC power demand using renewable energy (Pre_k^d), from batteries ($Pdch_k^d$ and Pch_k^d) and/or from the classical grid ($Pgrid_k^d$). Equation (4.11) presents the restriction for the power consumption.

$$P_k^d \leq Pre_k^d + Pgrid_k^d + Pdch_k^d - Pch_k^d \tag{4.11}$$

4.3.3 Constraints on batteries

The batteries are defined by their capacity, which is different for each DC and depends on how the intermittency of the renewable energies is managed on each site. The other quantities concerning the batteries depend on the total capacity of the batteries, DC by DC. This allows realistic behaviors for the batteries to be assumed. These limitations concern the practical level of use of the energy stored in the batteries, which cannot be completely emptied, for example. In addition, the power to charge or discharge a battery is also limited by the level of energy remaining in the associated battery, so that it is not possible to reach a forbidden energy level. Equations (4.12), (4.13) and (4.14) express these constraints:

$$0.2 \times BAT^d \leq B_k^d \leq 0.8 \times BAT^d \tag{4.12}$$

$$Pch_k^d \times \Delta t \times \eta_{ch} \leq 0.8 \times BAT^d - B_{k-1}^d \tag{4.13}$$

$$Pdch_k^d \times \Delta t \ / \ \eta_{dch} \leq B_{k-1}^d - 0.2 \times BAT^d \tag{4.14}$$

One may notice that we are not modeling any restrictions for charging and discharging simultaneously. Such restrictions would require the usage of binary variables that would significantly increase the required computational time to find the optimal solution to the problem. We performed experiments with a shorter duration (around 1 month), and the sizing results were the same between both versions: using and not using binary variables. Furthermore, it is possible to calculate an alternative solution for the linear program where there would be no charge and discharge at the same time slot by increasing or decreasing the value of the variables Pch_k^d and $Pdch_k^d$.

4.3.4 Linear program

This following linear program (LP) summarises what has been described before concerning the model that has to be respected to solve the tackled problem. All variables given by the solution obtained after the solving process are used to completely define both the renewable power supply part and the core operating process of the distributed low carbon cloud federation and the way each DC is used time slot by time slot for one year on the considered weather conditions. Comprehensive experiments have been led to highlight the pertinence of the approach. These experiments are shown in the next section, and a discussion is proposed in Section **??**.

$$\text{(LP)} \begin{cases} \text{minimize } \displaystyle\sum_{k=0}^{K-1}\sum_{d=1}^{D}(FPgrid_k^d + FPpv_k^d) + \sum_{d=1}^{D} FPbat^d \\ \\ \text{s.t.} \quad (4.1)\ (4.3)\ (4.4)\ (4.5)\ (4.7)\ (4.8)\ (4.10)\ (4.11)\ (4.12)\ (4.13)\ (4.14) \end{cases}$$

where all variables are positive real variables.

4.4 Experiments

In this section, we present the settings and the results of our experiments. More details for reproducing the experiments can be found in Appendix **??**.

4.4.1 Settings

4.4.1.1 Cloud infrastructure

The servers are homogeneous and based on equipment of real cloud infrastructure: the Taurus server of the Grid'5000 testbed[2]. The servers are equipped with two Intel Xeon E2630 CPUs, with a total of 12 cores. For modeling the power consumption of the servers, real measurements conducted by [2] were considered: in the idle state, each server consumes 97 W, and their maximum power consumption (when using 100% of the 12 cores) is 220 W. The value of Pcore is 10.25 W, and it was obtained by linear interpolation between the power consumption of the idle and the fully used state.

Each data center is equipped with 23,200 servers (and a total of 278,400 cores). This number matches what can be seen in production data centers of major cloud players: Microsoft operates over 4 million servers distributed over 200 DCs [28].

We considered a network with a 48-ary fat-tree topology linked by 2,880 switches with 48 ports each. The power consumption of the switches was based on real measurements by Hlavacs et al. [18]: the HP ProCurve 2810-48G was selected, with 48 ports and approximately 52W per device.

[2] `https://www.grid5000.fr/w/Lyon:Hardware#taurus`

For the location of the data centers, it was based on the real cloud infrastructure of Microsoft Azure[3], and different regions in different continents, hemispheres, and time zones were selected. Figure 4.1 presents the details of the locations.

Figure 4.1 Selected locations for the data centers.

We used values for the PUE inspired by real data from Microsoft Azure for each region: for the Americas, the PUE is 1.17 (DCs São Paulo and Virginia), Asia Pacific has a PUE of 1.405 (DCs Pune, Canberra, Singapore, and Seoul), and for the Europe region, Middle East, Africa the PUE is 1.185 (DCs Johannesburg, Dubai, and Paris) [33].

4.4.1.2 Workload

The workload used was generated using the Grog generator[4], a workload generator based on analysis of properties of the execution trace made available by Google in 2011 [9]. For reproducibility purposes, the parameters regarding the number of tasks were set to 350,000, the duration was 30 days, and it was executed 12 times (1 per month). The tasks have a duration of one hour.

