Digital Sufficiency in Data Centers : Studying the Impact of User Behaviors

Maël Madon

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

Université de Toulouse

Doctorat de l'Université de Toulouse

préparé à l'Université Toulouse III - Paul Sabatier

Étude de comportements de sobriété dans les centres de calculs

---

Thèse présentée et soutenue, le 30 avril 2024 par

Maël MADON

École doctorale
EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse

Spécialité
Informatique et Télécommunications

Unité de recherche
IRIT : Institut de Recherche en Informatique de Toulouse

Thèse dirigée par
Georges DA COSTA et Jean-Marc PIERSON

Composition du jury
Mme Ivona BRANDIC, Rapporteure, Vienna University of Technology (TU Wien)
M. Arnaud LEGRAND, Rapporteur, Université Grenoble Alpes
Mme Patricia LAGO, Examinatrice, Vrije Universiteit Amsterdam
M. Emmanuel JEANNOT, Examinateur, INRIA Bordeaux
M. Laurent LEFEVRE, Examinateur, INRIA Lyon
M. Georges DA COSTA, Directeur de thèse, Université Toulouse III - Paul Sabatier
M. Jean-Marc PIERSON, Co-directeur de thèse, Université Toulouse III - Paul Sabatier

Acknowledgments

La thèse a été pour moi un parcours exigeant, passionnant et riche en enseignements. Si ces trois dernières années ont été heureuses et agréables, je sais que je le dois aux personnes qui m'entourent.

Pour commencer, merci infiniment à vous, Georges et Jean-Marc. Je ne pense pas arriver à exprimer suffisamment ma gratitude pour votre soutien sans faute. Geo, merci pour ton aide technique, pratique et morale. Tu m'as aidé à être efficace et aller vers l'essentiel sans me tracasser. Tu as été un allié de proximité dans cette thèse, tant sur le plan professionnel que personnel. Jean-Marc, merci pour ton intégrité, tes relectures critiques et ta bonne humeur sans faille. Grâce à toi, nos réunions étaient toujours plus agréables, et j'en sortais systématiquement rempli d'énergie, de motivation... et de chocolat. Sachant à quel point vous êtes occupés, le temps que vous m'avez accordé m'honore. Geo et Jean-Marc, vous êtes pour moi des modèles auxquels j'aspire, mais aussi des amis.

I am sincerly grateful to Ivona Brandic and Arnaud Legrand for having agreed to act as rapporteurs of this thesis, and for their indepth reading. I am also thankful to Patricia Lago, Emmanuel Jeannot and Laurent Lefevre for accepting the invitation to be members of the jury.

Merci ensuite à tous les collègues de l'équipe SEPIA, qui ont fait du quatrième étage mon milieu naturel. Merci en particulier à Igor pour m'avoir accompagné tout au long du parcours ; Florent pour ta présence rassurante et amusante dans le bureau au début de ma thèse ; Millian pour la cohésion que tu donnes à l'équipe et tes valeurs d'inclusivité, de redistribution et d'intégrité scientifique ; Chiara pour avoir réussi une intégration éclair parmi nous ; Amal pour ton expérience et ta gentillesse et Patricia pour manager tout ce beau monde d'une main de maître. J'ajoute aussi les acolytes Emmanuel, Olivier, Julie et Armelle, partenaires de l'UPSIDUM, de discussions ou jeux de sociétés.

Même si mon travail n'y était pas directement rattaché, j'ai vécu au rythme des réunions du projet ANR Datazero2. Merci de nouveau à Jean-Marc et Georges de m'y avoir systématiquement convié, elles ont ajouté à ma thèse une dimension collective et conviviale. En plus de ceux déjà mentionnés au-dessus, merci aux autres joyeux lurons de Datazero : Jean-Marc N, Manal, Laurent, Damien, Jérôme, Louis-Claude, Stéphane, Veronika et Christophe.

Au niveau du labo, je voudrais remercier Léonor et tous les personnels du premier étage pour leur aide inconditionnelle. Merci aux membres de la mission Transition Ecologique de l'IRIT avec qui nous avons combattu ensemble notre éco-anxiété. Laure, j'ai une grande admiration pour l'énergie que tu as donné à cette mission sur la durée. Merci à Véronique, David, Adrien, Antonio, Vincent, Hugo, Malo, Maelyss... Nous avons réalisé des actions dont nous pouvons être fiers !

A warm thank you to Patricia Lago who welcomed me in Amsterdam for almost three months in 2022. I learnt a lot from our collaborations, allowing me to extend my vision to other fields and methods of research. I am very greatful to the members of your research group for their sense of hospitality. I keep great memories of my visit in this beautiful city!

Je tiens à remercier aussi toutes les personnes qui ont croisé mon chemin pendant ces trois ans, lors de conférences ou écoles d'été, et avec qui j'ai pu partager de belles discussions. Il serait impossible de toutes les citer ici, mais en voici quelques-unes qui m'ont marqué : Thibault Pirson et toute l'équipe de la SICT pour leur dévotion à faire de cette école d'été un évènement de très grande qualité ; Lou Grimal, un modèle pour moi de doctorante surdouée, surbookée et sur-enthousiaste ; Philipp Wiesner, my German counterpart who I instantly matched with and who opened my eyes to how research is performed in other countries ; the ICT4S community to which I am very proud to belong, for their pionnier and visionary work on ICT and sustainability ; la communauté française Ecoinfo qui a aussi commencé à travailler sur ces sujets quand personne ne s'en préoccupait ; Françoise Berthoud, qui ne s'en rappelle peut-être pas, mais à qui je dois ma première mise en relation avec Jean-Marc Pierson ; Denis Trystram et ses photos de nombres ; Martin Quinson pour m'avoir dit "Résiste !" au bon moment lors de la rédaction... C'est grâce à ces liens professionnels et personnels que nous avançons ensemble sur nos questions de recherche complexes.

This PhD thesis was founded by the Ecole Polytechnique which gave me complete freedom in the choice of research directions, for which I am very grateful. It would not have been possible without previous and collective works that I want to mention here. Thanks to Dror Feitelson and collaborators for their inspiring research and for maintaining the Parallel Workload Archive from which we downloaded our input data. Thanks again to Millian Poquet, main developer and maintainer of Batsim, for his willingness to help with his simulator and Nix. Thanks to SimGrid, which embodies, in my opinion, what every research tool should be: open-source, tested, maintained and validated. Thanks also to the Grid'5000 testbed with which I carried out some of the experiments.

Pour finir, mes remerciements se tournent naturellement vers mes proches et ma famille. Amis musiciens, grimpeurs, d'enfance ou de promo, vous m'apportez tous les jours joie et épanouissement. Jean-Claude, alias "grand-père", il ne s'est pas passé un mois durant ma thèse et mes études supérieures sans que je pense à toi, et toujours avec autant d'émotion. Je sais que tu aurais été fier de moi et aurais suivi de près mon parcours de jeune chercheur. Une pensée aussi pour Thierry, le deuxième modèle de chercheur de mon enfance. A mes parents, Gérard et Marie-Pascale. Je réalise chaque jour un peu plus l'immense cadeau que vous nous avez fait à Julie à et moi en nous apportant votre amour et votre soutien sans aucune contrepartie ni attente. Cette rigueur, cette radicalité et cet amour des choses simples de la vie, je les dois à toi, papa. Cet épanouissement, ce soin aux autres et la plupart de mes passions, je les dois à toi, maman. Et à toi, Julie, grande sœur musicienne, écrivaine et sociologue. J'ai l'impression d'avoir à la fois suivi tes pas et marché à tes côtés. Merci pour ton affection et tes encouragements sans limites qui me touchent énormément.

Last but not least, this PhD thesis, Marie, it's also yours. Yours because you kindly proofread many of my drafts. But most importantly, yours because you followed this journey with me from the very first day. You bring me your unconditional love and support. You know when to challenge me and when to reassure me. You make me fly and keep my feet on the ground. You are the best life partner I could ever wish for.

Abstract

The Information Technologies (IT) industry has an increasing carbon footprint (2.1–3.9% of global greenhouse gas emissions in 2020), incompatible with the rapid decarbonization needed to mitigate climate change. Data centers hold a significant share due to their electricity consumption amounting to 1% of the global electricity consumption in 2018. To reduce this footprint, research has mainly focused on energy efficiency measures and use of renewable energy. While these works are needed, they also convey the risk of rebound effects, i.e., a growth in demand as a result of the efficiency gains. For this reason, it appears essential to accompany them with sufficiency measures, i.e., a conscious use of digital technologies with the aim to decrease the total energy and resource consumption.

In this thesis, we introduce a model for data centers and their users. In the first part, we focus on direct users, interacting with the infrastructure by submitting jobs. We define five sufficiency behaviors they can adopt to reduce their stress on the infrastructure, namely Delay, Reconfig, Space Degrad, Time Degrad and Renounce. We characterize these behaviors through simulation on real-world inputs. The results allow us to classify them according to their energy saving potential, impact on scheduling metrics and effort required from users. One drawback of sufficiency behaviors is their inertia, that we explain with appropriate metrics. We investigate thereafter the behaviors' usefulness in the context of renewable energy management. A three-state energy feedback mechanism informs the users on the status of electricity production. We show that adopting the sufficiency behaviors when renewable energy is scarce leads to brown energy savings. Savings are proportional to the efforts made by users.

In the second part, we build upon our user model and implementation to tackle an open issue in distributed system simulation. Most works use recorded traces to simulate workloads, by replaying jobs of the same characteristics and same submission time. However, this model is problematic when the performance of the simulated infrastructure differs from that of the original infrastructure. We model and implement "replay with feedback", a way of using recorded traces, preserving the think time between jobs rather than the original dates of submission. We provide an in-depth analysis of our method's impact with the help of novel metrics.

In the last part, we shift our focus to indirect users of data centers by studying professional cloud users. We design and conduct a qualitative study to investigate what a sufficient use of the cloud would mean, in practice. The study involves three focus groups analyzed through thematic analysis. The results provide a preliminary picture of the nature of our digital professional needs, along with a list of "tactics towards sufficiency", concrete actions to focus on the essential while limiting environmental footprint.

This manuscript offers an insight into digital sufficiency in data centers, involving both simulation and social sciences. We hope that our open-source code and reproducible simulation campaigns will be useful for future works in that direction.

Résumé en français

L'industrie des technologies de l'information a une empreinte carbone croissante (2,1 à 3,9 % des émissions mondiales de gaz à effet de serre en 2020), incompatible avec la décarbonation rapide nécessaire pour atténuer le changement climatique. Les centres de calculs y contribuent significativement en raison de leur consommation d'électricité : 1 % de la consommation mondiale en 2018. Pour réduire cette empreinte, les travaux de recherche se sont principalement concentrés sur des mesures d'efficacité énergétique et l'utilisation d'énergies renouvelables. Si ces travaux sont nécessaires, ils entraînent également des effets rebond, à savoir une augmentation de la demande en réponse aux gains d'efficacité. Pour cette raison, il apparaît essentiel de les accompagner de mesures de sobriété, c'est-à-dire d'une utilisation raisonnée des technologies du numérique, afin de diminuer la quantité globale d'énergie et de ressources consommée.

Dans cette thèse, nous présentons un modèle de centre de calculs et de ses utilisateurices. Dans la première partie, nous nous concentrons sur les utilisateurices directes, qui interagissent avec l'infrastructure en soumettant des tâches. Nous définissons cinq leviers de sobriété qu'iels peuvent adopter pour réduire leur impact sur l'infrastructure, à savoir Délai, Reconfiguration, Dégradation Spatiale, Dégradation Temporelle et Renoncement. Nous caractérisons ces leviers à l'aide de simulations sur des données réelles. Les résultats nous permettent de les classer en fonction de leur potentiel d'économie d'énergie, de leur impact sur les métriques d'ordonnancement et de l'effort requis de la part des utilisateurices. L'un des inconvénients des leviers de sobriété est leur inertie, que nous expliquons à l'aide de métriques ad hoc. Nous étudions ensuite le potentiel des leviers dans un contexte de gestion des énergies renouvelables. Nous montrons que l'adoption des leviers de sobriété en période de faible production renouvelable conduit à des économies d'énergie non renouvelable. Les économies sont proportionnelles aux efforts fournis par les utilisateurices.

Dans la deuxième partie, nous nous appuyons sur notre modèle d'utilisateurices et son implémentation pour aborder un problème ouvert dans la simulation de systèmes distribués. La plupart des travaux utilisent des traces réelles pour simuler les soumissions dans l'infrastructure, en rejouant des tâches ayant les mêmes caractéristiques et les mêmes temps de soumission. Toutefois, ce modèle pose problème lorsque les performances simulées diffèrent de celles de l'infrastructure d'origine. Nous modélisons et mettons en œuvre le « rejeu avec feedback », une façon d'utiliser les traces réelles, en préservant le temps de réflexion entre les tâches plutôt que les temps de soumission originaux. Nous fournissons une analyse approfondie de l'impact de notre méthode à l'aide de nouvelles métriques.

Dans la dernière partie, nous nous concentrons sur l'utilisation indirecte des centres de calculs, en étudiant des utilisateurices de cloud professionnel. Nous menons une étude qualitative afin d'examiner ce que signifierait, en pratique, un usage sobre du cloud. L'étude comprend trois focus groups analysés par le biais d'une analyse thématique. Les résultats dressent une image préliminaire de la nature de nos besoins professionnels numériques, ainsi qu'une liste de « tactiques vers la sobriété », c'est-à-dire d'actions concrètes pour se

concentrer sur l'essentiel tout en limitant son empreinte environnementale.

Ce manuscrit offre un aperçu de la sobriété numérique dans les centres de calculs, impliquant à la fois simulation et sciences sociales. Nous espérons que nos codes libres et nos campagnes de simulation reproductibles seront utiles pour de futurs travaux dans cette direction.

List of acronyms

CPU Central Processing Unit. 14, 25–27, 55

DVFS Dynamic Voltage and Frequency Scaling. 14, 15, 19, 20, 128

FCFS First Come First Served. 28, 32, 55, 84, 86, 87, 93, 99

FLOPS Floating-Point Operations Per Second. 26

GHG Greenhouse Gases. 9, 11–13

GPU Graphics Processing Unit. 26

HPC High Performance Computing. 13, 19, 20, 23, 25, 55, 75, 77, 95, 127

IPCC Intergovernmental Panel on Climate Change. 5, 12, 16, 18

IT Information Technologies. v, 5, 6, 8–19, 25, 105, 106, 108, 121, 124–126, 130

LCA Life Cycle Assessment. 10, 11, 129

PUE Power Usage Effectiveness. 14, 16, 19, 27

QoS Quality of Service. 19, 20

RJMS Resource and Job Management System. 31, 82

SLA Service Level Agreement. 19, 128

SWF Standard Workload Format. 26, 34

Contents

Introduction

Contents

1 Context

As of 2023, six of the nine "planetary boundaries" have been overshot because of human activities [1]. It is urgent to reduce quickly and significantly our environmental impacts. Unfortunately, this does not seem to be the path that the industry of Information Technologies (IT) is following. Estimations of its carbon footprint range from 1.8% to 3.9% of global greenhouse gas emissions in 2020, and this share is likely in augmentation [2]. About a third of this footprint is attributable to data centers, the server farms that constitute the backbone of the Internet infrastructure. Their environmental footprint arises mainly from their electricity consumption, estimated to 2.4% of the total electricity consumption in France [3] and 1% globally [4].

To reduce this impact, researchers and companies have proposed many solutions: improving the cooling through better server layout, reducing the operating frequencies of servers, migrating the workload to follow renewable electricity production, to name only a few. Most of the time, these techniques allow for gains in *efficiency*, i.e., they optimize systems to consume fewer resources for the same service provided to the user. They have the potential to decrease the direct environmental impact of data centers, but typically fail to address the indirect effects arising from their use [5]. In particular, efficiency leads to more performant and more affordable services, which in turn stimulate consumption and tend to counter-act the energy saving potentials. This effect is known in economics as the *rebound effect* and apply particularly to digital goods and services, which are immaterial with low barriers to adoption [6]. For this reason, an increasing number of international bodies (like the Intergovernmental Panel on Climate Change (IPCC) in their 2022 report [7]) have started to acknowledge that efficiency measures must also be accompanied by *sufficiency* measures, i.e., strategies aiming at decreasing the absolute level of resource and energy demand.

The term "digital sufficiency" is relatively new and was theorized by Santarius et al. [8]. One of its main dimensions is "user sufficiency" which involves voluntarily decreasing the demand for digital services. We argue that environmentally-aware people are ready to make efforts, like for instance the users of French telecom operator Telecoop restricting their use of mobile data [9].

In this PhD thesis, we propose to study "sufficiency behaviors" for data center users, and their potential to reduce the environmental footprint of these infrastructures. We use mainly simulations, but also social science methods to try to reach an understanding of the problem in its entirety.

2 Problem statement and research challenges

Given the context explained in the previous section, one main question is guiding our work:

> **Overarching question:** Which are the levers of user sufficiency in data centers, and how effective are they to reduce the environmental impact of these infrastructures?

We make the distinction between "direct" and "indirect" users of data centers. Direct users are in close interaction with the infrastructure, mostly by submitting jobs or reserving virtual machines. Indirect users use services hosted on data centers, but have no precise vision of the underlying infrastructure. We firstly focus on the formers, for which we need a model. Hence our first question:

> **Question 1:** How to accurately model the interaction between direct users and the data center?

Once we have a suitable model, we can come back to the main question and decline it to these users:

> **Question 2:** Which "sufficiency behaviors" can be adopted by direct data center users, and how does user effort translate into footprint reduction?

Secondly, we move on to the second category of users. They perform a great variety of digital activities, indirectly using data centers through different types of software stack. Consequently, their sufficiency levers are different and more varied than for direct users. We focus on the case of cloud usage for flexible work, and investigate the following question:

> **Question 3:** What are the opportunities for digital sufficiency in cloud usage?

3 Main contributions

The main contributions of this PhD are listed below along with the research question(s) to which they are related. The two first contributions are transversal and developed in several chapters, whereas the last three correspond directly to specific chapters.

A model for direct data center users and sufficiency behaviors (*Q1–2*) Data center simulations require a model of their IT platform, workload and resource manager. We extend existing models with a mathematical representation of direct data center users. They interact with the data center by submitting jobs. They can react to internal events such as the termination of previous jobs or external events such as availability of renewable energy. We introduce a novel model of eco-responsible user behaviors called "sufficiency behaviors". They are voluntary modifications of the characteristics of submitted jobs in order to decrease the demand in the data center. We define five such behaviors, namely

- postponing the job submission (Delay),
- reducing the number of requested parallel resources (Reconfig),
- degrading the job to request less resources (Space Degrad),
- degrading the job to lower its execution time (Time Degrad), and
- renouncing the job submission (Renounce).

Batmen: an open-source software to simulate users of large-scale distributed systems (*Q1-2*) The software developed in this thesis constitutes an important contribution. First and foremost is Batmen, a plugin enabling the simulation of users inside the scientific resource management simulator Batsim. The simulated users can submit tasks dynamically and react to events. Batmen is open source and available in a GitLab repository where it is automatically built and tested by a continuous integration pipeline.
Additionally, we endeavored to ensure the reproducibility of all experiments presented in this manuscript. Therefore, all software used are open source and their versions are carefully reported. We always provide the scripts needed to reproduce the experiments and perform the subsequent data analysis.