[3] Azure global infrastructure: `https://infrastructuremap.microsoft.com/`.
[4] `https://pypi.org/project/grog/`

4.4.1.3 Photovoltaic power production

Global Solar Horizontal Irradiation (in Wh/m^2) data was collected from the MERRA-2 project [12], since it provides information for anywhere on earth. Figure 4.2 illustrates different solar irradiation of each location throughout the year 2021.

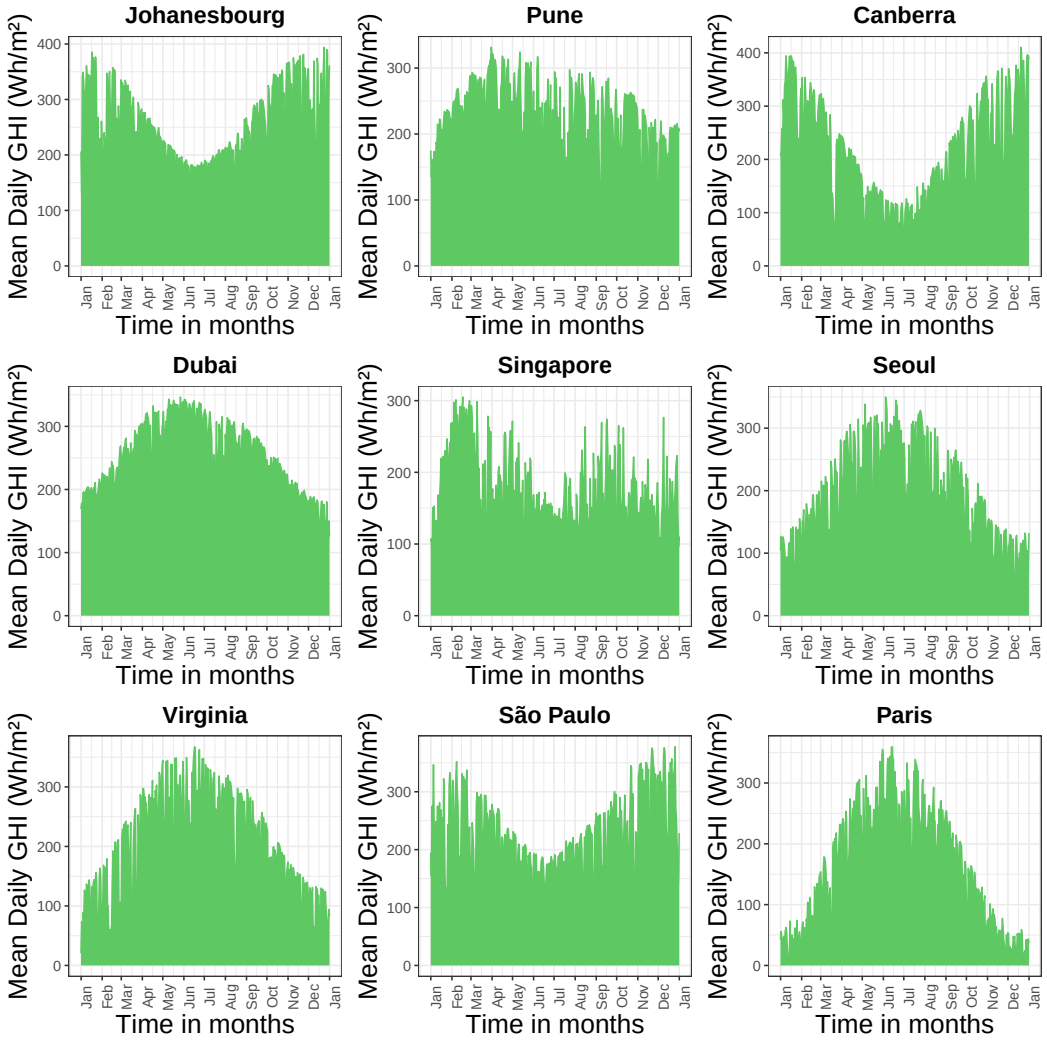

Figure 4.2 Average daily solar irradiation per location throughout the year 2021.

4.4.1.4 Carbon footprint

For solar panels, it is considered a lifetime of 30 years, and manufacturing $1m^2$ emits $250kg\,CO_2 - eq$, inspired from real measurements [36]. To compute the emissions in

Table 4.3 Emissions (in $g\,CO_2 - eq.kWh^{-1}$) for both PV usage and using the regular grid. Source for grid emissions: electricityMap, climate-transparency.org.

Location	Grid	PV
Johannesburg	900.6	24.90
Pune	702.8	27.96
Canberra	667.0	29.71
Dubai	530.0	24.84
Singapore	495.0	36.19
Seoul	415.6	34.00
Virginia	342.8	31.71
São Paulo	61.7	27.99
Paris	52.6	39.93

the form of $g\,CO_2 - eq.kWh^{-1}$ as stated in Section **??**, we considered the total solar irradiation that was produced during the year 2021 multiplied by 30 (to account for the PV module lifetime of 30 years). For the electrical grid, we also considered the real-world data of the carbon footprint ($g\,CO_2 - eq.kWh^{-1}$). Table 4.3 lists the carbon emission values for each region.