In-depth analysis of the sufficiency behaviors (*Q2*, Chap III–IV) The five sufficiency behaviors are characterized through simulation on real-world inputs. The results allow us to classify them according to their energy saving potential, impact on scheduling metrics and effort required from users. Results reveal the inertia of sufficiency behaviors, that we explain with appropriate metrics.
We investigate thereafter the behaviors' usefulness in the context of renewable energy management. A three-state energy feedback mechanism is introduced to inform the users on the status of electricity production. We show that adopting the sufficiency behaviors when renewable energy is scarce helps save non-locally produced electricity. Savings are proportional to the efforts made by users. The results allow us to discuss the relevance of our feedback mechanism and potential user incentives.

A model of workload replay accounting for user feedback (*Q1*, Chap V) Thanks to our user model and implementation we make a contribution to tackle an open issue in distributed system simulation. Most works use recorded traces to simulate workloads, by replaying jobs of the same characteristics and same submission time. However, this model is problematic when the performance of the simulated infrastructure differs from that of the original infrastructure. We provide a definition of "replay with feedback", a way of accounting for the impact of system performances on user submission behavior in simulations. We model and implement a novel model of replay with feedback and provide an in-depth analysis of its impact compared to the traditional replay model. For this analysis, we define three original metrics: *mean lateness*, *relative lateness* and *additional lateness*.

A qualitative study of digital sufficiency in cloud usage (*Q3*, Chap VI) In a last contribution, we shift our focus to indirect users of data centers by studying professional cloud users. We design and conduct a qualitative social science study to investigate what a sufficient use of the cloud would mean, in practice. The study involves three focus groups analyzed through thematic analysis. The results include (i) a list of main digital work activities and differences according to work settings, (ii) the perceptions of what makes certain digital activities or their cloud feature *necessary*, and (iii) a list of "tactics towards

sufficiency" i.e., concrete actions to focus on the essential while limiting environmental footprint.

4 Dissertation outline

The manuscript is organized as follows.

Chapter I presents the background of the thesis: environmental situation and share of the IT industry, classical footprint reduction techniques in data centers and the notion of "digital sufficiency". We continue with related works on user involvement for sustainability in data centers.

Chapter II introduces the model of a data center and its users. We also detail the simulation environment that we use and our software contribution Batmen.

Chapter III moves on to the definition of the sufficiency behaviors, and characterizes them individually with regard to energy consumption and scheduling metrics through a simulation campaign.

Chapter IV studies the sufficiency behaviors in the context of a data center powered by renewable energies. Another simulation campaign is presented and the resulting energy savings are compared to the effort required from users.

Chapter V presents our model of replay with feedback accounting for user reaction to system performance inside simulations based on real workload traces. We compare our method with the traditional replay model thanks to another set of experiments analyzed with three novel metrics.

Chapter VI looks at digital sufficiency in practice. We give the results of a qualitative study carried in the Netherlands among flexible workers using cloud solutions for their work. We investigate the opportunities for a more conscious use of these technologies.

The last chapter concludes our work and suggests short-term and long-term perspectives for future works on the topic.

The work presented in this manuscript has led to peer-reviewed publications [C1, C2, C3, J1, C4, C5], a detailed list of which is given at the end of the document.

Chapter I

Background and related works

Contents

This chapter gives perspectives on the topic of this thesis and discusses related literature. We start by providing context on the current environmental crisis, and the specific contribution of the IT industry, in Section 1. Section 2 reviews classical techniques for footprint reduction in data centers, and highlights their limitations. In Section 3, we introduce the concept of digital sufficiency that this PhD suggests applying to data centers. A necessary step towards sufficiency is to involve the users in reducing the footprint of data centers. We end the chapter with a review of literature on previous attempts to do so.

1 Environmental situation and IT industry

1.1 Planetary boundaries

In 2023 the "Earth Overshoot Day"[1] fell on August 2. It means that on that day, the ecological footprint of humanity was already equal to the amount of natural resources our planet is able to renew in one year. In other words, our way of life is unsustainable [11].

Overexploitation of land and agricultural activity has led to a decline of 60% of animal population between 1970 and 2014 [12]. It is the first time in history that a single species (humans) had such a dramatic impact on the other species.

In parallel, we are facing an unprecedented climate crisis due to the emission of more Greenhouse Gases (GHG) than natural sinks can absorb. For example, the concentration of carbon dioxide in the atmosphere has increased from its pre-industrial levels of around 278 parts per million (ppm) to 410 ppm on average today [13]. Carbon dioxide is the best-known of all GHG, which is why we commonly measure their emissions in tons of carbon dioxide equivalent, denoted tCO_2e. The GHG amplify the natural greenhouse effect, inducing an extra energy influx on the Earth's surface, a phenomenon called "radiative forcing". The extra energy is absorbed by oceans, lands and the atmosphere, in the form of heat. This in turn, has a number of ecological consequences: melting of glaciers and

[1] reported in https://overshoot.footprintnetwork.org/ based on data from the Footprint Data Foundation [10]

ice caps, sea level rising, biodiversity loss, extreme climate events etc. It is estimated that human activities are responsible for a global surface temperature increase of 1.07 °C in 2019 compared to pre-industrial levels [13].

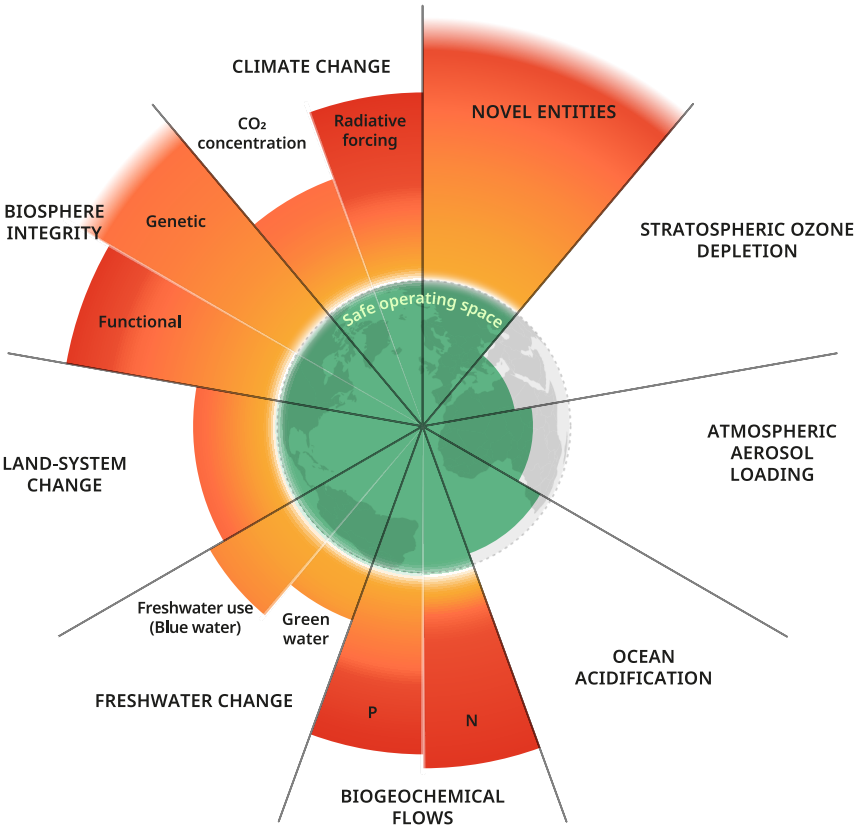


Figure I.1: The nine planetary boundaries in 2023.
Credit: Azote for Stockholm Resilience Centre, based on [1]. ©CC BY-NC-ND 3.0

Unfortunately, the list of environmental damage caused by human activity does not stop there. The issues previously mentioned (loss of biodiversity and climate change) constitute two of the nine "planetary boundaries" [14] illustrated in Figure I.1. With this framework, Rockström et al. intend to represent the geophysical limits of our planet. Under the green threshold, human activities remain within a "safe operating space", where the Earth system is able to safe-regulate. But beyond the thresholds, the system is not stable anymore and risks triggering non-linear environmental changes. Six of the nine boundaries are overshot according to the 2023 update [1]. All the challenge lies in bringing human activities back into the safe operating space while still guaranteeing social minimum and well-being for all in this planet. Or, to use the words of Kate Raworth, to "live within the donut" between social foundations and planetary boundaries[2] [15].

1.2 Direct environmental impacts of IT

For the IT industry, the environmental impacts come from the production, use and disposal of digital equipment. They can be estimated with Life Cycle Assessment (LCA) [16], a common impact assessment method. For example, Figure I.2 displays the results of the

[2]Kate Raworth is a British economist, known for her theory of "donut economics". The outer edge of the donut are the planetary boundaries, like in Figure I.1, while the inner edge represents the essential human needs.

LCA of a data center. We note first that environmental impacts cannot be summarized with only one value, but have to be expressed with several "impact indicators". This reflects the diversity of planetary boundaries introduced above. Also, we can see in this case that the use phase is dominating for all impact factors. In other words, most environmental impacts occur during use, through the electricity consumption of the infrastructure. The impacts related to the production and end-of-life are relatively small.

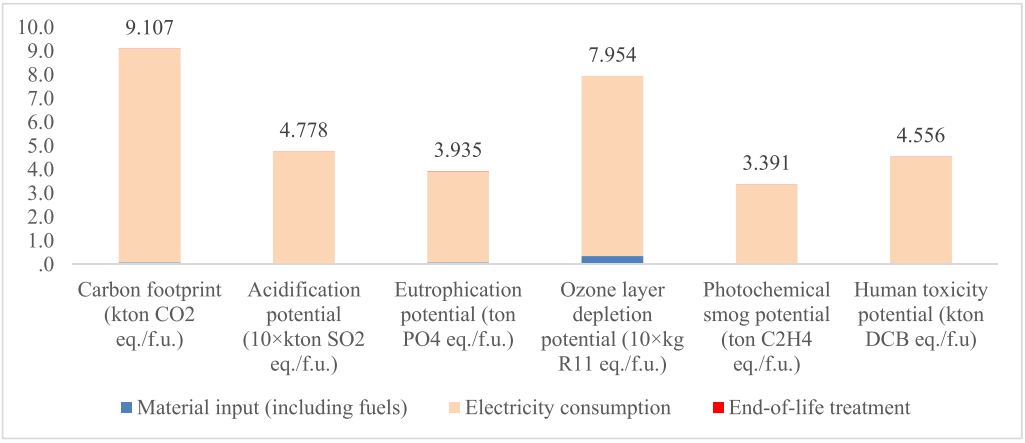


Figure I.2: Environmental impact analysis of a large-scale data center in Ankara, Turkey, based on data from 2020, split by life cycle phase [17].
Results of a Life Cycle Assessment (LCA) on 6 impact indicators performed with CCaLC2 LCA software and Ecoinvent 2 database. ©Reprinted from [17], with permission from Elsevier

Beware that these results are specific to data centers, which are large and intensively used infrastructures, and to the geographical location in Turkey, a country with a relatively high carbon intensity of electricity production. In France, the GHG emissions from data centers are rather divided into 25% for the use phase and 75% for the production phase on average [3, p.12]. The distribution of impacts per life cycle phase look also very different for small IT devices like smartphones or computers, where the manufacturing phase dominates.

A last takeaway from this example is that conducting a LCA is a long and complicated process, especially for objects as complex as IT devices. Performing such a precise environmental assessment for each and every data center or device released to the market appears out of reach.

Carbon footprint of the IT industry All the same, experts have tried to estimate the impacts of the industry as a whole. Full and detailed assessments on the topic are quite rare, and only focus on electricity consumption or GHG emissions. There are three main peer-reviewed sources: Andrea and Edler (2015) [18], Malmodin and Lundén (2023) [19] and Belkhir and Elmeligi (2018) [20]. All three studies are based on LCA of IT equipment, sorted in three tiers: data centers, networks and end-user devices. They use secondary data to estimate the quantities of each. Their results are given in Figure I.3.

According to the estimates, the IT industry is responsible for around 0.6 to 1.3 $GtCO_2e$ globally. However, these figures have been criticized as they are probably missing parts of the supply chain due to truncation errors. A recent estimate based on Input-Output analysis finds 1.8–2.1 Gt [21], while a meta-analysis gives 1.2–2.2 Gt for the carbon footprint of IT [2], including TVs and other consumer electronics. With the last range, **the IT industry would represent 2.1–3.9% of global GHG emissions** [2]. Moreover, the authors of all studies previously cited agree that these emissions will at best stabilize, but likely increase in the coming decade, due to growth in usage, emergence of new technologies

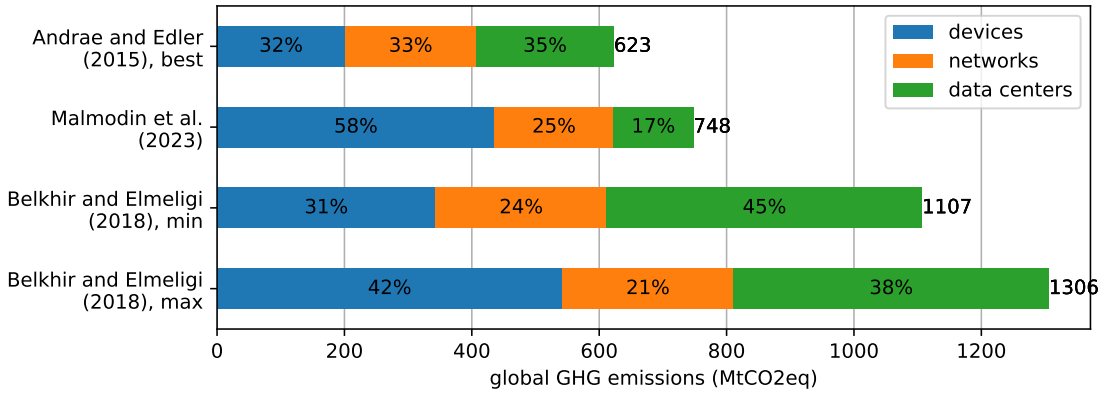


Figure I.3: Worldwide GHG emission estimates for IT industry in 2020, in $MtCO_2e$.
Figures from Malmodin et al. original paper [19], and Freitag et al. supplementary material [2] for Andrae and Edler [18] and Belkhir and Elmeligi [20].

and slowdown in efficiency gains, unless strong political actions are taken [2]. This trend seems incompatible with the necessity to drastically reduce global GHG emissions. A reduction of 44% by 2030 and 84% by 2050 would be needed to limit global warming to 1.5 °C (pathway C1 IPCC 2022 [7, p. 22]).

What we also learn from Figure I.3 is the share of emissions attributable to each tier. This distribution vary between sources, but we can remember that they are of the same order of magnitude. The emissions from devices are evenly shared between the production phase and the use phase, whereas **emissions from networks and data centers are dominated by the use phase**, i.e., their energy consumption [19], as we saw with the example of the Turkish data center (Figure I.2). In fact, global data center electricity consumption is estimated to 205 TWh in 2018 [4], or 1% of global electricity consumption.

1.3 Indirect environmental impacts of IT

In the previous part, we drew a picture of IT *direct negative impacts* on the environment. Yet, the application of digital technologies also have indirect implications, that can be both positive or negative. We refer to Hilty and Aebischer for a classification of all effects by three levels [5], illustrated in Figure I.4.

Indirect effects On the one hand, the use of digital technologies has the potential to optimize processes, thus lowering their footprint, substitute polluting technologies (the so-called "dematerialization"), or enable sustainable practices. They are the second- and third-order positive environmental effects in Figure I.4. For example, optimization algorithms have many applications to real-world problems, where they may allow for better resource management and lower environmental footprint, from smart grids [22] to vehicle routing [23]. As an example of positive systemic effect, digital technologies have helped to develop soft mobility solutions, such as public transports or car sharing [24].

Applying IT might also on the other hand induce the consumption of other items that were not in use before ("induction effects"). Additionally, they could render previous and functioning technologies prematurely obsolete ("obsolescence effects").

Rebound effects A particularly preoccupying issue is the rebound effect, that tends to counteract all positive consequences of the application of IT. The rebound effect — also

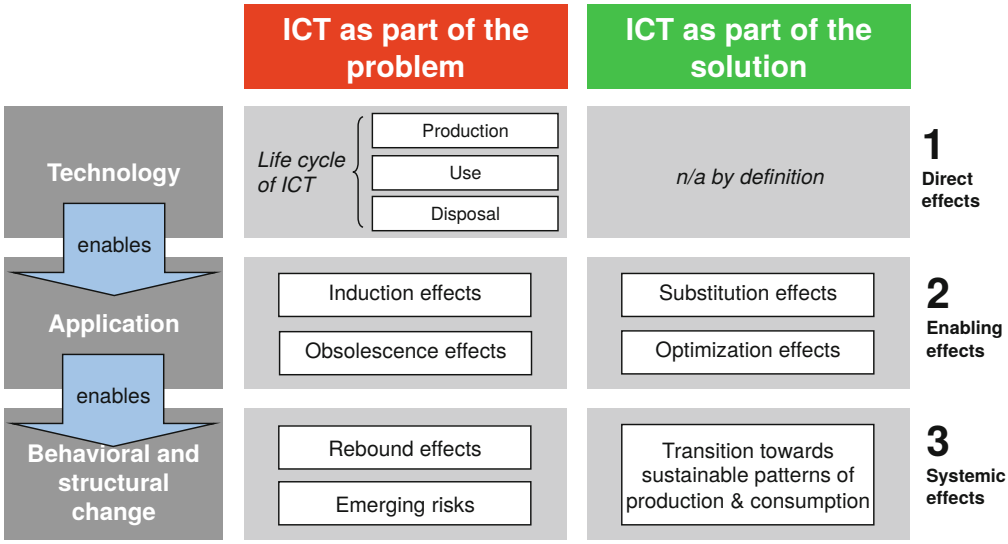


Figure I.4: Three-level model of environmental effects of IT.
©Reprinted from [5], with permission from Springer

known as Jevons' paradox after its discoverer [25] — is when energy efficiency measures do not reach their full energy saving potential [26]. For example, an analysis of Chinese city-level macroeconomic data reveals that the actual carbon reduction in these cities under the digital economy was only 40% of that expected, due to rebound [27]. IT are very prone to rebound effects because their immateriality, potential for virtualization and low entry barriers for adoption tend to transform efficiency into growth in usage [6]. The diversity and multifactorial nature of rebound mechanisms make them difficult to predict precisely. Fortunately, we are witnessing increasing efforts in the community to take them into account [28, 29], in order to assess the real enabling potential of IT for energy savings.

2 Sustainability levers in data centers

As we saw in Part 1.2, data centers have a considerable impact on the environment, mainly arising from their use phase. Malmodin et al. estimate the GHG emissions due to the production of data centers at 30 $MtCO_2e$ in 2020, and 95 Mt to their use phase, or 24% and 76%, respectively [19, Table 8]. Additionally, these infrastructures require a large amount of water for their cooling systems [30]. Consequently, data center sustainability occupies an increasing importance in the public discourse. For instance in Ireland, tensions are rising since they are projected to consume 25% of the country's energy by 2030 [31]. The same goes for the Netherlands, a relatively small country where many of them are hosted [32]. This increasing attention is also reflected in the academic literature, where we find a large body of works on strategies to reduce the environmental footprint of data centers.

In this section, we do not intend to give an exhaustive review of these works, but rather an overview of common techniques for footprint reduction. We rely on recent and comprehensive literature surveys on the topic in both cloud [33, 34] and High Performance Computing (HPC) [35] worlds, and only cite one or two articles per technique as examples. We divide the techniques in three main categories: energy efficiency (2.1), thermal and cooling management (2.2), and use of renewable energies (2.3).

2.1 Energy efficiency

Data centers, as large and complex distributed infrastructures, offer countless avenues for energy efficiency optimizations, at all levels. Below, we provide a short description of classical levers.

DVFS A commonly used technique for energy reduction is Dynamic Voltage and Frequency Scaling (DVFS). It entails lowering the voltage and frequency of computing systems in order to save energy or avoid overheating. Indeed, parts of the power consumption of the Central Processing Unit (CPU) is proportional to $V^2 f$ where V is the voltage and f the frequency. Lowering the frequency enables to lower the voltage needed for stable operation. The use of DVFS is frequently studied in data centers, either alone or in combination with other techniques. For example Kim et al. use it when scheduling applications with deadlines, and reach up to 45% energy gains in case of low utilization [36]. Guerout et al. study the efficiency of the DVFS governor *OnDemand* available in the Linux kernel and find 15% energy savings compared to running at maximum frequency [37].

Server shutdown When a server is idle, i.e., not used to compute any user task, it still consumes a non-negligible amount of power. To save this power, the server can be switched off. Since on/off cycles also takes time and energy, research is required to develop good heuristics. Orgerie et al. studied these cycles in detail and proposed shutdown strategies in periods of low utilization [38].

Virtualization and consolidation In cloud data centers, the computing resources available to users are virtual machines or containers, which are seemingly self-standing machines whose operating system and software are fully controlled by the user. Virtualization provides an abstraction layer on top of physical machines, allowing for optimizations like packing two or more underutilized virtual machines on the same physical server, to share computing resources [39], or shut down unused servers [40, 41].

Heterogeneous platform Sometimes, servers inside the data center do not have the same hardware characteristics and power consumption. Energy optimizations can be reached by leveraging this heterogeneity, whether it comes from CPUs [42], memory [43] or storage [44]. For example, the scheduler can prioritize energy-efficient servers, or use the best performing servers to finish the computation as fast as possible and be able to switch them off.

Note that the levers listed above are in no particular order and can reach comparable energy savings depending on the type of workload and infrastructure studied.

2.2 Thermal and cooling management

Given the high density of computing nodes in server rooms, a large amount of energy is needed to cool down the infrastructure. According to the Uptime Institute, the average Power Usage Effectiveness (PUE) of data centers in 2022 was 1.55 [45]. The PUE is a standard proxy for energy efficiency measuring the ratio between the amount of energy consumed by the whole infrastructure (including cooling, power distribution and ancillary facility functions), and the energy used by IT. A PUE of 1.55 indicates then that 65% electricity powers the IT equipment, the remaining 35% being lost in cooling etc. As a result, another part of the literature looks at improving thermal management.

Cooling management In this domain, the largest potential for efficiency gains probably come from outside the Computer Science field. Excess heat from the servers is typically removed with a closed-loop air flow going through an air conditioner. We see works in improving the rack layout for optimized air flow, reusing waste hot air to heat up other buildings, using outside air instead of air conditioning (free cooling) or directly cooling down the chips with liquid cooling [46].

Thermal-aware scheduling A complementary approach is to develop thermal-aware scheduling algorithms. For example Sun et al. describe and test an online scheduling heuristic that assigns jobs with respect to spatio-temporal temperature constraints [47]. The data center layout and cooling system are taken into account for the decision. In a previous work, we also studied the opportunity to pre-cool the data center in anticipation of periods of low renewable production [48].

2.3 Use of renewable energies

Finally, the last category of techniques for footprint reduction presented here concerns the use of decarbonized sources of electricity. Ensuring 100% renewable energy supply in data centers has become more and more common to fulfill the increasing number of commitments for net-zero emissions by the big tech companies [49]. Whether these commitments are achieved through power-purchase agreements or on-site renewables, they open many research challenges.

Workload adaptation to power envelope The main challenge of renewable energies is their intermittency. The power they produce varies during the day and between seasons. For example Pierson et al. study a data center self-supplied in renewable energies (solar and wind) in the project DataZero [50]. For its functioning, the power module has to negotiate with the IT module that adapts the computing load to the available power, using levers such as workload temporal shifting or DVFS [51]. The elasticity offered by virtual machines can also be used for such problems [52].

Geographic load shifting Another approach to tackle variability of production is to "follow the sun": migrating the workload to parts of the world where production is abundant. We find many such proposals in the literature, leveraging various techniques such as virtual machines migrations [53] or scheduling with deadlines [54] to solve the problem.

Use of electricity storage To enable better integration of renewable energies, some works study the use of electricity storage systems. We refer to a survey of solutions from Rostirolla et al. [55]. These systems can be short-term, e.g., batteries storing the excess photovoltaic production during the day to use it at night, or long-term, e.g., hydrogen storage systems able to store the energy for months to face the seasonal variability of production. From there, challenges include sizing the power infrastructure and IT platform according to the workload [56], or minimizing battery aging [57], to only name a few.

2.4 Limitations of these techniques

As we have seen in this section, research is abundant on techniques for footprint reduction in data centers. The picture we painted is incomplete, and we refer the reader to the surveys cited for a more comprehensive overview.

All the same, the techniques mentioned have one thing in common: their objective. They seek to either maximize performance under energy or CO_2 constraints, or minimize energy/CO_2 under performance constraints[3]. Therefore, these techniques aim to address *direct environmental effects* from the use of data centers, according to Hilty's taxonomy in Figure I.4. Doing so, we argue that they fail to acknowledge higher-order effects, and in particular rebound effects arising from efficiency.

In fact, efficiency indicators have improved in the last two decades: the average PUE decreased from 2.50 in 2007 to 1.55 in 2022 [45], server utilization is assumed to have risen (meaning that the infrastructures are better utilized) [58], and big companies are increasingly claiming net-zero emissions [49]. But in parallel, the number of Internet users rose from 3 billion in 2015 to 5.3 billion worldwide in 2022, and data center energy use from 200 to 240–340 TWh in the same period [59], despite the energy savings.

In other words, efficiency improvements were successful to decrease the *relative* environmental footprint, but did not succeed in reducing its *absolute* level. That is why we decide to rather focus on *sufficiency*, that we introduce in the next section.

3 Digital Sufficiency

Sufficiency is a relatively new concept. We dedicate this section to defining and explaining it, first in the general context (3.1), then zooming in the IT industry (3.2). We also review related research on digital sufficiency (3.3).

3.1 On the concept of sufficiency

Definition For the first time in 2022, the IPCC (working group III) includes "sufficiency policies" as an approach to mitigate climate change. They provide a definition:

> "Sufficiency policies are a set of measures and daily practices that avoid demand for energy, materials, land and water while delivering human well-being for all within planetary boundaries." [7, p. 35].

Compared to efficiency, the goal shifts from reducing the amount of resources needed *per unit of good or service* to reducing the *absolute amount of these resources*. While efficiency measures may lead to rebound effects by making the good or service more accessible, sufficiency measures tackle them, in principle, by setting limits to final consumption. The first part of the definition highlights the nature of sufficiency policies. They are not technical, like efficiency, but rather societal, cultural or political. They put the humans at the center of the equation, and invite reflecting on their needs for goods and services. The last part of the definition echoes Rockström's planetary bounderies and Raworth's "social foundations", introduced in Part 1.1. We note, however, that this approach involves asking complex ethical questions that need to be approached and answered with care to reduce the risk for social, financial or geographic inequality.

In the scientific literature Sufficiency has been attracting increasing attention in the last decade, both in the scientific literature and the public discourse. Jungell-Michelsson and Heikkurinen report a growing number of academic publications on the topic since the year 2000, and across a wide range of academic fields [60]. Among the earliest mentions

[3] The term "performance" is to be understood in the general sense, e.g., throughput, response time and quality of service, depending on the use case.

of the term is the work of Princen, advocating for a long-term consideration of sustainability and addressing over-consumption [61]. What seems to be the common point in the sufficiency literature is the idea to focus on what is *enough*, either through individual responsibility, political decisions or paradigm shift in business logic [60].

In the public discourse Sufficiency has also become very present in the public discourse. In the case of France, we find an early mention to it in 2012, under the term "sobriété" [sobriety], by the association Negawatt [62]. They make it one of three pillars of their scenario for energy transition intended to the general public. More recently, the French Agency for Ecological Transition made sufficiency the corner stone of their first scenario to carbon neutrality "Génération Frugale" [Frugal Generation] [63]. The French government also issued calls for energy sufficiency, at the end of 2022, although in this case the main trigger was the gas crisis caused by the war in Ukraine [64].

Related concepts Sufficiency overlaps with other concepts, such as post-growth or degrowth [65, 66], conviviality [67], frugality, voluntary simplicity or low-tech [68].

3.2 Definition of Digital Sufficiency

Coming back to the IT sector, the new technologies appear to have played a central role in speeding up our consumption of natural resources. All sectors of the economy are undergoing their "digitalization", from entertainment, education, research to transport and agriculture. The use of digital content, by being seemingly immaterial, tends to appear as environmentally harmless. However, as we previously saw, digital technologies do have a significant direct and indirect footprint. It is therefore not surprising that people have started to raise the question of sufficiency in IT.

The French community has been very active on the topic, again, under the term "sobriété numérique". The first mention is attributed to Frederic Bordage from the GreenIT association[4], in 2008. He developed it since then in many books and articles, as a reflection on the role of IT in our lives and a set of concrete practices [69]. Philosopher Fabrice Flipo gave it a more historical, social and political meaning [70]. According to him, public and private authorities have pushed for the digitization of our lifestyles in order to pursue economic growth. He warns of the limited impact of "sufficiency of small gestures" (e.g., turning off the light) in the current system. The think tank The Shift Project made "sobriété numérique" [digital sobriety] the title of their 2018 report, giving it big media coverage [71]. Since 2020, we witness many mentions to this expression in the French academic literature [72, 73], or reports from public institutions [74, 75].

In the international community, Hilty mentioned already in 2008 the need for sufficiency in IT, at the same time that he lays the foundation of his three-level model (Figure I.4) [76]. He explains it by making the distinction with efficiency. We have to wait until 2018, with the original (German) version of the book "Smart Green World?" [77], that Lange and Santarius coin and develop the term "digital sufficiency". They introduce it as one of three guiding principles for a sustainable digitalization, and refer to it as "as much digitalization as necessary, and as little as possible". They expand on it later in an article dedicated to theorizing the concept [8] which we explain below.

[4] GreenIT is a collective of experts on digital sufficiency, responsible IT, web eco-design and low-techs https://www.greenit.fr/

Definition by Santarius et al. Digital sufficiency is, according to them,

> "any strategy aimed at directly or indirectly decreasing the absolute level of resource and energy demand from the production or application of IT" [8].

They do not claim to provide a definition of our "basic digital needs", but rather conceive a conceptual framework consisting of four dimensions:

- **User sufficiency:** users apply digital devices frugally and make use of IT in a way that fosters sufficiency-oriented lifestyles.
- **Hardware sufficiency:** producing and designing hardware for longevity, repairability, and with the least possible resource and energy demand.
- **Software sufficiency:** software development and implementation that ensures long-term functionality and the lowest possible data traffic and hardware utilization for task performance.
- **Economic sufficiency:** IT-borne improvements are used to nurture public and common good instead of economic growth.

These dimensions are intertwined and allow tackling both direct and indirect effects. They help to identify challenges that prevent resource or energy saving, for which the authors suggest counter-measures and policies. Digital sufficiency measures are broken down by stakeholder and sufficiency dimension. A few examples are: blocking advertisement by default (service provider), producing long-lasting and reparable hardware (manufacturer), developing open-source software (software developer), returning the device to collection points (private user) or regulating data collection (policy-maker) [8, Table 1].

3.3 Digital Sufficiency research

The two previous parts gave some background on sufficiency and digital sufficiency. The works cited are theoretical, laying the foundations of the concepts. In this part, we review studies adopting a more applied angle, investigating sufficiency in practice.

Sufficiency is linked with identifying basic human needs, or "human well-being for all within planetary boundaries" according to the IPCC definition. For a review of previous propositions and methods to define basic needs, we refer to review by Millward-Hopkins et al. [78]. They base their own work on the Decent Living Standards (DLS) [79], which are an inventory of universal material requirements to fulfill basic human needs. They derive quantitative thresholds for each of them, in terms of square meters of floor space per person, kilometers of mobility, etc. Then, they convert these values into primary energy requirements (Joules), which allow them to make comparisons with the world's energy production, current distribution of resources, and discuss scenarios for the future. Gorge et al. investigate our interpretation of needs in society, through a series of interviews of people with diverse background engaged in sufficiency [80].

Directly related to digital sufficiency is the work of Widdicks et al. They explore which Internet use is deemed as *necessary* and *unnecessary* in their own personal daily lives, and discuss the adaptations they developed to disconnect or reduce their use [81]. In a more recent work, they investigate how to design for "more meaningful and moderate online experiences" [82], by organizing design workshops with external participants and carefully analyzing the output material (transcripts, prototypes, post-its) through thematic coding. They ultimately develop concrete design recommendations, like "Internet speed bumps" to

set limits to digital consumption or "stripping back layers of service" to only retain the most meaningful content.

Elgaaied-Gambier et al. study the willingness of Internet users to adopt proenvironmental behaviors [83]. They start by collecting a list of eco-friendly online behaviors from diverse sources, and carry out a first qualitative study to collect general perceptions on IT footprint. Thereafter, they conduct two quantitative studies to understand the consumers' self-attribution of responsibility in the reduction of their digital footprint.

Finally, we find a large body of work on "eco-feedback" or interventions for behavior change, especially from within the field of Human-Computer Interaction [84, 85, 86]. For example, Noureddine et al. study the impact of displaying the energy consumption of software to users [87]. They find that eco-feedback helps to raise awareness and willingness to react with eco-responsible behaviors. However, users lack knowledge on the most efficient behavior to adopt. We found no similar study on HPC users or indirect data center users.

4 Sufficiency in data centers

So far, we have shown that IT has a negative environmental footprint (Section 1), that many levers exist to reduce it at the scale of data centers (Section 2) but that they are levers of efficiency rather than sufficiency (Section 3). What would *sufficiency levers* be, in data centers?

Apart from this thesis, we are not aware of any work studying this question in these terms. However, sufficiency is, as we saw, about rightsizing our consumption to what is deemed enough. It requires taking into account the users of data centers, and involving them in the quest for sustainability. We found some works on this topic, that we summarize below.

4.1 Involving the users

Green contracts Garg et al. study a green cloud architecture, including several cloud providers periodically publishing "green offers" with price, time, CO_2 rating etc., and users issuing cloud requests specifying a desired Quality of Service (QoS) [88]. A middleware matches the requests and offers, aiming to minimize CO_2 footprint without impacting QoS. Some authors suggest the use of "green Service Level Agreement (SLA)" [89, 90, 91, 92, 93, 94] to formalize the guarantees requested by users, and the QoS degradation they are willing to accept. For example, Amokrane et al. leverage spatial and temporal variability of renewables and electricity prices to maximize profit of cloud providers with users specifying a carbon emission limit [92]. They make a linear formalization of the problem and test heuristics with thorough simulations. Haque et al. use similar methods to study a data center with local renewable production and whose users can specify a minimum percentage of green energy to run their jobs [91]. The provider earns a premium for meeting green SLA or is penalized if violating.

Rewarding users Instead or in addition to green SLA, we find propositions of incentives for users to accept QoS degradation. Many works look at monetary incentives through energy-aware pricing schemes. Users are charged based on past energy consumption [95], price of electricity [96] or a mix between local PUE, electricity price and renewable production [97]. The different approaches are validated through simulations or experiments on testbeds. Borghesi et al. exploit the energy savings obtained with DVFS to propose four dynamic pricing schemes [98]. They show that it is possible to save operational costs

that way, without penalizing users.
Other authors propose non-monetary rewards, such as certified "greenness" [99]. More interestingly, Georgiou et al. reward the users by giving priority to jobs based on their owners' past energy usage [100]. Their scheduling policy is implemented in the popular HPC resource manager SLURM. Experiments show that more energy-efficient jobs yield better scheduling metrics.
Unfortunately, in all the studies, the actual effectiveness of the reward mechanism to change the users' behavior is never studied. They would require implementation in production systems or rolling out quantitative studies.

In all the works cited above, users are seen either as requesting a certain level of environmental performance, or being rewarded for accepting QoS degradation. The environmental performances are achieved through the common techniques presented in Section 2: DVFS, geo-distributed data centers, etc. In the following, we focus on works considering the users as an active lever to lower the infrastructure's footprint.

4.2 Users as flexibility

Orgerie et al. look at energy savings achievable through accepting temporal delay in the start times of jobs [101]. They carry a simulation campaign with real-world traces, varying the proportion of users accepting the degradation. Their results show that 3% energy can be saved if all users accept delay, and that this delay averages to 15 hours. Cappiello et al. combines delay acceptation with variable carbon intensity of electricity [102]. Users submit job requests, specifying their availability to postpone deployment with acceptable delay. The cloud provider sorts possible cloud sites per carbon intensity and answers with several choices (immediate or delayed).

Guyon et al. look at the impact of spatial reconfiguration of jobs on energy consumption [103]. They simulate a scientific cloud with a bin-packing scheduler. Users can decide to submit their jobs on a smaller number of computing nodes ('big', 'medium' or 'little' version). Having all users submitting their jobs in 'medium' allow for 20% energy savings compared to 'big', thanks to better bin-packing and server switch off.

In another article, they combine the two levers previously mentioned by proposing their users to accept both delay and spatial reconfiguration in their jobs [104]. Experimental validation is performed with an ah'hoc simulator and a relatively small workload. They reach improvements in energy consumption of 2% if all users accept delay, and 5% if they also accept reconfiguration.

Basmadjian et al. make a comprehensive proposal in the All4Green project [105]. This project studies a collaboration between the energy supplier, the data center and customers, with the objective to better match the supply and demand of electricity. Several mechanisms are leveraged: "internal flexibilities", namely task migration, use of batteries, and adjustment in the cooling temperature, and "external flexibilities", namely delay and performance degradation for users. Their approach allows for 38% energy savings with internal flexibilities only, and a further 5.5% with external flexibilities.

In this thesis, we define and study such user flexibilites, that we call "sufficiency behaviors". Three of them encompass the levers mentioned above: Delay (temporal shifting), Reconfig (downsizing of requested resources) and Degrad (performance degradation). We introduce an additional user flexibility, absent from the literature: renouncing the job submission. This behavior is included in a deliberate approach of digital sufficiency [8]. We give an overview of the comparison between our work and the related works in Table I.1.