Regarding the batteries, the emissions are only considered for the manufacturing step—$59kg\,CO_2 - eq$ per kWh. In our experiments, the considered lifetime of the batteries is ten years. Therefore, the input used is equal to $5.9kg\,CO_2 - eq$ per kWh, given that we simulated one year.

4.4.1.5 Execution environment

We ran the experiments on a machine with an Intel i9-11950H CPU, and 32 GB of RAM. The solver used was the Gurobi Optimizer (version 9.5.2). The execution time for solving the LP with the inputs listed in the previous sections — which resulted in a total of 394,263 variables — was in the order of 30 seconds.

4.4.2 Results

In this section, we present the results in terms of the computed optimal area of the PVs and capacity of the batteries, the source of energy that was consumed by the DCs operation (grid, batteries, or PV panels), and the total emissions of the cloud operation, generated from both manufacturing PVs and batteries, and power consumption of the regular electrical grid. Furthermore, to assess the solution computed by the LP, we compare it with two other scenarios: i) only power from the regular electrical grid is used to supply the DCs (represent current DCs), and ii) only power generated from the PV panels, and stored and discharged from the batteries are used to supply the DCs. Finally, we present an evaluation using metrics to assess the environmental impact of the results.

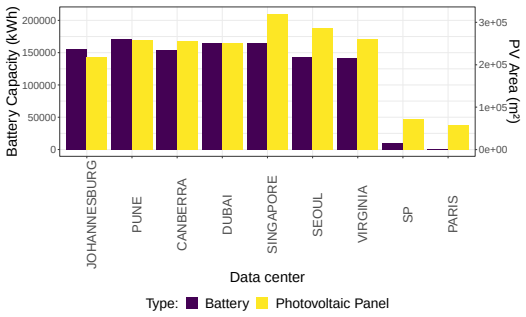

Figure 4.3 Optimal result for the area of PV panels and capacity of the batteries.

Figure 4.3 illustrates the area of the photovoltaic panels and the capacity of the batteries computed from the LP using the inputs described in Section **??**.

To analyze the sources of energy that supplied the DCs operation, we present in Figure 4.4 the percentage that each source (grid, renewable, and batteries) was used to daily supply the DCs throughout the year. Figure 4.5 is a fine-grain visualization of the DC operation regarding the power consumed or produced: it illustrates hour-by-hour the DC total power demand, how much power was consumed from the grid, discharged from the batteries, and produced by the PV panels.

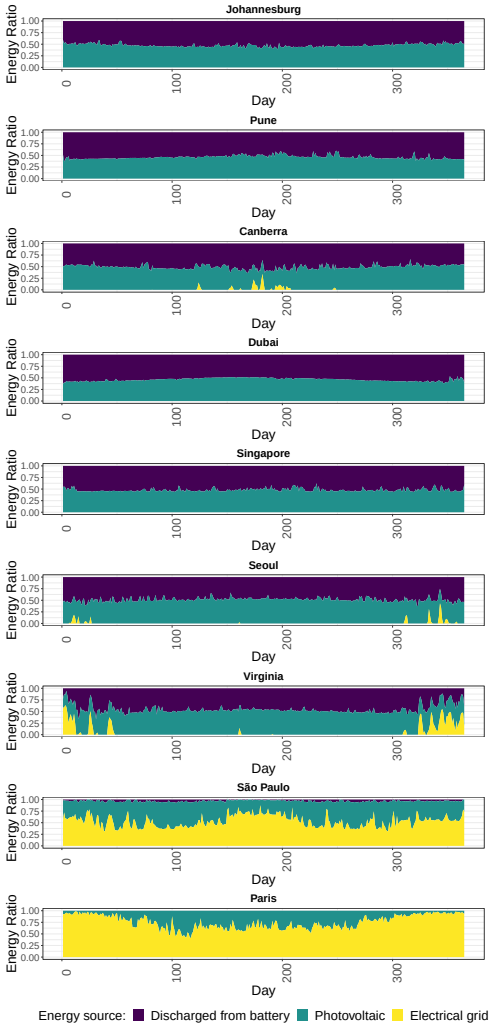

Figure 4.4 Composition of the DCs' daily energy consumption throughout the year considering the different sources of energy, where 1.0 is the DC's total energy consumption.

In order to assess the optimal solution of the LP, we compared it with two other scenarios in terms of total carbon emissions ($t\,CO_2 - eq$): i) the DCs are only supplied by power from the regular electrical grid, and ii) the DCs are only supplied by power from the photovoltaic panels and batteries. Table 4.4 presents the results. In comparison with the first scenario (only grid power), the reduction in the CO_2 emissions was approximately 85%, and it was approximately 30% for the second scenario (only renewable power).