	RE?*	Delay	Reconfig	Degrad	**Renounce**
Orgerie 2008 [101]		✓			
Cappiello 2014 [102]	✓	✓			
Guyon 2019 [103]			✓		
Guyon 2018 [104]		✓	✓		
All4Green 2018 [105]	✓	✓		✓	
This thesis	✓	✓	✓	✓	✓

*is the work in the context of Renewable Energy integration?

Table I.1: Summary of related works and their links to the sufficiency behaviors defined in this thesis (Chapter III)

To the best of our knowledge, our work is the first to combine all four user levers together. These levers are the center of the study, compared to some others where they play only a secondary role. We provide a clear definition and characterization of them, and study their potential in the context of renewable energy integration. Finally, unlike most studies, we use a state-of-the-art simulator and provide all the software and material to reproduce the experiments as open-source repositories.

Chapter II

Model and simulation environment

Contents

In this chapter, we describe our model for a data center and its environment. We also present the simulation tools used to implement this model and run the experiments presented in Chapters III, IV and V.

1 Data center model

Our data center model is illustrated in Figure II.1. It includes:

- **the users**: people using the data center, by submitting *jobs* to it;
- **the jobs**: computational tasks to be carried out;
- **the IT platform**: group of servers using electricity to execute the *jobs*;
- **the scheduler**: program receiving the *job* requests from the *users* and assigning them to specific servers in the *IT platform* to execute them.

In the following sections, we give a detailed description of each component.

1.1 Users

1.1.1 Definition

Direct and indirect users Data center users, in the most general sense of the term, include all Internet users, since the data centers are the backbone of the Internet. In fact, people make use of data centers at very different levels, from using a mail client to launching HPC simulations. In this thesis, we make the distinction between *direct* and *indirect* users, which has some similarities with the distinction between Infrastructure-, Platform and Software-as-a-Serice provisioning models commonly made in the cloud paradigm. Here, the difference is made on the level of closeness — and therefore, of control — that users have with the computing platform.

We assume that everything running on a data center is ultimately submitted by a direct user, and serves the indirect users through a more or less long chain of intermediaries. For example, an employee using a cloud-supported text editor must have a subscription

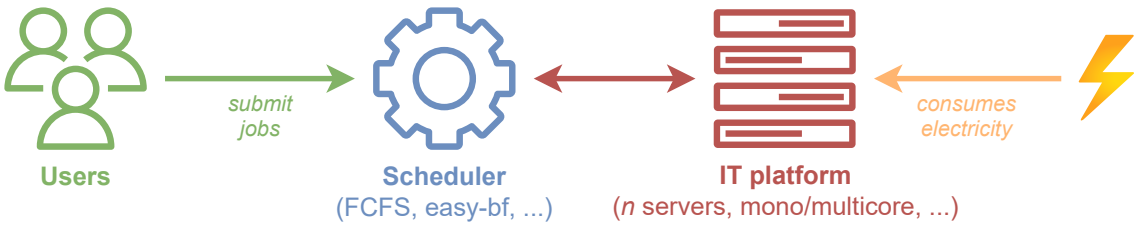


Figure II.1: Overview of the data center model

provided by her company (Software-as-a-Service). This company has a contract with a cloud provider, who allocated a certain number of servers to meet the company's demand. In this case, the direct user is the cloud provider, closely interacting with the hardware infrastructure.

In this thesis, we mainly focus on direct users (Chapters III, IV and V). We provide a model for them, and include them in simulations. The indirect users are not taken into account in the model, but rather addressed in Chapter VI through qualitative research.

Model Giving a model of direct users means modelling the way they submit jobs to the platform. Hence the generic definition:

Definition 1. *A (direct)* ***user*** *is defined by a unique identifier and a submission behavior. A* ***submission behavior*** *is an algorithm that decides which job is submitted and when, depending on the context.*

It is inside the submission behavior that lies all the model's intelligence. There are two main types (1.1.2), *generative* or *replay*, mirroring the two most common ways of modelling the workload in data center simulations. Submission behaviors are further characterized by the type of events they react to (1.1.3).

1.1.2 Types of submission behavior

Generative submission behavior The algorithm of a generative submission behavior creates the jobs to submit by following a mathematical model. *Example: user Anna submits one job every hour.* In order to reflect observations, the submission pattern follows most of the time a probability law. *Example: user Patrick submits a job of size s every t seconds. s follows an exponential law and t follows a Poisson distribution.*

Replay submission behavior In a replay submission behavior, the algorithm uses a historical record of user submission from a real infrastructure to reproduce the patterns of submission. We give an example of such a historical record in the next section. **In this thesis, all the users are modelled with this type of submission behavior.**

In both cases, algorithms defining the submission behavior can react to context.

1.1.3 Reaction to context

In real infrastructures, users adapt their behavior to their environment. Similarly, the submission algorithm can also react to context or events such as:

- the date (time of the day, day of the week, ...);

- external events, e.g., price or carbon intensity of electricity, peek use (release of a new product for a company, conference deadline for a researcher. . .);
- internal events, e.g., server utilization or termination of a previous job.

The aim of this PhD is to study how effective users can be to react to external events linked to electricity production.

1.2 Jobs and workload

1.2.1 Jobs

What we call *job* is a self-standing computing task submitted by a user. In theory, it represents any type of computation that someone can be interested to run in a data center, e.g., a script generating pay stubs, the training of a machine learning model or a virtual machine hosting a web server. In general, a job needs to perform computations on the CPU, memory read/write, I/O with disks and network etc. However, the specific representation we chose for a job is very simple and accounts only for the CPU. This is closer to the model of a batch task that behaves independently of the context, as we could encounter in HPC.

Definition 2. *A **job** is represented by a computing load, without communication. It is characterized by*

- *a quantity of floating-point operations to execute, and*
- *a number $r \in \mathbb{N}^{+*}$ of requested parallel resources.*

It is assumed that the r computing resources share the computing load evenly.

Remark In case of a homogeneous IT platform, where all servers have the same performance (which is the assumption we make in the following), it is equivalent to define a job with an **execution time** instead of a quantity of operations. The execution time is denoted d (for 'duration').

In addition to the above characteristics, a number of dates are associated to a job along its lifecycle:

- a **submission time** a (for 'arrival'): time at which the user submits the job to the scheduler;
- a **start time** $s \geq a$: time at which the job starts its execution in the IT platform (if applicable);
- a **finish time** $f = s + d$: time at which the job finishes its execution.

For some scheduler, other information provided by the user can be needed for a job, like the **walltime** w (upper bound on execution time given by the user) or the **due date** (deadline on the date of termination).

1.2.2 Workload

A data center has to manage hundreds of jobs that are submitted every hour by users. They constitute what is called the *workload*:

Definition 3. *A **workload** is a list $(j_1, ..., j_n)$ of jobs, each with an associated submission time.*

Workloads vary greatly with the type and size of data center and their specific users. To make reliable simulations, it is important to model them accurately. As we said before, they can be generated from a mathematical model or taken from a historical record from a real infrastructure. In the latter case, we talk about *recorded jobs* and *recorded (workload) traces*:

Definition 4. *A **recorded job** j_i is a set of information collected on a real job, that characterizes its execution in the original infrastructure. This set of information contains at least d_i, r_i, a_i and f_i, but also the walltime w_i if applicable, the job ID, user ID etc.*

Definition 5. *A **recorded trace** is a list $(j_1, ..., j_n)$ of recorded jobs, ordered by submission time.*

Recorded traces are precious sources of information for the replay submission behavior. For example, the Parallel Workloads Archive[1] gathers to date a collection of 40 workload logs from production systems around the world. These workload logs are given in the Standard Workload Format (SWF)[2], which is a space-separated data format, with one recorded job per line, including all the job information described above, and more. A field 'user ID' is also included, allowing to split the workload by user of the data center.

For the simulations presented in this thesis, we used exclusively the workload logs from the Parallel Workloads Archive as inputs.

1.3 IT platform

The IT platform represents the actual computing hardware infrastructure. In reality, a data center contains hundreds of servers, located inside racks and interconnected with network cables. Each server is a self-standing computer, containing one or several CPU, its own memory and network card, and possibly other peripherals (e.g., disk, Graphics Processing Unit (GPU)) depending on the use case. All the racks are arranged in a room, with extra facilities like cooling system, power distribution units, emergency power generators etc.

In our model, we take a simple definition for a server:

Definition 6. *A **server** is a CPU that can run the jobs. It is defined by a name, a number of cores and a performance (in Floating-Point Operations Per Second (FLOPS)).*

*A server can be in two **main states**: switched on or switched off, and two **intermediary states**: switching on or switching off. It can only compute a job if it is in the state "switched on". Transitioning from state "switched on" to "switched off" takes a constant amount of time T_{soff} and passes through the state "switching off". The opposite transition takes a time T_{son} and passes through the state "switching on".*

Note that, according to the definition above, servers can be monocore or multicore. This will have implications for the scheduler. Our model for IT platform follows:

Definition 7. *The **IT platform** is a set of servers.*

[1] https://www.cs.huji.ac.il/labs/parallel/workload/

[2] https://www.cs.huji.ac.il/labs/parallel/workload/swf.html

Model assumptions Additionally, our model makes a series of simplifying assumptions:

- homogeneous platform: all servers are identical (same performance, same number of core, same power consumption);
- perfect communication: communications are instantaneous, without latency;
- no failure: the servers never break down and always behave as expected;
- extra facilities are not modelled and their power consumption is not taken into account (this is equivalent to consider a PUE[3] = 1).

1.4 Energy model

Let $P_{DC}(t)$ be the power consumption of the data center at time t. In our model, the only components that consume energy are the servers. As a result, with $P^{(i)}$ the power consumption of server i and N the number of servers in the platform, we have:

$$P_{DC}(t) = \sum_{i=1}^{N} P^{(i)}(t) \tag{II.1}$$

The power consumption of server i is defined per state. In the state "switched on", this power consumption depends on the server utilization $u^{(i)}(t)$, i.e., the percentage of maximum CPU cycles used, following an affine relationship commonly used in the literature [106]: $P^{(i)}(t) = P_{idle} + u^{(i)}(t) * P_{dyn}$, with P_{idle} and P_{dyn} two constants. In any other state, the power consumption is considered constant.

Since the jobs in our model are compute-only jobs, with exclusive access to the cores (see Section 1.2), they use 100% of their allocated cores. The server utilization is then proportional to the number $k^{(i)}(t)$ of cores used. The power drawn by each core is considered constant and denoted P_{core}. In the end, we have:

$$P^{(i)}(t) = \begin{cases} P_{off} & \text{if the server is switched off} \\ P_{son} & \text{if the server is switching on} \\ P_{soff} & \text{if the server is switching off} \\ P_{idle} + k^{(i)}(t) * P_{core} & \text{otherwise} \end{cases} \tag{II.2}$$

1.5 Scheduler

The scheduler makes the link between the users and the IT platform. It receives job submissions from users and decides on when to execute them and which servers to allocate to them.

Definition 8. *A **scheduler** is an algorithm. It maintains a queue of submitted jobs and knows at any time the state of the IT platform (i.e., which servers are used, idle, switched off etc.). It is in charge of deciding which job will be executed when, and by which server(s). It has the possibility to switch servers on or off.*

Scheduling decisions must follow a set of rules, that constitute what we call the "scheduling hypotheses". Our model follows these scheduling hypotheses:

[3] The Power Usage Effectiveness (PUE) is a common metric for data centers operators. It corresponds to the ratio of the total energy used by the facility to the energy delivered to the computing equipment.

1.5.1 Scheduling hypotheses

1. **requested cores:** a job with r requested parallel resources must be executed on exactly r cores in the IT platform;
2. **server state:** only the cores of "switched on" servers can execute jobs;
3. **exclusive access:** a core executes maximum one job at a time;
4. **no preemption:** once started, a job cannot be interrupted or reallocated to other servers;
5. **job kill:** a job can be killed, for example because it reached its walltime or due date. In this case, the cores it occupies are immediately released, and the job is considered as failed.

As a result, the lifecycle of a job in our model is first to be submitted, then executing, and finally either terminated or killed. A job can only be executed on idle cores, i.e., cores that are switched on but not executing any job.

In the remaining of this section we present the extra scheduling hypotheses corresponding to a monocore and multicore context. We also give examples of common scheduling algorithms for both contexts.

1.5.2 Monocore schedulers

In a monocore context, all the servers in the IT platform have only one core. In addition to the 5 scheduling hypotheses described above, we add the hypothesis below:

6. **cross-server execution:** a job can be executed on multiple servers

Consequently, testing if a job can be executed now is equivalent to compare its requested resources with the number of idle servers. If jobs in the queue are requesting more resources than those available, the scheduler uses some job property (submission time, due date, slowdown...) to decide on which job will be given priority.

We give as an example the pseudocode of two monocore scheduling algorithms used in this thesis. First Come First Served (FCFS) (Algorithm 1) schedules the jobs in strict submission time order. EASY-backfilling (Algorithm 2) uses the execution time estimates ("walltimes") given by users to fill idle servers with small jobs.

1.5.3 Multicore schedulers

In a multicore context, the servers have several cores. Since we assume the IT platform to be homogeneous (see Section 1.3), they all have the same number of cores K. In contrast to monocore context, we make the following scheduling hypothesis:

6. **single-server execution:** jobs must be executed on a single server

Consequently, schedulers do not accept jobs with a number of requested resources r greater than K. Testing if a job can be executed is equivalent to check if there is a server in the platform with at least r idle cores.

In this thesis, when we are in a multicore context, we use a bin-packing algorithm. The pseudocode of this algorithm is given in Algorithm 3. This algorithm has been made energy-aware by greedily switching off servers when they are idle, and switching them on again when needed.

Algorithm 1 FCFS (First Come First Served) scheduling algorithm

```
▷ called every time an event happens (job submitted/terminated/killed) ◁
function FCFS()
    N ← number of idle servers
    jobQ ← queue of waiting jobs, in submission time order
    repeat
        j ← jobQ.first()
        r ← number of requested resources of j
        if r ≤ N then
            execute j on r idle servers
            N ← N − r
            jobQ.pop(j)
    until (jobQ is empty) or (r > N)
```

Algorithm 2 EASY-backfilling scheduling algorithm

```
▷ called every time an event happens (job submitted/terminated/killed) ◁
function EASY()
    FCFS()                                          ▷ execute jobs in FCFS order
    N ← number of idle servers
    jobQ ← queue of waiting jobs, in submission time order
    if jobQ is not empty then                       ▷ try to backfill small jobs
        T ← expected start time of jobQ.first()
        for all job j in the rest of jobQ do
            r ← requested number of resources of j
            w ← walltime of j
            if (r ≤ N) and (now + w ≤ T) then       ▷ j won't disturb the first job
                execute j on r idle servers
                N ← N − r
                jobQ.pop(j)
```

Algorithm 3 Bin-packing scheduling algorithm, with server switch off

```
▷ called every time an event happens (job submitted/terminated/killed) ◁
▷ notation: "job size" = number of requested resources
            "server size" = number of idle cores ◁
function BinPacking()
    jobQ ← queue of waiting jobs, in decreasing size order
    for all job j in jobQ do
        r ← number of requested resources of j
        serverQ ← queue of switched-on servers, in increasing size order
        for all server s in serverQ do
            k ← number of idle cores of s
            if r ≤ k then
                k ← k − r
                execute j on s
                break for
        if no switched-on server found for j then
            serverQ ← queue of switching-on servers, in increasing number of unreserved
            cores
            for all server s in serverQ do
                k ← number of unreserved cores of s
                if r ≤ k then
                    k ← k − r
                    execute j on s as soon as s is switched on
                    break for
            if no switching-on server found for j then
                if the list of switched-off server is not empty then
                    take a switched-off server s and ask it to switch on (taking T_son seconds)
                    reserve r resources on s
                    execute j on s as soon as it is on
        if a server was found for j then
            jobQ.pop(j)
    for all server s idle do                    ▷ switch off all servers left idle
        ask s to switch off (taking T_soff seconds)
```

2 Simulation tools

In this section, we give an overview of the software used in this thesis to implement the model described in the previous section and run the experiments. The specific experimental details will be given for each experimental campaign in the corresponding chapters.

2.1 Data center simulator: Batsim and SimGrid

SimGrid The core of the simulation is managed by SimGrid[4] [107], a distributed system simulator widely used and trusted in the community. SimGrid includes a fine-grain simulation of the IT platform, whose elements (servers with their computing speed, number of core, interconnection etc.) are described in a XML file. SimGrid enables the simulation of the platform's power consumption through its energy plugin[5], following the energy model described in Section 1.4.

We chose this simulator because it is state-of-the-art, scientifically validated and benefits from more than 20 years of development. The software is open-source and well-maintained, using modern methods of software development and automatic testing. Finally, the community is active and easily reachable.

Batsim SimGrid simulates the IT platform: the bare-metal computations and interactions between machines. It does not include a simulation of the Resource and Job Management System (RJMS). For this, we use **Batsim**[6] [108], a software layer on top of SimGrid. Batsim simulates the RJMS including the job manager (job arrivals, queue of waiting jobs, ...) and resource manager (list of available servers, possibility to switch on/off, ...), using discrete event simulation.

The scheduler has to be written in a separate program, the "batsched" ("Batsim scheduler"). It interacts with Batsim through a set of events defined by the Batsim Protocol[7] and exchanged via a ZeroMQ socket.

A Batsim simulation takes as input a workload (in JSON) and a platform (in XML), and is launched in parallel with the scheduler. It reads the workload to reproduce the job arrivals, simulates the IT platform thanks to SimGrid, sends events to the scheduler (e.g., job arrival, job termination) and receives orders from the scheduler (e.g., execute job, reject job, switch off server).

Among the few other available simulators using SimGrid, Batsim was chosen because it corresponds the most to our needs. Like SimGrid, it is robust, open-source, tested and still maintained. Moreover, Batsim is built to make reproducible experiments, and eases the interactions with the declarative package manager Nix, described after (see Section 2.3).

2.2 Batmen: our Batsim plugin to simulate users

Batsim and SimGrid allow us to simulate three of the four components of our data center model (Section 1): the jobs, the IT platform and the scheduler. However, when this thesis started, there were no tool available in the SimGrid ecosystem to simulate the fourth component: users of the data center. During this PhD, I developed Batmen, a plugin for Batsim enabling the simulation of users.

[4] https://simgrid.org/, v3.31 to 3.34
[5] https://simgrid.org/doc/latest/Plugins.html?highlight=energy#host-energy
[6] https://batsim.org/, v4.1 and 4.2
[7] https://batsim.readthedocs.io/en/latest/protocol.html

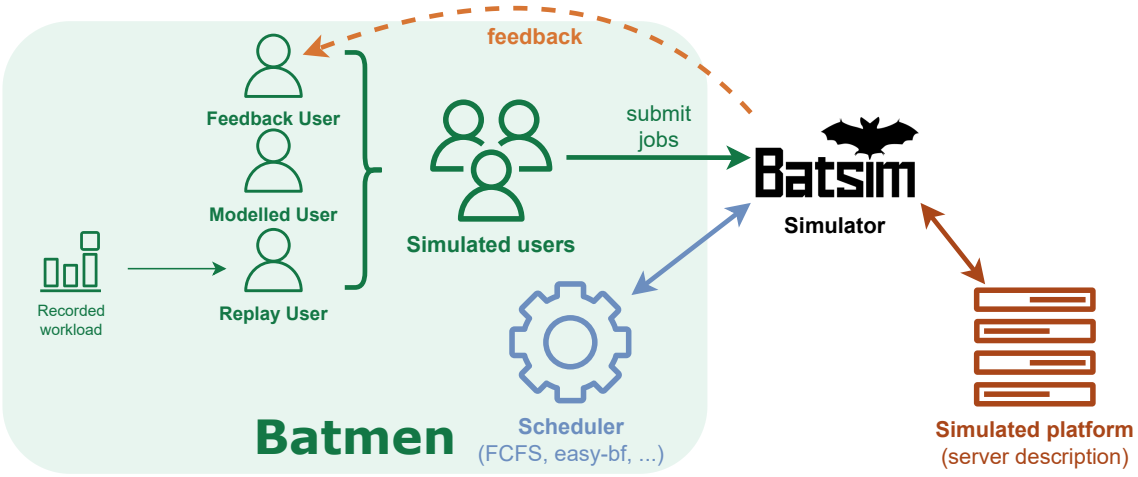


Figure II.2: Batmen diagram

2.2.1 General description

Batmen is a software written in C++, whose development started in July 2021. It is originally a fork of batsched[8]. Batmen is open source under license LGPLv3 and available in our GitLab repository: `https://gitlab.irit.fr/sepia-pub/mael/batmen`. It contains more than 10k lines of source code and 668 commits (in v3.1). The code is automatically built and tested by a CI (Continuous Integration) pipeline. All the source code is located in the folder `src/`, and compiled with Meson/Ninja. The folder `test/` contains a set of integration tests managed with Pytest.

In terms of global organization, Batmen takes the role of this second program which dialogues with Batsim. It therefore includes the scheduler, as well as the users who submit jobs dynamically during the simulation. The program architecture follows this organization, and is divided in three main parts:

- the core files defining the overall structure and interaction with Batsim, located directly under the `src/` folder. These files are mostly kept from the original project batsched.

- `src/scheds`: the implementation of different schedulers for Batsim. This is where the schedulers FCFS, EASY-backfilling and bin-packing previously described (Section 1.5) are located. Schedulers and their parameters can be selected from Batmen Command Line Interface (CLI).

- `src/users`: the utility files enabling the interaction with users, and the implementation of specific user submission behaviors (see Section 1.1). We describe it in more details in the next section. The set of users to be used in a simulation can be described in a JSON file and passed to Batmen CLI.

The architecture of Batmen is illustrated in Figure II.2.

[8] set of C++ schedulers for Batsim: `https://framagit.org/batsim/batsched`

2.2.2 Interaction with users

The interaction with users happens through what we call the "broker". In Batsim, jobs are commonly read from a predefined input workload[9] in JSON. In contrast, the broker manages a pool of simulated users and dynamically submits their new jobs during the course of the simulation. Technically, these jobs come on top of the input workload read by Batsim, although we leave this static workload empty most of the time.

Implementation, in the folder `src/users`:

- class `DynScheduler`: superclass managing the interaction with the broker from the scheduler. A scheduler inheriting from this class will handle dynamic jobs submitted by the broker, if any. To do so, it asks the broker for the next submission date and uses Batsim `REQUESTED_CALL` to call it back when it has something to submit. Then, it uses Batsim "dynamic job registration" feature to inject the new submissions inside the simulation. A `DynScheduler` also forwards to the broker feedback on job status.
- class `Broker`: implements the broker, submitting the jobs on behalf of the users. Maintains a `user_queue`: a list of users ordered by next submission time. The broker is regularly called by the scheduler through three public methods:
  - `jobs_to_submit`: returns the list all jobs to submit right now (submitted by any of the users),
  - `next_submission`: returns the date of the next submission by any of the users,
  - `feedback_job_status`: acknowledges the latest execution-related activity and forward the information to the relevant users
- class `User`: an individual user, defined by her submission behavior. It contains methods corresponding to those of the broker (`jobs_to_submit`, `next_submission` and `wake_on_feedback`), but on an individual level.

2.2.3 Types of user

So far, 11 categories of users are available for use in Batmen. The list of available categories and their input parameters is in the file `src/users/user_description_file_template.json`. Their implementations are in the files `user_*.cpp`.

The currently available categories can be grouped into three types (see Figure II.2):

- **modeled users** (superclass `User`): using a mathematical model to generate the jobs to submit (corresponds to a *generative* submission behavior).
- **replay users** (superclass `ReplayUser`): replaying a workload trace given as input (corresponds to a *replay* submission behavior). Alterations from the original trace are allowed, like the "sufficiency behaviors" in Chapters III and IV.
- **feedback users** (superclass `FeedbackUser`): taking feedback on the status of previous jobs into account when submitting the next one. Used in combination with `ReplayUser` in Chapter V.

Implementing your own user type inside one of these categories should be relatively low-effort (around 100 lines of code).

[9] input workload in Batsim documentation: https://batsim.readthedocs.io/en/latest/input-workload.html

2.3 Reproducible experimental environment

Along this thesis, we endeavored to make our experiments reproducible, as recommended by many in the community [109]. We understand the term "reproducibility" in the sense of the ACM terminology [110]:

> The measurement can be obtained with stated precision by a different team using the same measurement procedure, the same measuring system, under the same operating conditions, in the same or a different location on multiple trials. For computational experiments, this means that an independent group can obtain the same result using the author's own artifacts.

This means that our data, software and methods are precisely documented and open source. In this section we give an overview of our experimental environment and what helped us to achieve reproducibility.

2.3.1 Input data

As stated before, all the experiments are realized with input data from the Parallel Workloads Archive, which is a collection of workload traces from different parallel infrastructure around the world. The data is publically accessible and widely used in the community for more than 20 years. The workload traces are available in the Standard Workload Format (SWF), making it easy to use and compare different traces between them. Information are also provided on the architecture of the infrastructure in which the workload was recorded.

2.3.2 Tools for parsing

I created tools for parsing, filtering and transforming the SWF into a format matching our needs. Both of them are open source under license GNU GPLv3:

- **batmen-tools**[10] is a Python script allowing to parse the SWF, apply filters, split by user and transform into a JSON format readable by Batmen.
- **swf2userSession**[11] is a Python script to read a workload trace in the SWF, decompose it into user sessions, analyze the dependencies between sessions and store the results in the Session Annotated Batsim JSON format (SABjson). Is is used to perform "replay with feedback" in Chapter V.

2.3.3 Software version management

Software are constantly evolving with the new features, bug fixes and code refactoring. In order to make sure that an external person would obtain the same results when performing the same experimental campaign as ours, the exact version of the software used should be specified.

This is made possible with the package manager **Nix**[12]. Nix ensures reproducibility through offering declarative experimental environments, i.e., shell sessions in which every software dependency is declared with its specific version (release tag or Git commit number). These versions are specified with a Nix-specific syntax in a text file.

[10] source code and documentation available at https://gitlab.irit.fr/sepia-pub/mael/batmen-tools

[11] source code and documentation available at https://gitlab.irit.fr/sepia-pub/mael/swf2userSessions

[12] https://nixos.org/

We used Nix to manage the dependencies of our software and pin their specific versions in our experiments.

2.3.4 Notebooks

Finally, to ease understanding and reproduction of our experimental campaigns, we detail their steps inside **notebooks**, which are files that include both text and code. A notebook is organized in "cells" that can be run independently. In our case, we use Jupyter Notebooks[13], with a mix of Markdown for the textual parts and bash/Python for the code. It provides a way for the readers to both see the experimental details, and re-run the simulations on their own machine.

We use Python and the library pandas[14] for data analysis, which we also record in notebooks. As a result, each figure and table presented in the result and discussion sections of this manuscript correspond to one cell in a notebook. Note that we also used notebooks during the research phase and tracked their versions with Git. It is therefore possible to retrace our scientific research approach by looking at the commit histories.

The notebooks and associated Nix file constitute a self-contained *artifact* to reproduce our experiments and data analysis. We include a reproducibility paragraph in each chapter, with a link the repository containing the artifact, when applicable.

[13] https://jupyter.org/
[14] https://pandas.pydata.org/

Chapter III

Definition and characterization of sufficiency behaviors

Contents

Digital technologies are increasingly contributing to global warming, among other environmental impacts. As we already explained, there is a fundamental problem with *efficiency* measures: they are likely to be outbalanced by a *rebound effect* in demand. Indeed, emphasis is often put to make energy optimization as effortless as possible to end-users. On the contrary, we argue that users of digital technologies must be brought back into the loop, made aware of their impact and empowered to mitigate it.

In this chapter, we consider direct users of a data center as they are defined in the previous chapter (i.e., submitting jobs to the infrastructure). We define five so-called "sufficiency behaviors" for them. Users could choose to lower their demand by delaying, reconfiguring, temporally degrading, spatially degrading or even renouncing their job submissions. These behaviors have the potential to reduce the load in the data center, hence its energy consumption. We provide an experimental characterization of each of these levers through simulation, trying to answer the following question:

> What is the effect of each sufficiency behavior to reduce energy consumption, and how do they compare?

The chapter is organized as follows: Section 1 describes our approach by defining the sufficiency behaviors and complementing the model presented in Chapter II. Section 2 presents the experimental setup for characterizing these behaviors. The results are provided in Section 3 and discussed in Section 4. We present in Section 5 the limitations of the study. The last section concludes the chapter. This work has been published in 2022 at the conference Euro-Par [C1].

1 Description of the approach

1.1 Context: demand response

For the sake of characterizing the behaviors, we place ourselves in a context of demand response. Demand response entails reducing the electricity *consumption* in response to low availability of electricity *production*. For example, some electricity markets have Coincident Peak Pricing programs, in which industrial consumers are charged a high price when electricity demand in the grid is the highest. These peak pricing events last typically 15 minutes [111] or one hour [112] but are only known *afterwards*, e.g., at the end of the month. The electricity supplier may send warnings to the consumer that a peak load event will happen in the next few hours.

Data centers are good candidates to participate in demand response programs, since they are large consumers of electricity and have a more flexible load than other industrial facilities [113].

In our model, a demand response event will be represented by a time window of few hours (called "demand response window" or "DR window" in short) during which the objective is to reduce electricity consumption as much as possible. This event is supposed unknown in advance. We want to investigate how the sufficiency behaviors can help us achieve this objective.

1.2 Sufficiency behaviors

1.2.1 Definitions

As an echo to the digital sufficiency dimension "user sufficiency" defined by Santarius et al. [8] (see I.3.2), we propose a definition of "sufficiency behavior" in our context:

Definition 9. *We say that a user submitting jobs to a data center adopts a* ***sufficiency behavior*** *if she changes the characteristics of the job she planned to submit with the objective to decrease the demand in the data center.*

Naturally, this definition is quite abstract as it requires knowing the original job the user "planned to submit" before determining weather the submission behavior was "sufficient" or not. Luckily, this is easy in our model: the nominal submission behavior is determined by the submission behavior algorithm (see Definition 1). Anytime but in the DR window, the users follow their algorithm. During the DR window, we make the assumption that they will modify their jobs according to five types of behaviors illustrated in Figure III.1 and described below. The decision is independent for each job.

We give in the following a short definition of each sufficiency behavior, along with one example and the way they are modelled.

Delay The user decides not to submit the job now, but later.
For example, a user could adopt this behavior if the job has a low priority: it needs to be completed at some point but not as soon as possible.
➢ the submission time of the job is postponed to the end of the DR window.

Reconfig The user reconfigures the job to request less resources in exchange for a longer execution time.
For example, a user launching an image processing job in parallel on 16 cores could launch it on 8 cores instead.

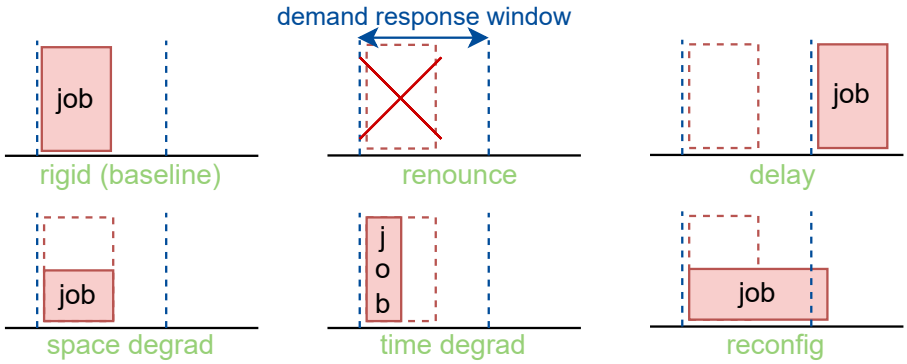


Figure III.1: **The five sufficiency behaviors and the baseline**

➢ the number of requested cores of the job is divided by two, rounded up, and the execution time is increased accordingly.
We make the hypothesis of perfect speedup, i.e., if the job takes 10 s to execute on three cores, it will take $(10 * 3)/2 = 15$ s on two cores.

Space Degrad The user degrades the job to request less resources.
For example, if the job was a video recommendation algorithm, it could recommend five videos instead of ten.
➢ the number of requested cores of the job is divided by two, rounded up. The execution time remains the same.

Time Degrad The user degrades the job to lower its execution time.
For example, if the job was a linear solver, we can lower its execution time by sacrificing on the accuracy of the approximate solution.
➢ the execution time of the job is divided by two. The number of requested resources remains the same.

Renounce The user decides not to submit the job, and never submit it in the future.
For example, the job could be an optional test that is judged not relevant enough with regard to the energy consumption required to run it.
➢ the job is not submitted and deleted from the workload.

We use one more type of behavior, which is the control behavior, or baseline:

Rigid The user submits the job normally, without adapting her submission behavior.
It could be because the job is critical, or simply because the user does not wish to make an effort.
➢ the job is submitted with all its original characteristics.

1.2.2 Important remarks

From the previous definitions, it is worth noting two things:

1. Behaviors Space Degrad and Reconfig are only available for multicore jobs. If the original job requests only one resource, the two behaviors have no effect.

2. Among the five behaviors presented above, only Reconfig and Delay preserve the mass of the jobs, i.e., their initial quantity of foating-point operations to execute. Time Degrad divides this quantity by two. Space Degrad divides this quantity by up to two (depending on the rounding). Renounce reduces this quantity to zero.

1.3 Data center model

We use the data center model described in Chapter II. The users are "replay users", who adapt their submissions from the input workload during the demand response window according to the chosen behavior. The platform is multicore, so the scheduling hypotheses of a multicore context are followed. The scheduler is a bin-packing scheduler with server shutdown, as in similar works [104, 103]. The pseudocode for this scheduler is given in Algorithm 3 (Chapter II, Section 1.5).

2 Experimental setup

This section describes the software, workload, platform and experimental choices used for the experiments (see Figure III.2).

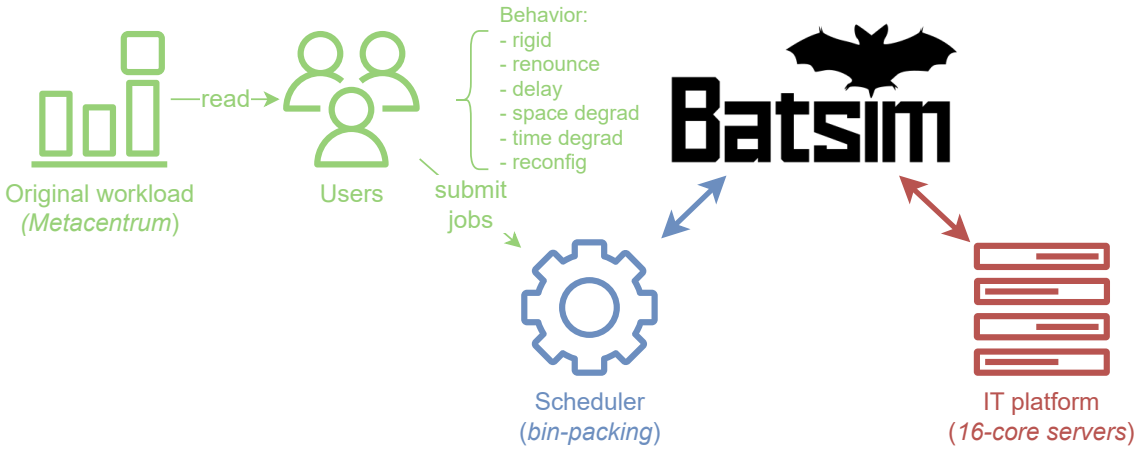


Figure III.2: **The simulated system**

2.1 Software used for simulation

To simulate our system, we use the simulation environment described in Chapter II (Section 2): Batsim v4.2 and SimGrid v3.34 for the data center simulation and Batmen v3.1 for the scheduler and user simulation.

We developed six user categories for Batmen, corresponding to the six behaviors of Figure III.1. They take the form of six classes inheriting from the class `ReplayUser`: `ReplayUserRigid`, `DMUserReconfig`, `DMUserDegradSpace`, `DMUserDegradTemp`, `DMUserRenounce` and `DMUserDelay`. These user categories take two inputs:

- their workload in Batsim JSON format
- the DR window, represented by a pair of dates $(begin, end)$

The users will replay their input workload except in the DR window, where they act according to their behavior.

2.2 Workload

We chose a recent workload trace from the Parallel Workload Archive: the 2-year trace from MetaCentrum (national grid of the Czech Republic)[1]. The platform is very heterogeneous and underwent majors changes during the logging period [114]. For the purpose of our study, we perform the following filtering:

1. The workload is truncated to keep only 6 months (June 1 to November 30, 2014).
   We chose this period as there was no major change in the infrastructure.

2. All the clusters whose nodes have more than 16 cores are removed.
   We target a homogeneous simulated platform with 16-core servers.

3. All jobs with a number of requested resources greater than 16 are removed.
   Our multicore scheduler forbids cross-server execution.

4. All jobs with an execution time greater than one day are removed.
   We want to avoid having too much inertia in our system.

Steps 1 and 2 keep a number of servers from the original platform totaling 6304 cores. From the truncated workload, steps 3 and 4 exclude 2.7% of jobs, making up 73.7% of the mass (in core-hour).

The distribution of requested cores in the filtered workload is represented in Figure III.3. The median job duration is 136 seconds and 3213 seconds on average.

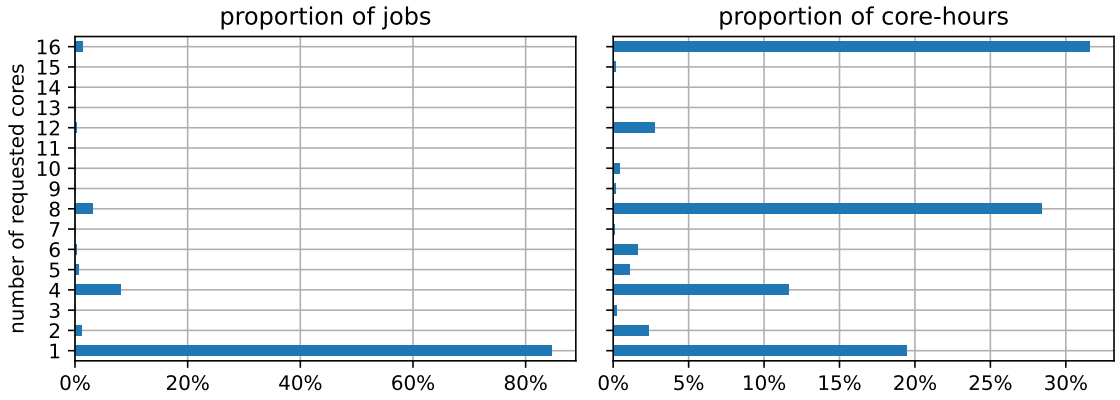


Figure III.3: **Distribution of requested resources in the filtered workload**

2.3 Platform

We create a simulated platform adapted to the filtered workload, with $6304*(1-0.737)/16 =$ **104 homogeneous 16-core servers**.