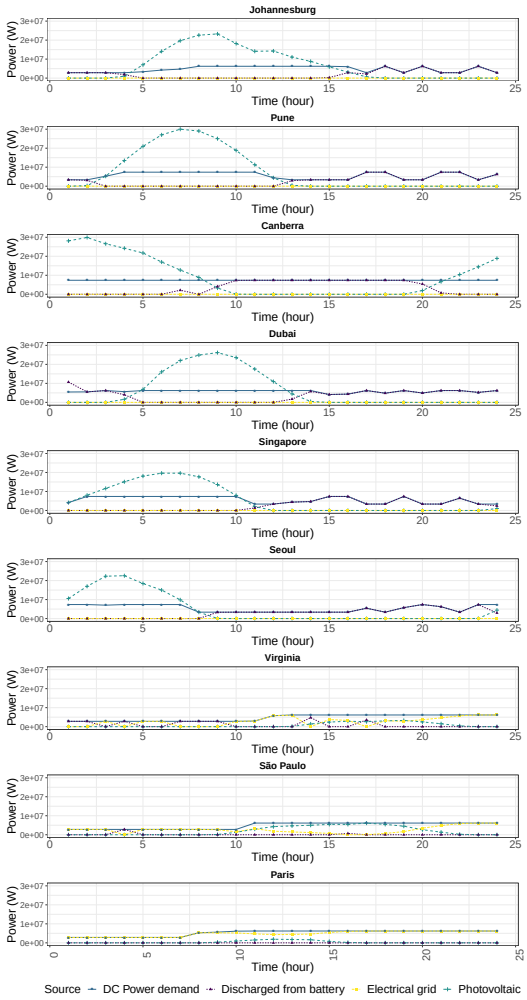

Figure 4.5 Composition of the DCs' hourly power consumption throughout the first day of the year. Time follows the Universal Time (UT) standard.

Table 4.4 Total emissions for the different scenarios.

Scenarios	**Emissions ($t\,\mathbf{CO_2} - eq$)**
Electrical grid	201211.3
PV and batteries	42370.6
PV, batteries, and grid	29600.6

To further evaluate these scenarios, we present in Table 4.5 results in terms of the average load each DC executed throughout the year. Equation (4.15) represents how the metric was computed for each DC d.

$$\frac{\sum_k w_k^d}{C^d \times K} \tag{4.15}$$

Table 4.5 Average DC load throughout the year

Location	Grid	PV + Bat	PV + Bat + Grid
Johannesburg	0	79.31	86.20
Pune	10.25	82.07	89.34
Canberra	99.72	66.62	67.95
Dubai	99.97	93.93	95.11
Singapore	99.93	72.6	85.18
Seoul	99.99	81.87	65.39
Virginia	100.0	88.54	75.51
São Paulo	100.0	63.67	59.06
Paris	100.0	81.24	86.11

To evaluate the environmental impact of the solution, we used metrics extracted from [25]. The first metric, the Green Energy Coefficient (or GEC), is the ratio between the total green power generated and the DC total energy consumption, and it can illustrate the oversizing of the green power supply infrastructure. The second metric is the CO_2 savings, which represents the emissions reduction after DC equipment upgrade or flexibility mechanisms. CO_2 savings is computed as seen in Equation 4.16, where: $CO2_{current}$ represents the system studied after the modifications (the result of the linear program for the sizing of PVs and batteries) and $CO2_{baseline}$ the system in its original state. Here, it was considered that $CO2_{baseline}$ has the same workload allocation of $CO2_{current}$; the difference between the two is that $CO2_{baseline}$ does not have PVs and batteries, and thus only consumes power from the grid. Table 4.6 shows the computed values for both metrics.

$$CO2_{savings} = \left(1 - \frac{CO2_{current}}{CO2_{baseline}}\right) \times 100 \tag{4.16}$$

In order to assess the robustness of the sizing process for the area of PV panels and the capacity of the batteries, it is necessary to take into account other meteorological conditions, given that the DCs will operate for decades and not only for one year. The metric selected is the Mean Absolute Percentage Error (MAPE) defined by: $\frac{1}{n}\sum_{i=1}^{n}\frac{|R_i - F_i|}{R_i}$, where n represents the number of values being considered, i the index of the value being considered, R_i the real value for the year, and F_i the

Table 4.6 Results of the sustainability metrics for the experiments

Location	GEC	CO_2 savings (%)
Johannesburg	1.47	93.93
Pune	1.45	91.5
Canberra	1.57	89.59
Dubai	1.59	89.1
Singapore	1.42	85.75
Seoul	1.53	82.51
Virginia	1.46	75.99
São Paulo	0.5	20.05
Paris	0.24	5.25

estimated value (in this case, the computed sizing for the year 2021 that was used in the experiments). Table 4.7 presents the results of the MAPE for both the area of PV and capacity of the batteries when we solve the LP using as input the solar irradiation for the years 2018, 2019, and 2020. Results indicate a variation of less than 10% in the different DCs over the years.

Table 4.7 Evaluating sizing for different years using the MAPE metric (values are in %)

Location	PV Area	Battery Capacity
Johannesburg	1.72	1.64
Pune	3.72	0.76
Canberra	8.62	4.25
Dubai	2.31	2.88
Singapore	7.22	0.34
Seoul	3.15	1.11
Virginia	2.2	0.87
São Paulo	5.81	8.05
Paris	2.76	0

4.5 Analysis and Discussion

These results permit the evaluation of the carbon footprint impact of different electricity supply policies for Clouds. On the one hand, as shown in Table 4.4, there is a significant reduction to obtain by including renewable energy in the electricity sources of DCs. We observe a 5-fold decrease in the footprint in our experiments.