The power constants for the energy model (see Chapter II, Section 1.4) are given in Table III.1.

[1] file `METACENTRUM-2013-3.swf` available at https://www.cs.huji.ac.il/labs/parallel/workload/l_metacentrum2/index.html

Table III.1: **Power constants for the servers** and time to switch on (T_{son}) and switch off (T_{son}). Measurements in Taurus Grid'5000 cluster made by Guyon et al. [103].

P_{idle}	P_{core}	P_{off}	P_{son}	P_{soff}	T_{son}	T_{soff}
100 W	7.3125 W	9.75 W	100 W	125 W	150 s	6 s

2.4 Experimental campaign

Let's recall the research question raised in the introduction of this chapter: *What is the effect of each sufficiency behavior to reduce energy consumption, and how do they compare?* To investigate it, we adopt the following scenario:

Working scenario We imagine a data center functioning at nominal load. Some jobs are currently running, and the users have the possibility to submit their jobs to the scheduler. Suddenly, the data center operator receives a warning that a peak electricity consumption is detected. He forwards this alert to the users of the platform. They decide to adapt their submissions by adopting a sufficiency behavior. At the end of the alert, the users return to their normal behavior.

Experimental campaign We simulate the aforementioned scenario on every weekday (Monday to Friday) of our input workload between June 1 and October 23, 2014. This makes a total of 105 different experiments. For each day, we start the simulation 24 hours before and finish it 24 hours after, so that the total simulation duration is three full days of data center operation. This way, and since the selected jobs in the workload have an execution time lower than one day, we make sure that the infrastructure runs at nominal load on day 2 and has absorbed the event by the end of day 3. The demand response event arises at 16:00 on day 2[2]. We study two lengths for the DR window: one and four hours[3]. For the purpose of characterization, we try the sufficiency behaviors one by one, i.e, we assume that all users adopt the same behavior during the DR window.

Total number of simulations Eleven simulations are launched for each experiment: the baseline simulation with all users keeping a Rigid behavior, and the five other behaviors, on the two window lengths. In the end, 105 days $*$ (1 baseline + 5 behaviors $*$ 2 window lenghts) = **1155 3-day simulations** are launched.

Time and carbon footprint of the campaign The campaign launched in parallel on a 2 x 8-core Intel Xeon E5-2630 v3 machine completed in less than two hours. Launched in France and considering only the electricity consumption to run the campaign, this has a carbon footprint of around 25 g CO2e (calculated using `https://green-algorithms.org` v2.1 [115]).

[2]The choice of taking a weekday and this specific time of day is justified by a characterization of 26 years' coincident peak pricing data [112], given that the MetaCentrum trace also displays diurnal and weekday/weekend patterns.

[3]We also tried other starting times (drawn at random) and other window lengths (0.5 and 2 hours) but decided not to report their results here as they are not leading to different conclusions.

Reproducibility All the material to reproduce our experimental campaign and its analysis are available in a GitLab repository[4]. It contains the Nix files defining the software dependencies, the scripts to launch the experiments and the Notebooks to analyze the output data and produce the graphs included in this chapter.

3 Results: energy and scheduling metrics

The experimental campaign produces around 10 gigabytes of output data, including scheduling and energy consumption logs. In this section, we start by explaining the results on a single experiment. Then, we look at the distribution of the 105 inputs. Finally, we analyze the outputs by presenting the effect of each behavior on energy consumption and scheduling metrics.

3.1 Results on one experiment

The results of one experiment consist of a set of outputs produced by Batsim. We use two of them:

- The job trace: a CSV file with one line per job, containing all the information about the jobs (id, timestamps of submission, start and finish, final status, etc.). Thanks to the job traces we calculate the *scheduling metrics* presented in 3.3.
- The energy consumption trace: a time series of energy consumption in the simulated platform. From it, we caclulate the *energy metrics* presented in 3.4.

The job trace gives us an exhaustive source of information on what happened in the platform. Thanks to the visualization tool Evalys[5], we provide a graphical representation of it on an example in Figure III.4.

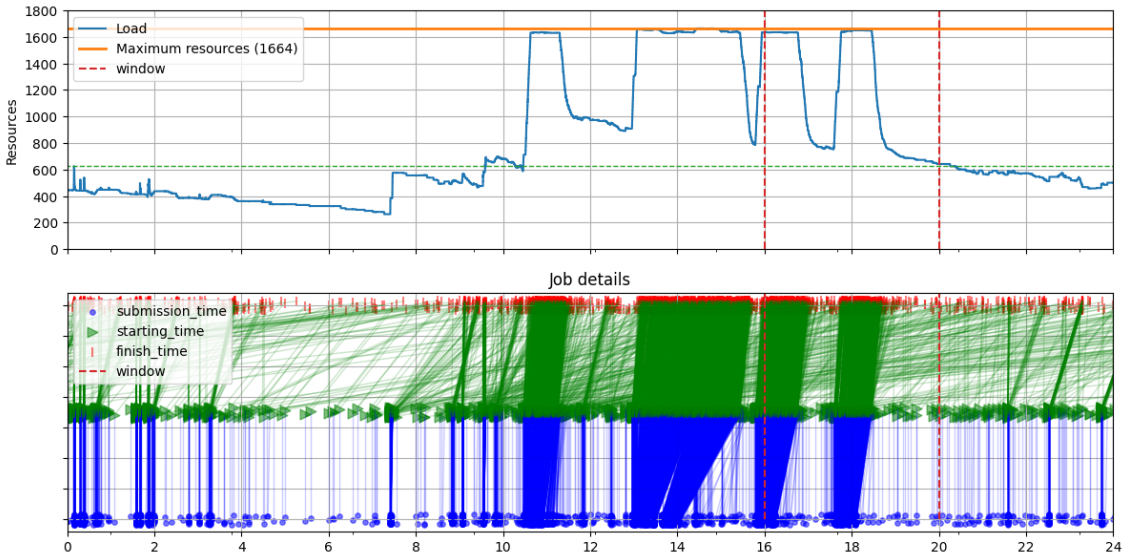
Comments on Figure III.4 In this example, by looking at the bottom graph for behavior Rigid (Figure III.4a), we see that few jobs are submitted during the night (between 2:00 and 8:00). The burst of activity is between 10:00 and 18:00, which corresponds to usual working hours. Most of the time, the submitted jobs are started immediately by the scheduler: the line between the blue dot and green triangle is vertical. The only periods when the job executions are delayed with regard to their submission times are between 13:00 and 14:00, around 16:00, and around 18:00. These periods correspond to periods of saturation in the infrastructure, as we see in the top graph: almost all the cores are in use, the new jobs have to wait in the queue.

The effect of adopting the behavior Delay during the DR window is clearly visible in Figure III.4b. The left part of the figure, before 16:00, is identical to Rigid. But during the DR window, no jobs are submitted by the users. They submit them all at once at the end of the window instead, i.e., at 20:00. This translates into a decreasing number of computing cores during the DR window, as we see on the top graph. The load peeks to the maximum at the end of the window.

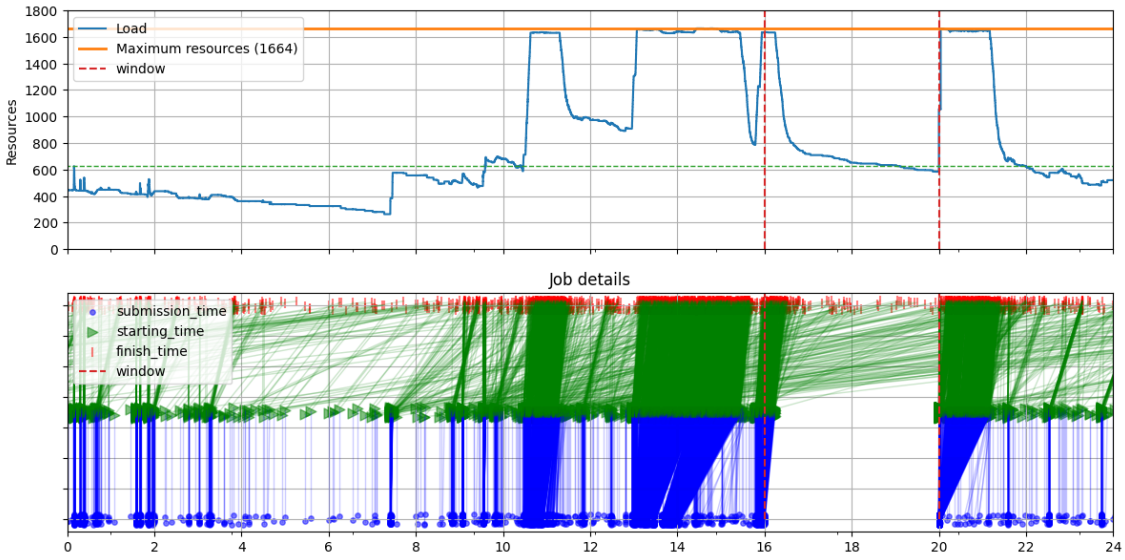
Same representations for the four other behaviors on the another day can be found at the end of the notebook of experiments (no need to download the repository, the notebooks are browsable on the GitLab platform).

[4] experiment repository available at `https://gitlab.irit.fr/sepia-pub/open-science/demand-response-user` (use the tag maelPhD)

[5] data analysis library developed in Python: `https://evalys.readthedocs.io/`



(a) Rigid behavior



(b) Delay behavior, DR window between 16:00 and 20:00

Figure III.4: **Example of output from one simulation**: Tuesday June 5 2014, with two behaviors. Visualization using Python library Evalys.
How to read these graphs? X-axis: time in hour. Bottom graphs: a blue dot represents a job arrival, a green triangle the beginning of its execution and a red line its end. Top graphs: the number of computing cores in the infrastructure. The maximum number of cores is 1664, represented by the orange horizontal line.

3.2 Statistics on the input data

From the previous example, we understand that the potential of each behavior to reduce the load during the DR window greatly depend on the shape of the input. In particular, the number of jobs submitted during the window might influence the results. Same goes with the baseline server utilization during that window (computing load with behavior Rigid). Figure III.5 provides information on the distribution of our input data.

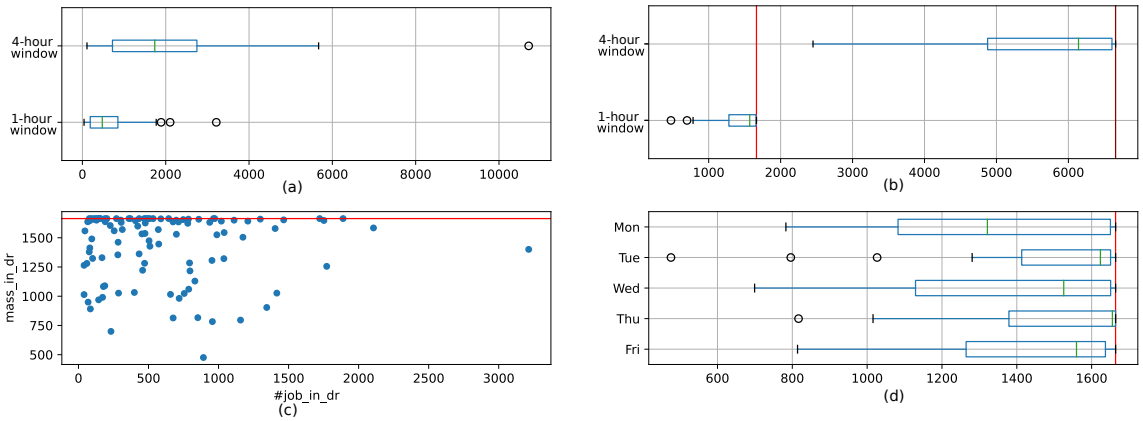


Figure III.5: **Descriptive statistics for the 105 experiments.**
(a) number of jobs submitted in window;
(b) computing load (in core-hour) in window in Rigid experiment;
(c) computing load in window by number of submitted jobs (1-hour window);
(d) computing load in window by weekday (1-hour window).
The red lines correspond to the maximum load reachable in our 1664-core infrastructure.

How to read the boxplots?

In this manuscript, we often present data with box-and-whisker plots. When we do so, we follow the conventions taken by pandas, the Python visualization library that we use. From `pandas.DataFrame.boxplot` documentation:

> The box extends from the Q1 to Q3 quartile values of the data, with a line at the median (Q2). The whiskers extend from the edges of box to show the range of the data. By default, they extend no more than 1.5 * IQR (IQR = Q3 - Q1) from the edges of the box, ending at the farthest data point within that interval. Outliers are plotted as separate dots.

When displayed, the mean of the data is represented by a triangle.

Comments on Figure III.5 First, we observe in (a) a great variability in the number of jobs submitted by the users during the DR windows. For example, for the 4-hour windows, these numbers range from min = 110 to max = 10713, with 50% of the points being between Q1= 722 and Q3 = 2749. *We remind that these jobs are the only jobs on which the sufficiency behaviors can have an impact.*

From (b), we note that more than half of the inputs produces a saturation of the infrastructure during the DR window, with Rigid behavior. In fact, the median load during the DR window is 1570 (resp. 6169) core-hours, for a maximum reachable load of 1664 (resp. 6656). In other words: the DR windows coincide with periods of high activity

in the platform. This activity is highest on Thursdays, followed by Tuesdays and Fridays (looking and the median in Figure (d)).

It is important to note, however, that the baseline load in the DR window and the number of jobs submitted during it are *not* correlated (Figure (c)). This is due to the fact that (i) jobs can have very different sizes (number of resources and execution time) i.e., contribute very differently to the load, and (ii) the load during the DR window also depends on jobs previously submitted to the platform.

3.3 Impact of sufficiency behaviors on energy

In this section, we present and compare the impact of each behavior on energy consumption during the DR window, then overall.

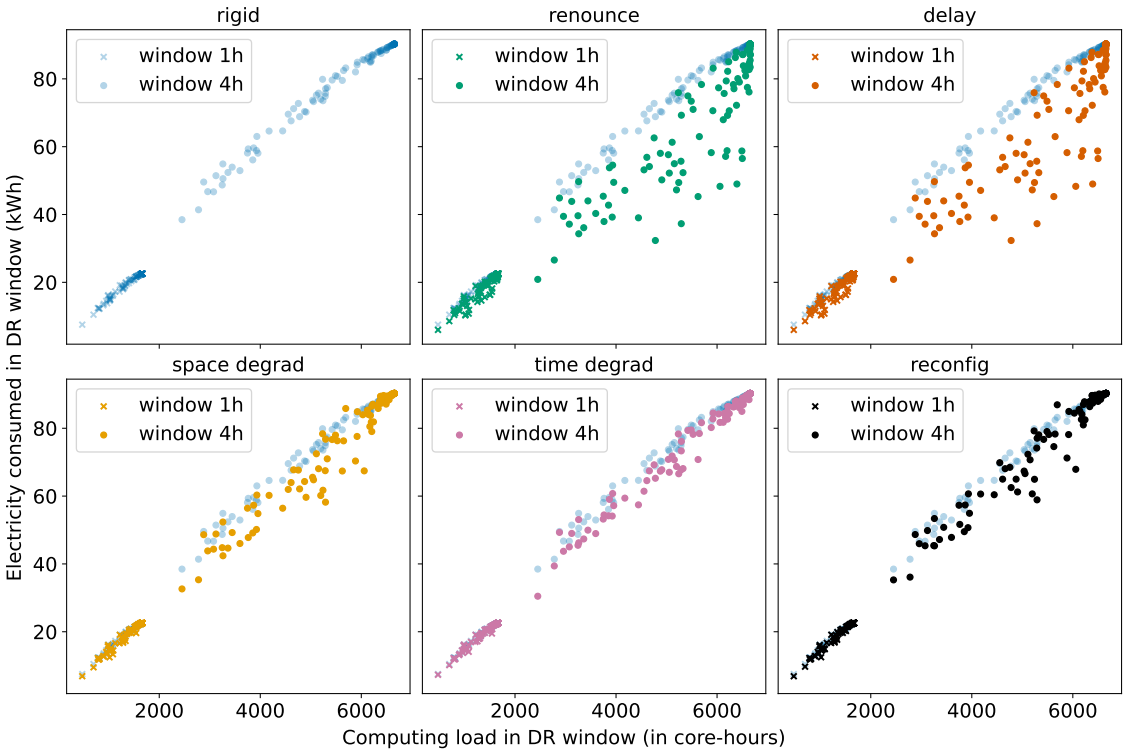
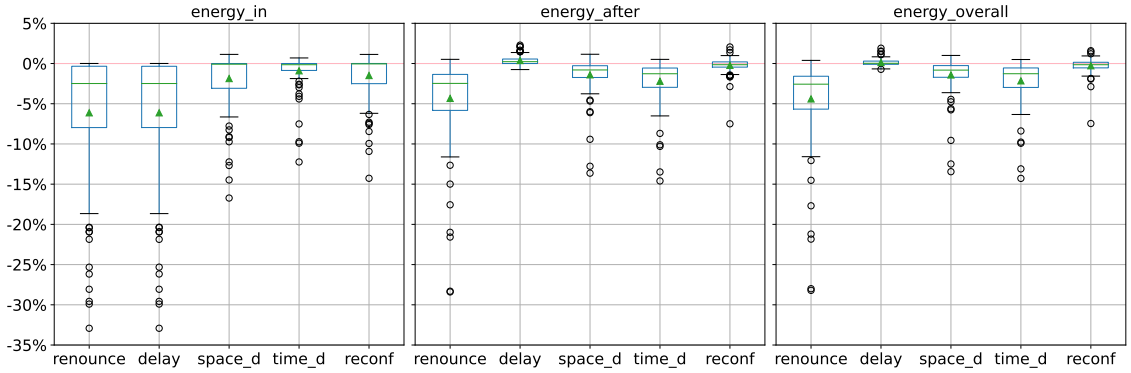


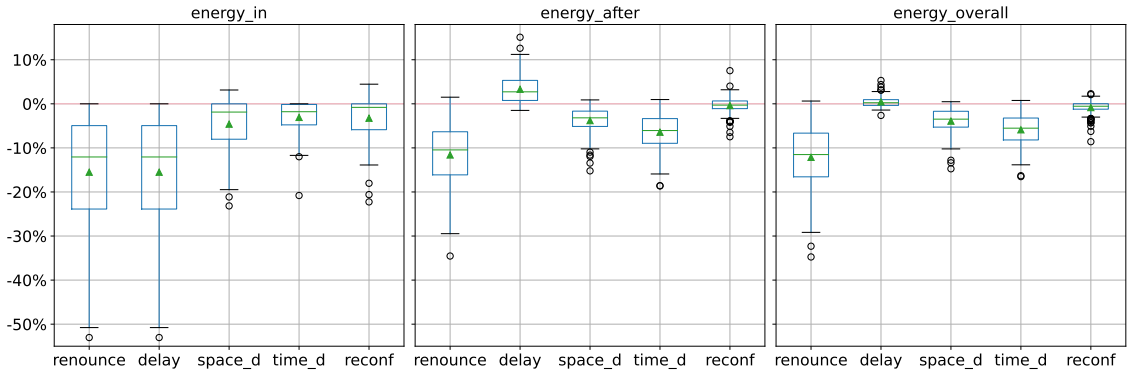
Figure III.6: **Energy consumed in each simulation.**
Y-axis: energy consumed (in kWh) during the demand response window. X-axis: computing load (in core-hour) in window for the baseline behavior.

3.3.1 Energy in window

Figure III.6 displays the energy consumed during the DR window for every experiment and every behavior. Values are scattered by the total load in the infrastructure during the window for the baseline (i.e., Rigid) behavior. For that behavior, we observe an almost linear relationship between load and energy consumed. This is due to our energy model described in Section 1.4 being almost linear with the number of cores used (see more explanation in Appendix 6). Deviations from the linear line are due to situations favoring a more or less optimal packing from the scheduler inside the 16-core servers.
For the five other behaviors, the data points for Rigid are left as a comparison. Overall, we observe a vertical shift of the data points corresponding to sufficiency behaviors. This



(a) 1-hour demand response window



(b) 4-hour demand response window

Figure III.7: **Energy metrics per behavior relatively to the baseline behavior.** `space_d` and `time_d` stand for Space Degrad and Time Degrad, respectively.

means that they consumed less energy globally, compared to the baseline.
Behaviors Renounce and Delay perform identically for this metric. Indeed, inside the DR window, their effect is the same: users stop submitting. The result is a lower energy consumption compared to the baseline. This gain is the best we can expect.
Behaviors Space Degrad and Reconfig display results that are similar to one another. In fact, only during the DR window, these two behaviors have the similar effect of reducing the number of resources required. The difference comes in the longer term, because reconfigured jobs execute for longer than degraded ones.
Finally, Time Degrad seems the less impactful of all on this metric. The reason is that temporal degradation has no immediate effect, but rather leads to a load reduction in the middle term, since the jobs have shorter execution times.

In addition, one would expect a positive correlation between the baseline load of the platform and the relative energy gains of the five behaviors compared to Rigid. It would translate into an increasing distance between the colored dots and the blue dots in the graphs, as the load increases. Counter-intuitively, this does not seem to happen. This phenomenon will be explained below.

Since the experimental campaign shows very scattered results, we chose to represent the energy gains as box plots, relatively to the energy consumed in the baseline (Figure III.7). We can read for example on the leftmost graphs that Renounce, the most radical behavior, allows energy savings of up to 33% in the window for a 1-hour window, and 53% for a 4-hour window (6.1% resp. 15.5% on average). The savings do not go up to 100% because jobs that were submitted before the window are still running in the infrastructure.

3.3.2 Energy after the window, and overall

In addition to the energy consumed *in* the window, Figure III.7 shows the impact of the different behaviors on the energy consumed *after* the demand response event, i.e., from 17:00 or 20:00 on day2 (depending on the window length) to 24:00 on day3. For this second metric, Delay performs very differently compared to Renounce. All the jobs originally submitted within the window get postponed, resulting in an extra power consumption after the window: +0.4% (resp. +3.3%) on average for a 1-hour (resp. 4-hour) window. This behavior remains neutral with respect to overall energy consumption (*in* + *after* the window).
Interestingly, Reconfig, which is the other behavior to preserve the mass of the jobs, yields better results in some cases. Up to -8.6% overall energy consumption could be reached because the reconfigured jobs "fit better in the holes" with the bin-packing algorithm. This phenomenon was already observed by Guyon et al. [103]. It remains marginal in our case, with 0.3% and 0.9% average reductions in overall energy for 1-hour and 4-hour DR window, respectively.
Space Degrad yields better results than Reconfig after the window and overall. We recall that they were similar during the window. As stated before, the difference is that Space Degrad truely cancels some load, while Reconfig increase the duration of the jobs, postponing the load to later.
Finally, if Time Degrad is the worse behavior in terms of energy gains in the window, it proves significantly better than Space Degrad after, and overall. The reason is to be found in the distribution of our inputs. In fact, we recall that Time Degrad affects *all the jobs*, and divide their execution time by two (see "Important remarks" in Section 1.2.2). In contrast, Space Degrad makes a rounding when diving the number of resources. This effectively leads to a halving of the mass only when the number of resources is even. In particular, this behavior has no effect on jobs requiring only one core, accounting for 85% of jobs in the filtered workload and 21% of the mass (see Figure III.3).

From the comparison of Figure III.7a with III.7b, we can say that **the larger the window, the better the energy gains**. This is due to inertia of the system: with a longer window, a behavior on the submitted jobs has more time to make a difference compared to the residual jobs that are still running in the infrastructure. This effect will be analyzed further in the Discussion.

All the energy results are summarized in a ranking of behavior in Table III.2.

3.4 Impact of sufficiency behaviors on scheduling metrics

In this section, we present and compare the impact that each behavior has on the scheduling. For this, we start by defining our scheduling metrics, display them on the baseline experiment, and finally on each behavior.

3.4.1 Definitions: waiting time and slowdown

We use two usual scheduling metrics from the literature: mean waiting time and mean slowdown.

Definition 10. *The **waiting time** of a job is the time that elapsed between the submission of the job and the beginning of its execution in the infrastructure. With the formalism from Section 1.2:*

$$waiting\ time = start\ time - submission\ time = s - a \tag{III.1}$$

Mean waiting time is the average waiting time over all the jobs in the simulation. A common reproach made to this metric is that it gives as much weight to waiting times measured on short jobs than on long jobs. Arguably, a delay of one hour on a 30-second job is less acceptable for a user than the same delay on a three-day job. A solution is to use slowdown, that gives an expression of the waiting time relatively to the length of the job:

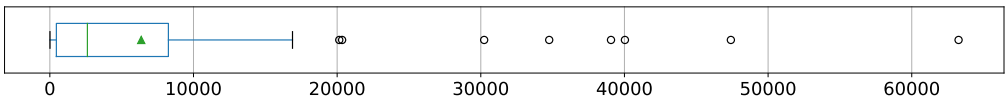
Definition 11. *The **slowdown** of a job is the total time spent by the job in the system divided by its execution time.*

$$slowdown = \frac{finish\ time - submission\ time}{execution\ time} = \frac{f-a}{d} = 1 + \frac{waiting\ time}{d} \tag{III.2}$$

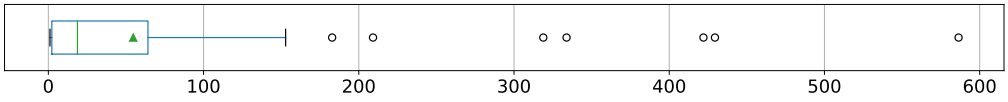
For each experiment, the mean waiting time and mean slowdown are calculated on all jobs submitted in the same period as the metric *energy_in + energy_after*: between 16:00 on day2 (beginning of the DR window) and 24:00 on day3 (end of the experiment).

3.4.2 Scheduling metrics on the baseline

Figure III.8 shows the scheduling metrics for the Rigid behavior. We observe that for more than half of the experiments the mean waiting time is below one hour (median = 2614 s) and the mean slowdown below 19 (we remind that the median duration in our input workload is 136 s). These are cases where the infrastructure is not saturated during the studied period, and the queue of waiting job is often empty. On the other hand, there are also cases of high congestion, for example the six outliers at more than 30000s = 8.3h mean waiting time.



(a) Mean waiting time (in seconds)



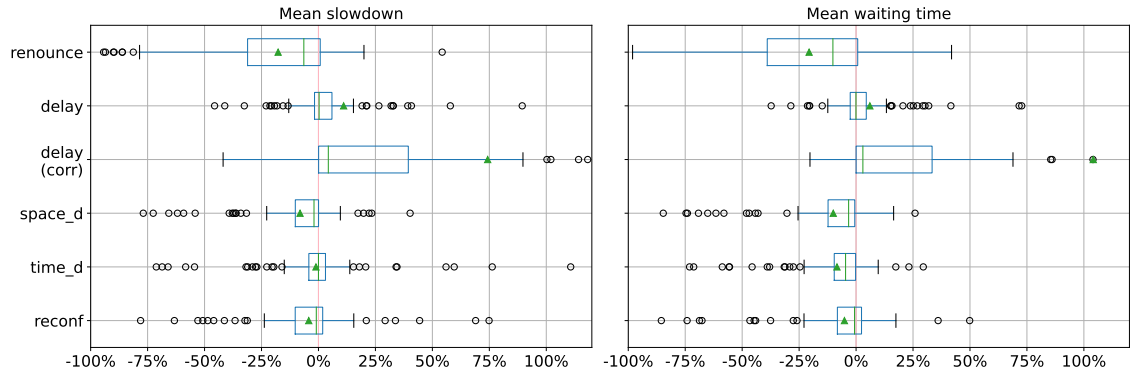
(b) Mean slowdown (dimensionless)

Figure III.8: **Scheduling metric distribution for the 105 experiments, baseline behavior** (calculated between 16:00 on day2 and 24:00 on day3).

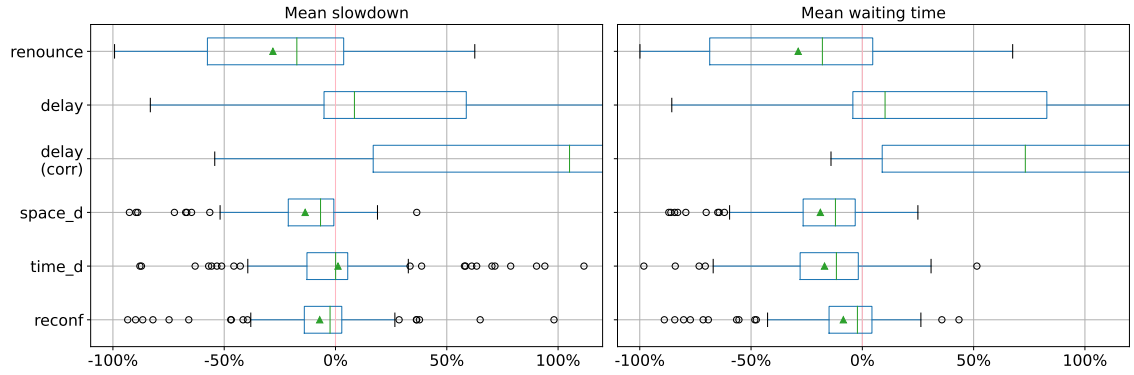
3.4.3 Scheduling metrics on sufficiency behaviors

The results for the other behaviors are plotted in Figure III.9. The values are displayed relatively to the baseline. Interpretation: if the mean for behavior X is at −25% on the metric *mean slowdown*, it means that behavior X is has a mean slowdown 25% lower than the baseline, on average, over the 105 experiments.

Adapting the scheduling metrics One problem is that the behaviors modify basic characteristics of the jobs, some of which — namely submission time and execution time — are used to define the scheduling metrics. Consequently, we have to make adaptations.



(a) 1-hour demand response window



(b) 4-hour demand response window

Figure III.9: **Scheduling metrics per behavior relatively to the baseline behavior.** The metrics are calculated between 16:00, day2 and 24:00, day3. For behavior Delay, "(corr)" indicates the use of corrected metrics, where we use the original submission time to calculate the metrics (and not the submission time after delay).

- For behavior Renounce, some jobs are canceled. Their start and finish times are therefore not defined. In this case, the mean waiting time and slowdown are calculated only on the subset of non-renounced jobs, which is a different subset than the baseline or the other behaviors.
- Behaviors Space Degrad, Time Degrad and Reconfig may change the execution time, used in the definition of slowdown. In this case, the slowdowns are calculated with the *new* execution times.
- For the behavior Delay, we provide both *corrected* and *uncorrected* metrics. The uncorrected slowdown and waiting time are calculated in relation to the *new* (delayed) submission times, while the corrected ones use the *original* submission times (from the baseline).

Comments on Figure III.9 Looking at the distribution of mean waiting times, it can be observed that the four behaviors Renounce, Space Degrad, Time Degrad and Reconfig (in this order) shorten the waiting time on average. This is not surprising, as the first three behaviors decrease the total mass of jobs to compute, and the fourth allows for a better packing. Yet, this metric get worsened in a significant number of cases (around 50% of the cases for Reconfig and 25% for the two Degrad and Renounce), probably due to bad

choices of the non-clairvoyant scheduler.
The behavior Delay stands out from the others as it affects the scheduling negatively in most cases, even for the uncorrected metrics. It gets even worse when including the extra waiting time from the delayed job in the calculation of the corrected metrics. In fact, Delay leads to a burst of submissions at the end of the DR window, leading to congestions in the queues, hence higher waiting time and slowdown.
Note that the distributions of mean waiting time and mean slowdown look very similar. Only for behavior Time Degrad do we observe a difference: it ranks third in mean waiting time, but fourth in mean slowdown (ranking made by considering the means in the distributions). The distribution of mean slowdown for this behavior is actually centered around zero, which means that Time Degrad yields results similar to the baseline for this metric. The explanation is that Time Degrad has two effects that counter-balance each other for the metric *mean slowdown*. On the one hand, this behavior reduces the submitted load, which in turn reduces the *waiting time* of most jobs. The waiting time is in the numerator in the definition of slowdown (last formula of Equation III.2). But on the other hand, Time Degrad also reduces the *execution time* of jobs submitted in the window, appearing in the denominator of the same formula. If we had used the original execution times rather than the new (degraded) ones, results would have been probably similar to waiting time results.

Same as the energy results, scheduling results are summarized in Table III.2 in form of a ranking.

4 Discussion

In this section, we provide an explanation of the energy results thanks to two quantities that we introduce: the fluid and residual mass. Then, we synthesize the results of each behavior and provide a comparison of them.

4.1 The fluid-residual ratio: an explanation of the results

We saw previously when analyzing the achievable energy savings in the DR window on Figure III.6 that the gains cannot be explained by the infrastructure load in the window. In fact, it is possible that the load is very high because of a large mass of job submitted *before* the window, although the load on which the users have an influence is the mass submitted *during* the window. We call these two quantities the **residual mass**, submitted outside the window, and the **fluid mass**, submitted inside the window. Fluid and residual mass are represented in an example in Figure III.10.

Since our approach does not impact the residual mass, we understand from Figure III.10 that the maximum gains that can be achieved with the sufficiency behaviors is to remove all the fluid mass (which is done by Renounce). In other words, **the relative energy gains during the window are at most equal to the proportion of fluid mass in the window**.

To verify this claim, we plotted in Figure III.11 the energy gains as a function of the fluid-residual ratio. We added a dashed and dotted line of equation $y = -x$, indicating the best possible gains. For example, a dot on this line means that 30% energy reduction is reached when the ratio fluid-residual in the window was 30%-70%. We see that no dot is under this line. The best possible gains are almost achieved by Renounce and Delay, though they hardly ever reach the line. The reason is the non-linearity of the energy model, as explained in Appendix 6. In fact, even if a user renounces a 6-core job, the server it was supposed to run onto might either remain powered on because it

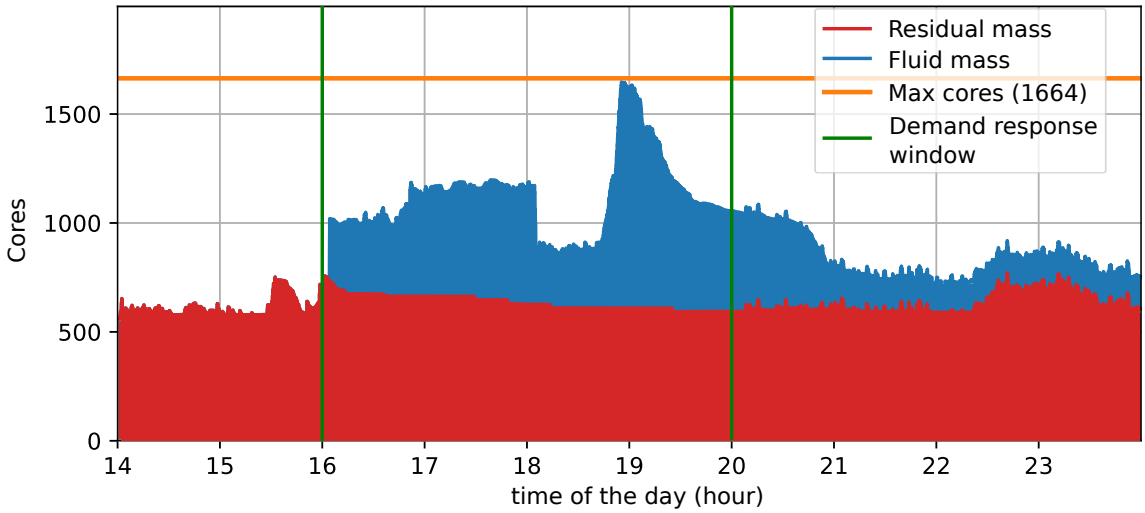


Figure III.10: **Example of fluid and residual mass** (Thursday Jun 26 2014)

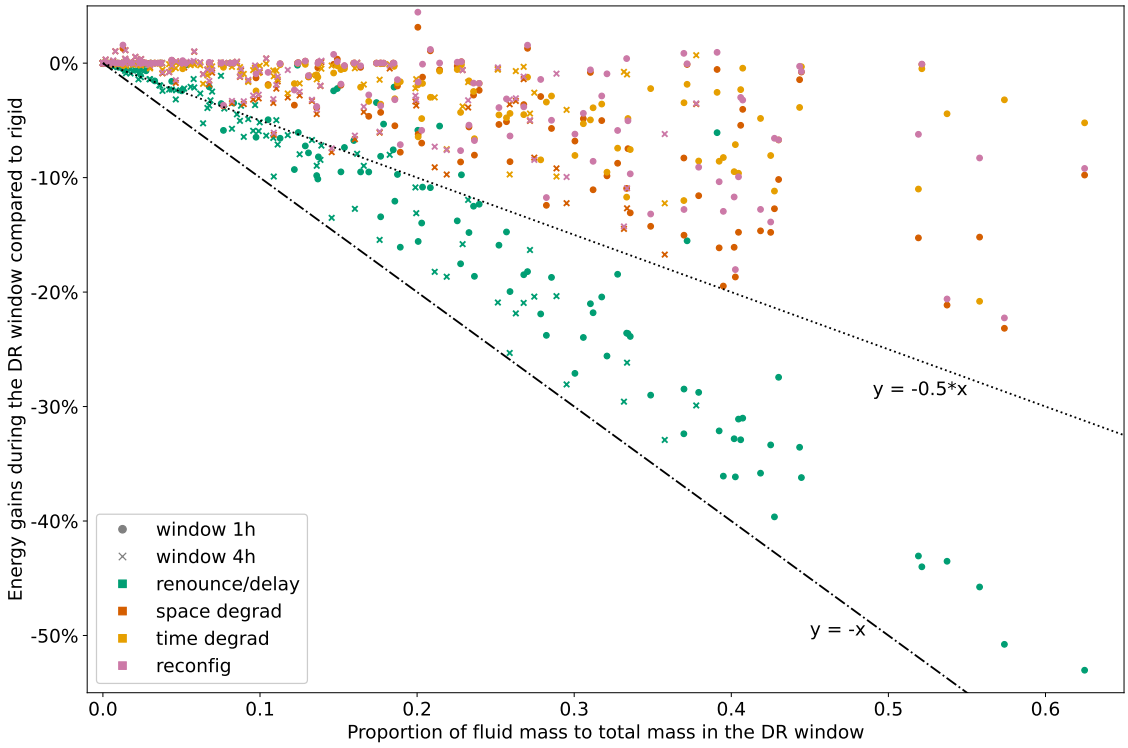


Figure III.11: **Energy gains in function of the fluid-residual ratio.** Only one plot for the behaviors Renounce and Delay because they are identical for this metric.

has something else to compute, or be switched off. In the first case, the server saves the dynamic power consumption, but still consumes the fixed P_{idle} part. In the second case, the server continues to consume some energy during the switching off period.