Many Cloud providers have committed to using 100% renewable energy supplies for their DCs in the following years. On the other hand, this objective of 100% renewable is, in our opinion, more ideological than pragmatic, and there is more benefit to obtain by combining grid and renewable electricity. We observe in our experiments a further reduction of a fourth in the optimal solution compared to the 100% renewable scenario. This study thus gives further insight into the debate of energy sources in Clouds.

The locations used in this paper for the different DCs allow us to benefit from the diversity of latitudes, hemispheres, and climates, as shown in Figure 4.1. This variety of longitudes and hemispheres permits mitigation of the impact of seasonal and daily variations of solar irradiation on electricity production and always has at least some DCs with good PV production, as shown in Figure 4.2. The diversity of climates is highlighted by the case of Singapore's solar production, which is the second lowest with Paris, while its location close to the equator could permit better irradiation.

As indicated in Table 4.3, we observe significant heterogeneity in the carbon footprint of grid electricity of the different DCs, which results in two categories for the optimal solution: i) Paris and São Paulo, DCs with a reduced number of PVs and batteries (no battery in Paris), and ii) the other locations have quite similar sizes of PV and batteries. In the second category, the larger PV area is mainly associated with low solar irradiation. It might appear counterintuitive to allocate more PVs to locations with lower solar production, but this is more comprehensive considering the static part of the power consumption of DCs. When the workload is mainly sent to locations with solar production, the electricity consumption of DC also includes a static part for the idle consumption of servers and the interconnection network, as referred to in Equation (4.1). This static electricity consumption implies either using the carbon-intensive grid or a sizing of PV and batteries that matches the demand, even during winter days of low PV production. This results in a large PV and battery sizing — the PVs are producing up to 1.6 times the DC energy consumption as seen

in Table 4.6 — and, as shown in Figure 4.4, the grid energy consumption of these DCs is very low.

The results for Paris and São Paulo show the carbon footprint of ESDs compared to grid electricity. There is a small benefit in São Paulo for intensive usage, so with reduced sizing, there is no benefit in Paris to using batteries. The PV sizing on these DCs is reduced, probably due to the fact that little energy can be stored in case of overproduction.

The detail of hourly electricity consumption is highlighted in Figure 4.5. The workload is allocated in DCs with PV production. If all this production is used, or the corresponding DCs are full, then the allocation is driven by the battery state of charge, and when none of these possibilities are available, the allocation is for the DC with the lowest grid electricity footprint. For example, in the last hours, the electricity consumption of the different DCs is furnished by battery discharge, in the limit of a state of charge, and the remaining is allocated in the DCs of Paris, São Paulo, and Virginia. Thus, the DC of Virginia consumes grid electricity in two cases: either when Paris and São Paulo DC are full (from hours 10 to 24), or when the DC is empty and only local electricity can be used (hours 3, 5, and 6 in Figure 4.5). The follow-the-sun approach can be partially observed between hour 7 and 8, when Seoul PV production fall and the workload is transferred to Paris, with grid consumption. Then, at hour 10, the same append between PV production in Singapore and grid electricity in São Paulo and at hour 11 between Pune and Virginia. The figure also shows the impact of location, season, and PV sizing on the solar production between Pune and Canberra, large PV production in the best hours, and the tiny production in Paris.

Table 4.5 presents the impact of the different scenarios for energy sources on the load of the different DCs. In the first scenario, the workload is only allocated based on the grid electricity footprint. Thus, we could expect the workload order to be the same as the footprint order per kWh. However, the consumption does not only depend on

the workload but also the PUE of the different DCs. We can thus observe a higher workload in Dubai compared to Singapore, considering that Dubai has electricity with a slightly higher footprint but the lowest PUE. Globally, the range of values highlights the workload variations, requesting at least 4 DCs, at most 8, and most of the time 7. The second scenario considers a model without grid electricity. The allocation is surprisingly distinct from the solar irradiation of the different DCs. For example, the DC of Paris has the lowest yearly irradiation but the median workload in this scenario. Its workload is higher than the one in Johannesburg, which has the second-highest yearly irradiation and is on a similar longitude. The workload is thus not only driven by yearly irradiation. The extremely low PV production in Paris during winter, associated with the static part of the electricity consumption in each data center, implies a high sizing of PV and battery, which lead to a high production during the other seasons that permit a large workload. On Johannesburg, the seasonal variation is lower, so static constraints do not drive PV sizing. Another surprising result is that the DCs with the lowest workload in this scenario are the 4 in the southern hemisphere (including Singapore). This contradicts the intuition of "follow-the-summer" allocation. The case of São Paulo and Canberra could be similar to the one of Johannesburg with the value of the minimal daily production in Virginia and Seoul. The largest workload concerns DCs with the more stable production (Dubai and Pune) and the lowest minimum daily production (Virginia, Paris, and Seoul). Finally, for the last complete scenario, the DCs with the largest workload are the 3 with the largest irradiation (Dubai, Pune, and Johannesburg), followed by Paris with the lowest grid electricity footprint. The only surprise is the workload of São Paulo, which is low considering its low grid electricity footprint and high solar irradiation, and the workload of Singapore, which is high considering its low PV production. Concerning São Paulo, this is probably because it has the second-lowest grid footprint. This implies a low battery sizing, thus a low PV sizing, and finally, it mainly receives workload only when no more DC can provide electricity from PV or battery discharge, and when the DC of Paris is full, that makes many constraints.

Considering Singapore, it is probably due to its position close to the equator, which implies no “winter” season, and its large PV sizing. Finally, the reduction of carbon footprint of each DC between the complete scenario (PV + bat + grid) and the scenario with only grid electricity is showed in Table 4.6. It shows a small decrease in Paris and São Paulo, and a large decrease in the other locations, correlated to the electricity footprint.

4.6 Summary

In this paper, we tackled the problem of greening a distributed cloud data center (DC) federation to lower its carbon footprint. The IT part of the cloud platform already exists, and the idea is to add the equipment on site to introduce renewable energy into the brown energy from the classical grid into the power supply of the DCs. Since the sun is shining everywhere on earth, we have proposed photovoltaic panels (PVs) to produce renewable energy and batteries as storage devices to mitigate the intrinsic intermittency of this energy during the day. The question is how to size the PV array and associated battery size, given an existing federation of DCs distributed around the earth. We have provided a formulation of the problem as a linear program. The particularity of our formulation is that we do not need integer variables; a solution is possible using only real variables given our objective and the context of the problem. As a result, the linear program allows to optimally solve large problem sizes, e.g., minimize the carbon footprint of a nine-site federation, each with its own weather conditions, upon a one-year horizon, hour by hour. We have demonstrated that our program is able to calculate the optimal sizing for PVs and batteries in just a few minutes. Numerous experiments have brought forward results that we have analyzed and discussed to explain what these results express. As an example, an interesting result, depending on the DC locations considered, the optimal solution to reduce the carbon footprint is a hybrid configuration between using PVs and the regular

electrical grid. Moreover, batteries are not always mandatory in each location. As an example, an interesting result, depending on the DC locations considered, the optimal solution to reduce the carbon footprint is to maintain an energy mix through a hybrid configuration including both PVs and a classical grid where the production is low carbon instead of proposing an all renewable platform. Moreover, batteries are not always mandatory in each location. Furthermore Finally, our model has the flexibility to be extended to assess other scenarios (more DCs, other locations, values for carbon emissions, or workloads) and it may help decision-makers build their strategy to reduce the environmental impact of the cloud operation.

In future work, we plan to propose a sizing process that also includes the IT part. Since this investment has been made for years, another perspective is to introduce uncertainty into this sizing process to obtain a more robust distributed DC platform that can provide satisfying service to clients even if the weather conditions change and the submitted workload evolves. The goal always being to remain as virtuous as possible.

General Discussion

5

5.1 General Remarks and Future Research Directions

In this thesis we aimed to ...

contar a historinha dos capitulos,

no fim falar das possiveis work directions para trabalhos futuros ...

5.2 Work Dissemination

Many communications arose from the work performed during this thesis. The following first set presents the communications that are directly related to the contributions presented in this thesis:

- **Danilo Carastan-Santos**, and Raphael Y. de Camargo. "*Obtaining dynamic scheduling policies with simulation and machine learning*". In the International Conference for High Performance Computing, Networking, Storage and Analysis (SC), ACM Press, **2017**. Full paper at the main track of one of the major international conferences in High Performance Computing. This paper comprises the contributions presented in

- **Danilo Carastan-Santos**, Raphael Y. de Camargo, Denis Trystram, Salah Zrigui. "*One can only gain by replacing EASY Backfilling: A simple scheduling policies case study.*". 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), **2019**. Full paper at the main track of

a well recognized international conference in High Performance Computing. This paper comrpises the contributions presented in

The following second set presents the communications that arose in a satellite manner – in the form of collaborations and/or supplementary work – during the thesis. All of these works relate to the thesis subject in some way, either by the HPC resource management or the HPC applications aspects:

- Luis Sant'Ana, **Danilo Carastan-Santos**, Daniel Cordeiro and Raphael Y. de Camargo. "*Real-Time Scheduling Policy Selection from Queue and Machine States.*". 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), **2019**. Full paper at the main track of a well recognized international conference in High Performance Computing.

- **Danilo Carastan-Santos**, David C. Martins-Jr, Siang W. Song, Luiz C. S. Rozante, and Raphael Y. de Camargo. "*A hybrid CPU-GPU-MIC algorithm for minimal hitting set enumeration.*". Concurrency and Computation: Practice and Experience, **2018**. Full paper in well recognized journal in High Performance Computing.

- Luis Sant'Ana, **Danilo Carastan-Santos**, Daniel Cordeiro and Raphael Y. de Camargo. "*Analysis of Potential Online Scheduling Improvements by Real-Time Strategy Selection.*". XIX Simpósio de Computação de Alto-Desempenho (WSCAD), **2018**. Full paper in a recognized Brazilian High Performance Computing workshop.

- **Danilo Carastan-Santos**, David C. Martins-Jr, Siang W. Song, Luiz C. S. Rozante, and Raphael Y. de Camargo. "*A hybrid CPU-GPU-MIC algorithm for hitting set problem.*". XVIII Simpósio de Computação de Alto-Desempenho (WSCAD), **2017**. Full paper in a recognized Brazilian High Performance Computing workshop.

Bibliography

[1] Bilge Acun, Benjamin Lee, Fiodar Kazhamiaka, et al. “Carbon Explorer: A Holistic Approach for Designing Carbon Aware Datacenters”. In: *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (2023) (cit. on p. 6).

[2] Ehsan Ahvar, Anne-Cécile Orgerie, and Adrien Lebre. “Estimating Energy Consumption of Cloud, Fog, and Edge Computing Infrastructures”. In: *IEEE Transactions on Sustainable Computing* 7.2 (2022), pp. 277–288 (cit. on pp. 41, 50).

[3] Hashim Ali, Muhammad Zakarya, Izaz Ur Rahman, Ayaz Ali Khan, and Rajkumar Buyya. “FollowMe@LS: Electricity price and source aware resource management in geographically distributed heterogeneous datacenters”. In: *Journal of Systems and Software* 175 (2021), p. 110907 (cit. on p. 24).

[4] M. Baumann, J. F. Peters, M. Weil, and A. Grunwald. “ CO_2 Footprint and Life-Cycle Costs of Electrochemical Energy Storage for Stationary Grid Applications”. In: *Energy Technology* 5.7 (2017), pp. 1071–1083. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/ente.201600622` (cit. on p. 36).

[5] Manal Benaissa, Jean-Marc Nicod, and Georges Da Costa. “Standalone Data-Center Sizing Combating the Over-Provisioning of the IT and Electrical Parts”. In: *Proceedings of the Workshop on Cloud Computing (WCC)*. 2022 (cit. on p. 5).

[6] B. Camus, F. Dufossé, A. Blavette, M. Quinson, and A. Orgerie. “Network-Aware Energy-Efficient Virtual Machine Management in Distributed Cloud Infrastructures with On-Site Photovoltaic Production”. In: *2018 30th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*. Lyon, France: IEEE, 2018, pp. 86–92 (cit. on pp. 10, 11).

[7] Benjamin Camus, Fanny Dufossé, and AnneCécile Orgerie. “A stochastic approach for optimizing green energy consumption in distributed clouds”. In: *SMARTGREENS 2017 - International Conference on Smart Cities and Green ICT Systems*. Porto, Portugal: SMARTGREENS, 2017 (cit. on pp. 10, 11).

[8] Henri Casanova, Arnaud Giersch, Arnaud Legrand, Martin Quinson, and Frédéric Suter. “Versatile, scalable, and accurate simulation of distributed applications and platforms”. In: *Journal of Parallel and Distributed Computing* 74.10 (2014), pp. 2899–2917 (cit. on p. 20).

[9] Georges Da Costa, Léo Grange, and Inès de Courchelle. “Modeling, classifying and generating large-scale Google-like workload”. In: *Sustainable Computing: Informatics and Systems* 19 (2018), pp. 305–314 (cit. on p. 51).

[10] Anja Feldmann, Oliver Gasser, Franziska Lichtblau, et al. “The Lockdown Effect: Implications of the COVID-19 Pandemic on the Internet Traffic”. In: *Broadband Coverage in Germany; 15th ITG-Symposium*. VDE. 2021, pp. 1–5 (cit. on p. 4).

[11] Jessie Frazelle. “Power to the People”. In: *ACM Queue* 18.2 (2020), pp. 5–18 (cit. on p. 12).

[12] Ronald Gelaro, Will McCarty, Max J. Suárez, et al. “The Modern-Era Retrospective Analysis for Research and Applications, Version 2 (MERRA-2)”. In: *Journal of Climate* 30.14 (2017), pp. 5419–5454 (cit. on p. 52).

[13] Greenpeace. *Clicking Clean*. Greenpeace International, 2017 (cit. on p. 4).

[14] Ori Hadary, Luke Marshall, Ishai Menache, et al. “Protean: VM Allocation Service at Scale”. In: *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*. 2020, pp. 845–861 (cit. on p. 22).

[15] Maroua Haddad, Jean-Marc Nicod, Marie-Cécile Péra, and Christophe Varnier. “Standalone renewable power system scheduling for a green data center using integer linear programming”. In: *Journal of Scheduling* 24 (2021), pp. 523–541 (cit. on p. 44).

[16] Marwa Haddad, Georges Da Costa, Jean-Marc Nicod, et al. “Combined IT and power supply infrastructure sizing for standalone green data centers”. In: *Sustainable Computing: Informatics and Systems* 30 (2021), p. 100505 (cit. on p. 5).

[17] Helmut Hlavacs, Georges Da Costa, and Jean-Marc Pierson. “Energy Consumption of Residential and Professional Switches”. In: *2009 International Conference on Computational Science and Engineering*. Vol. 1. 2009, pp. 240–246 (cit. on p. 10).

[18] Helmut Hlavacs, Georges Da Costa, and Jean-Marc Pierson. “Energy Consumption of Residential and Professional Switches”. In: *2009 International Conference on Computational Science and Engineering*. Vol. 1. 2009, pp. 240–246 (cit. on pp. 41, 50).

[19] Martijn Koot and Fons Wijnhoven. “Usage impact on data center electricity needs: A system dynamic forecasting model”. In: *Applied Energy* 291 (2021), p. 116798 (cit. on p. 3).

[20] Eric Masanet, Arman Shehabi, Nuoa Lei, Sarah Smith, and Jonathan Koomey. “Recalibrating global data center energy-use estimates”. In: *Science* 367.6481 (2020), pp. 984–986 (cit. on p. 3).

[21] Rajeev Muralidhar, Renata Borovica-Gajic, and Rajkumar Buyya. “Energy Efficient Computing Systems: Architectures, Abstractions and Modeling to Techniques and Standards”. In: *arXiv preprint arXiv:2007.09976* (2020) (cit. on p. 3).

[22] Haider Niaz, Mohammad H. Shams, Mohammadamin Zarei, and J. Jay Liu. “Leveraging renewable oversupply using a chance-constrained optimization approach for a sustainable datacenter and hydrogen refueling station: Case study of California”. In: *Journal of Power Sources* 540 (2022) (cit. on p. 5).

[23] Padma Priya, R., and D. Rekha. “Sustainability modelling and green energy optimisation in microgrid powered distributed FogMicroDataCenters in rural area”. In: *Wireless Networks* 27.8 (2021), pp. 5519–5532 (cit. on p. 4).

[24] Md Mustafizur Rahman, Abayomi Olufemi Oni, Eskinder Gemechu, and Amit Kumar. "Assessment of energy storage technologies: A review". In: *Energy Conversion and Management* 223 (2020), p. 113295 (cit. on p. 36).

[25] V. Dinesh Reddy, Brian Setz, G. Subrahmanya V. R. K. Rao, G. R. Gangadharan, and Marco Aiello. "Metrics for Sustainable Data Centers". In: *IEEE Transactions on Sustainable Computing* 2.3 (2017), pp. 290–303 (cit. on p. 57).

[26] Charles Reiss, John Wilkes, and Joseph L Hellerstein. "Google cluster-usage traces: format+ schema". In: *Google Inc., White Paper* (2011) (cit. on p. 22).

[27] Marc Richter, Pio Lombardi, Bartlomiej Arendarski, et al. "A vision for energy decarbonization: Planning sustainable tertiary sites as net-zero energy systems". In: *Energies* 14.17 (2021) (cit. on p. 5).

[28] John Roach. "Microsoft's virtual datacenter grounds "the cloud" in reality". In: *Microsoft Innovation Stories* (Apr. 20, 2021) (cit. on p. 50).

[29] Enida Sheme, Sébastien Lafond, Dorian Minarolli, Elinda Kajo Meçe, and Simon Holmbacka. "Battery Size Impact in Green Coverage of Datacenters Powered by Renewable Energy: A Latitude Comparison". In: *Advances in Internet, Data & Web Technologies*. Ed. by Leonard Barolli, Fatos Xhafa, Nadeem Javaid, Evjola Spaho, and Vladi Kolici. Cham: Springer International Publishing, 2018, pp. 548–559 (cit. on p. 6).

[30] Jie Song, Peimeng Zhu, Yanfeng Zhang, and Ge Yu. "Versatility or validity: A comprehensive review on simulation of Datacenters powered by Renewable Energy mix". In: *Future Generation Computer Systems* 136 (2022), pp. 326–341 (cit. on p. 6).

[31] Pedro Velho, Lucas Schnorr, Henri Casanova, and Arnaud Legrand. "On the Validity of Flow-level TCP Network Models for Grid and Cloud Simulations". In: *ACM Transactions on Modeling and Computer Simulation* 23.4 (Oct. 2013) (cit. on pp. 15, 21).

[32] Lukas Wagner, Simone Mastroianni, and Andreas Hinsch. "Reverse Manufacturing Enables Perovskite Photovoltaics to Reach the Carbon Footprint Limit of a Glass Substrate". In: *Joule* 4.4 (2020), pp. 882–901 (cit. on p. 36).

[33] Noelle Walsh. "How Microsoft measures datacenter water and energy use to improve Azure Cloud sustainability". In: *Microsoft Azure Blog* (Apr. 22, 2022) (cit. on p. 51).

[34] Di Wang, Chuangang Ren, Anand Sivasubramaniam, Bhuvan Urgaonkar, and Hosam Fathy. "Energy Storage in Datacenters: What, Where, and How Much?" In: *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*. SIGMETRICS '12. London, England, UK, 2012, pp. 187–198 (cit. on pp. 43, 44).

[35] Minxian Xu and Rajkumar Buyya. "Managing renewable energy and carbon footprint in multi-cloud computing environments". In: *Journal of Parallel and Distributed Computing* 135 (2020), pp. 191–202 (cit. on p. 24).

[36] Dajun Yue, Fengqi You, and Seth B. Darling. "Domestic and overseas manufacturing scenarios of silicon-based photovoltaics: Life cycle energy and environmental comparative analysis". In: *Solar Energy* 105 (2014), pp. 669–678 (cit. on p. 52).

Abstract

Falar do consumo de energia
impacto ambiental dos dcs
emissao de carbono
tecnicas carbon aware
carbon responsive
sizing e contention
follow the renewables

Résumé

Falar do consumo de energia
impacto ambiental dos dcs
emissao de carbono
tecnicas carbon aware
carbon responsive
sizing e contention
follow the renewables