In some cases, however, Renounce and Delay dots are far away from the line. These are cases of **saturation**, when many jobs are waiting in the queue. The removal of the fluid mass is fully or partly compensated by the execution of the awaiting residual mass.

For Space Degrad and Time Degrad, gains are expected to be of half the fluid mass at most, since users divide their submitted mass by two. Reconfig is expected to have the same maximum, with even less gains in practice, since it preserves the mass. This second optimal is represented by the dotted line, of equation $y = x/2$. In practice, the results are even more scattered and further away from their optimal than Renounce/Delay. This is due to several factors: (i) the energy/load non-linearity and saturation effects mentioned above, (ii) the rounding up when dividing by two the resources for Space Degrad, and (iii) the temporality of the gains for Time Degrad: the load diminution is not achieved straight away but after half of the original execution time elapsed.

Note that we still observe a few experiments (with Space Degrad and Reconfig) that do even better than the second optimal. These are probably cases where the baseline had a very bad configuration (many idle cores inside switched on servers), and freeing some resources allowed the scheduler to achieve a better packing, leading to even more energy gains than half of the fluid mass saved that way. On the other end of the spectrum, some experiments make negative gains. These are saturation cases where the baseline also had a bad configuration i.e., many idle cores. Applying the behavior led to smaller jobs that could better fit in the holes, but no energy gains because of numerous waiting jobs in the queue.

4.2 Pros and cons of each behavior

Table III.2: **Summary ranking of the five sufficiency behaviors** according to their impact on energy consumption and scheduling metrics.

behavior	energy in	energy overall	sched. met.	"acceptability"
Renounce	1st	1st	1st	5th
Delay	1st	5th	5th	2nd
Space Degrad	3rd	3rd	2nd	3rd
Reconfig	3rd	4th	4th	1st
Time Degrad	5th	2nd	2nd	3rd

energy in: from Figure III.7. The boxplot largely overlap, but Renounce and Delay are the same, best on average, median and max achievable reduction. Space Degrad and Reconfig are very similar, so we placed them *ex aequo*. Time Degrad has the distribution most clustered around zero.
energy overall: from Figure III.7, ranking clear from the boxplots.
scheduling metrics: means for metric *mean waiting time* in Figure III.9.
acceptability is opinion-based, it reflects the size of the effort asked from the user. Justifications are given in the text below.

Based on the results of our experimental campaign presented in the previous section, we provide in Table III.2 a ranking of the five sufficiency behaviors according to the different metrics. This ranking constitutes the characterization of these behaviors. In the following, we discuss this ranking for each behavior.

Renounce First of all, the behavior Renounce performs the best for all the metrics studied. By deleting all the fluid mass, it reaches the maximum energy gains possible with our approach. It also lightens the scheduling significantly. The problem with this behavior is that the sacrifice required from the user is huge, and often unthinkable. For this reason, we ranked this behavior last in "acceptability". Renouncing is radical, but we argue that it is appropriate to consider it among other levers in view of the scale of the efforts required to achieve environmental agreements. We think that such a behavior is too often overlooked in similar studies.

Reconfig On the other end of the spectrum, the behavior Reconfig seems to be the most acceptable to the users, as it does not degrade the initial job and provides better waiting time and slowdown than Delay for both the jobs within and after the window. Reconfig is a good trade-off to achieve some optimizations with a low effort from the user, especially in combination with bin-packing schedulers and on/off policies (see [103]).

Delay Delay also preserves the mass of the jobs, which is why we ranked it second behind Reconfig in terms of acceptability. Same as Renounce, it reaches the optimal energy gains during the window. However, it introduces an overhead in overall energy consumption and slowdown compared to the baseline behavior. Note that this overhead would probably be less important in real life, because (i) all users would not resubmit exactly at the end of the window and (ii) they would probably adapt their submission behavior if they experience congestion. We introduce later in Chapter V a model that takes feedback from the infrastructure into account when replaying workload traces in simulation.

Space degrad The two Degrad behaviors consist in degrading the quality of service. For a job with an even number of required resources, spacial degradation is equivalent to temporal degradation, as it divides by two the quantity of operations performed by the job. In practice, it is not always possible to apply one or the other behavior due to the characteristics of the job. They are therefore ranked ex aequo in "acceptability", second to last behind Renounce. As a result, Space Degrad ranks second or third in all four columns of Table III.2. It is a good trade-off between the radical Renounce and low-effort Reconfig.

Time degrad Compared to Space Degrad, Time Degrad does not provide short-term energy gains. Consequently, it is not adapted in a context of demand response. All the same, it brings interesting results in energy overall and scheduling metrics, especially since it consistently halves the mass of jobs compared to the Space Degrad, as explained before. It is therefore a suitable behavior if short-term gains are not the priority, or if a spacial degradation / reconfiguration is not possible on the job.

5 Limitations

The work presented in this chapter contains a number of limitations that we enumerate below, going from simplifications in the model to more general limits of the approach.

5.1 Model simplifications

In the data center model, latency and bottleneck effects in the communications are not taken into account. Also, we suppose perfect speedup, i.e., a job executed on two cores will take exactly twice longer than the same job executed on four cores. Finally, the energy

consumption accounted for is only the one from the CPUs, neglecting other sources of electricity consumption like memory, network or cooling.

All the same, we do not think that removing these simplifications would fundamentally change our results, i.e., the comparison between behaviors. Moreover, our experiments are reproducible and use powerful simulation tools (Batsim and SimGrid). This allows to overcome these simplifications quite easily if needed.

5.2 Methodological limitations

We identified threats to the validity of our method to answer the research question. The major ones are listed below.

- In the experiments, only one behavior was applied at once, and to all the jobs submitted in the DR window. However, in reality, some behaviors cannot apply to some jobs (e.g., "black-box" jobs that are not reconfigurable). On the contrary, a user could apply a combination of behaviors. We designed the experiments this way in order to exhibit the particularity of each behavior, and not to reflect the reality.
- The experiments were run with only one type of scheduler (bin-packing). There is a risk that it favors Reconfig and Space Degrad more than what another common scheduler (FCFS or easy-backfilling) would have.
- Only one input trace was used: MetaCentrum, which is a research and not an industrial infrastructure. We performed some filtering from it (see 2.2) to fit our experimental constraints. In particular, long jobs were removed from the trace, yet they contribute a lot to the residual mass. As a result, gains from the sufficiency behaviors might be smaller in practice, if the size of the window is not comparable with the duration of the jobs.
- With simulation, we miss more complex human behaviors that might have occurred in a real platform.
- We do not provide a metric to quantify the "acceptability" of behaviors to users. This acceptability depends on the users and the characteristics of the jobs, e.g., their priority.

5.3 On the sufficiency behaviors

The five sufficiency behaviors defined and characterized in this chapter suppose a certain level of knowledge and capacity of action from the users. Firstly, they are limited to "direct data center users" (see model in Chapter II Part 1.1.1), such as HPC users. It is unclear how a sufficiency behavior from an indirect user (like the focus group participants in Chapter VI) would translate in the infrastructure. Secondly, our approach assumes that users have information on when to act and how to act. In practice, a user might not have access to the status of the infrastructure she is using. She might also not have the training or possibility to modify the application she is launching. In fact, the reconfigurations required on the job for Degrad and Reconfig are most of the time non-trivial. Lastly, users do not always have the choice to act. Sometimes, they are employed by a company or need to deliver results that require them to submit jobs. In this case, the incentive has to come from the top.

Our approach is limited to sufficiency behaviors on new job submissions. We did not study user actions on jobs previously submitted, such as killing a running job, putting it

on hold or freezing it in the queue if it has not started yet. These behaviors require another level of user knowledge and involvement, but constitute a powerful lever to go beyond the fluid/residual limit. For that sake, we could also have studied behaviors with anticipation, in which users start acting on their submissions before the demand response event.

Finally, we decided to characterize the effect of the different sufficiency behaviors by looking at electricity consumption and choosing the context of demand response. This is quite restrictive as the concept of sufficiency is much broader than that (see Introduction), including, for example, reducing the demand for computation by rethinking our digital needs. Accounting for electricity consumption only focuses on the use phase, and neglects the other phases of the life cycle (production, distribution, end of life). It also does not take into account the other environmental impacts. Nevertheless, since the literature on digital sufficiency is very young, we remind that this study is certainly perfectible, but also one of the first proposition to study sufficiency levers with simulation.

6 Conclusion

In this chapter, we defined five types of "sufficiency behaviors", i.e., actions that a user can take when submitting jobs to a data center in order to decrease the environmental footprint. These behaviors are delaying, reconfiguring, spatially degrading, temporally degrading and renouncing the job submission. Thanks to an extensive and reproducible simulation campaign on 105 days of a real workload, we were able to characterize the impact of these behaviors on energy consumption and scheduling metrics.

Renounce is unsurprisingly the most efficient behavior, with 16% energy savings on average on a 4-hour time window, if applied by all users during this time window. However, it is also the behavior that asks the biggest sacrifice to the users. Delay and Reconfig require the least effort to users, as they preserve the integrity of the job. Delay is very adapted for short-term gains: it postpones the "fluid mass" in the window, i.e., the computing load due to the jobs submitted during the window. It leads however to a peak of submission in the future which significantly affects the energy and scheduling metrics then. Reconfig allows for some energy savings during the window (-3% on average compared to the baseline if applied for four hours), comparable to Space Degrad. But on the long term, Reconfig is almost a zero-sum game. Finally, the two Degrad behaviors yield interesting results, as they cut off computing load. Time Degrad is not adapted in the short term, but it consistently lightens the load in the future.

Now that we have a better understanding of the characteristics of these behaviors, we are inclined to wonder how they could help a data center functioning more sustainably in a more realistic scenario, where users would adopt a mix of behaviors. This is what we investigate in the next chapter, in the context of intermittent energy production.

Appendix

Energy model quasi-linear with number of cores

Let n_{cores} the number of non-idle cores, and $n_{on}, n_{off}, n_{son}, n_{soff}$ the number of servers switched on, switched off, switching on and switching off at time t ($n_{on} + n_{off} + n_{son} + n_{soff} = N$). Since the power constants are the same for all servers in the platform, we

have from Equations II.1 and II.2:

$$\begin{aligned} P_{DC}(t) &= \sum_{i=1}^{N} P^{(i)}(t) \\ &= \sum_{\text{servers off}} P_{off} + \sum_{\text{servers son}} P_{son} + \sum_{\text{servers soff}} P_{soff} + \sum_{\text{servers on}} (P_{idle} + k^{(i)}(t) P_{core}) \\ &= n_{off} P_{off} + n_{son} P_{son} + n_{soff} P_{soff} + n_{on} P_{idle} + P_{core} \sum_{\text{servers on}} k^{(i)}(t) \end{aligned}$$

Replacing with numerical values from Table III.1:

$$\begin{aligned} P_{DC}(t) &= 9.75 n_{off} + 100 n_{son} + 125 n_{soff} + 100 n_{on} + 7.31 n_{cores} \\ &\approx 10(n_{off} + n_{son} + n_{soff} + n_{on}) + 90(n_{son} + n_{soff} + n_{on}) + 15 n_{soff} + 7 n_{cores} \\ &\approx 10N + 90(N - n_{off}) + 15 n_{soff} + 7 n_{cores} \end{aligned}$$

From the last equation, we see that P_{DC} depends on the number of servers N (constant), the number of servers off and switching off, and the number of non-idle cores. In our case, and with the bin-packing scheduler with greedy shutdown, it is reasonable to make two additional assumptions to simplify the equation:

- we neglect the time to switch on and switch off, so $n_{soff} = n_{son} = 0$ and $N = n_{off} + n_{on}$
- we suppose perfect packing, i.e., $n_{on} \approx n_{core}/16$

This way, we obtain:

$$P_{DC}(t) \approx 10N + 90 n_{on} + 7 n_{cores} = 10N + \frac{90 * n_{core}}{16} + 7 n_{cores} = 10N + 12 n_{cores} \quad \text{(III.3)}$$

Which explains why we observe an almost linear relationship between energy consumed and number of non-idle cores in Figure III.6.

Chapter IV

Sufficiency behaviors to deal with intermittent energy sources

Contents

The previous chapter introduces five "sufficiency behaviors", that direct users of a data center can adopt to reduce the environmental footprint of their job submissions. The associated experimental campaign tests the behaviors one by one, as if all the users would choose the same behavior. This is useful for characterizing precisely the effect of each behavior, but remains quite theoretical. In reality, we can expect that users would adapt their submission behavior differently depending on their own context, incentives and the nature of their jobs.

As electricity is coming more and more from renewable sources, we see a potential for sufficiency behaviors in the management of intermittency of electricity production. Similarly to the Low-tech Magazine[1] [116], a solar-powered website that goes offline when battery is low, we propose to study a "sufficient data center": a renewable-energy-powered IT infrastructure in which the users contribute the environmental effort by adapting their submission behavior when energy is scarce.

We adopt the same outline as in the previous chapter: problem statement in Section 1, description of the experimental campaign in Section 2, followed by the results (Section 3) and a discussion in which we answer the research questions (Section 4). Finally, the limits of the approach are exposed in Section 5 before closing the chapter on concluding thoughts.

The work presented in this chapter was made in collaboration with Jolyne Gatt, an intern at our team in the period from February to July 2023. Jolyne was supervised by Georges Da Costa and myself. She suggested the approach, implemented it in Batmen and run the experiments. The data analysis as well as the writing was done jointly. This work was accepted for publication at the conference ICT4S 2024 [C4].

[1] https://solar.lowtechmagazine.com

1 Description of the approach

This section provides some context on renewable-energy-powered data centers, before stating our research questions. Then, we explain our approach by introducing the three-state energy feedback model, and the way it is combined with the sufficiency behaviors.

1.1 Context

Data centers and renewable energies To reduce the environmental impact associated with electricity supply to their data centers, large IT companies have been taking 100% renewable energy commitments in the last decade [49]. Most of the time, these commitments are achieved through green tariffs or power purchase agreements [49]. A problem with these contracts is that they do not directly create the new renewable capacity necessary to cover the electricity demand. For this reason, companies have started going one step further, and installing on-site renewable sources on their data centers (e.g., Google in Belgium [117]). Often, the renewable electricity produced is actually sold to the market, leaving the burden of balancing demand with intermittent supply to grid operators. The last step for a truly sustainable data center would be to self-consume this electricity to become fully autonomous in energy supply. This is the aim of ANR project DataZero2, in which I have been involved during my PhD.

DataZero project DataZero[2] is a consortium of academics and industrials investigating since 2015 new ways to power and manage a data center operated only by renewable energies [50]. The architecture includes photovoltaic panels, wind turbines, fuel cells and batteries to produce and store electricity. These different units are managed by the Power Decision Module, in charge of arbitrating the source involvement decisions with regard to weather forecast and internal objectives. In mirror, the IT Decision Module delivers a plan for server state (switched on/off, computing frequency) based on IT demand forecast and power envelope. The two modules interact with each other to find an agreement via a Negotiation Module.

1.2 Research questions

The approach presented in this chapter is part of this context. We consider an off-grid data center and investigate the digital sufficiency behaviors' potential to reduce the load in periods of low electricity production. We suppose that the users are informed of the status of renewable production at the moment of submission through a three-state energy feedback mechanism: green state when production is abundant, red state when production is low, and yellow in-between. We aim to provide answers to the following research questions:

(1) How much does user effort impact energy consumption in critical periods? Is there a threshold where more effort does not result in more energy savings?

(2) Does action in the intermediate "yellow" energy state have an effect on energy consumption, or is it sufficient to only consider nominal and low-renewable states?

1.3 Energy state model

The users adapt their job submissions to the current state of renewable energy production. When production is low, they are invited to submit fewer and smaller jobs. To simplify the

[2] https://www.irit.fr/datazero/datazero2/

information that is given to them, we define three "energy states", named like the colors of a traffic light:

- **Green state**: The system is alright: there is enough energy to power the whole platform.
  We are in this state when the energy production is greater than the max energy consumption of the platform E_{full}.

- **Yellow state**: The system is disturbed: the production is not enough to power the whole platform, but it can power a big part of it.
  We are in this state when the energy production is below E_{full}*, but above a certain threshold* τ*, that we fixed arbitrarily to* $0.5E_{full}$.

- **Red state**: The system is critical: the production is low, we must reduce the energy consumption of the platform.
  We are in this state when the energy production is below the threshold τ.

This three-state model can be seen as an eco-feedback design [85], providing simple yet actionable information to the users. Compared to a two-state model, it allows for more expressiveness. In our case, the yellow state indicates that a user effort would be appreciated, but the situation is not critical. It can also be seen as a transition state between green and red, giving the information that the system will soon become critical or is not critical anymore but not yet calm. Finally, this model provides a useful abstraction layer. It is easy to change the way the energy states are defined, by selecting different thresholds or changing the metrics on which they are based. For example, one could define the energy states based on level of battery or instantaneous data center power consumption instead of renewable production like here.

Note that our energy state model is entirely defined, for a given renewable energy production data, by the thresholds on max energy consumption (50% and 100%, in our case). Time is discretized into units of time during which the system is considered to be stable. The time intervals during which one energy state occurred are called "state windows" or simply "windows". They can last one or several units of time. For example, if energy production is under the red threshold from 6:00 to 8:00, we say that a "red state window" of 2 hours occurred.

1.4 Behaviors for each energy states

The goal of the "traffic light" approach previously described is to inform the users efficiently on the phases where renewable energy is scarce. In response, we suppose that the users will adopt the sufficiency behaviors defined in Chapter III (Section 1.2 and Figure III.1): Renounce, Delay, Space Degrad, Time Degrad or Reconfig. Among them, only the model for behavior Delay is adjusted to our specific context. In fact, Delay was modelled by postponing the job submission *to the end of the demand response window*, which implies two assumptions. First, it assumes that the user knows when the window will finish, or is at least notified. Second, it supposes that she connects to the system exactly at this time to resubmit the job. With the energy-state approach, the end of a state window is not known in advance, since it depends on real-time information on power production. Moreover, the feedback mechanism is supposed to be *passive*: the information is only given to the users when they are connected to the platform, and not through notifications. Consequently, we will consider a different version of behavior Delay, named See You Later:

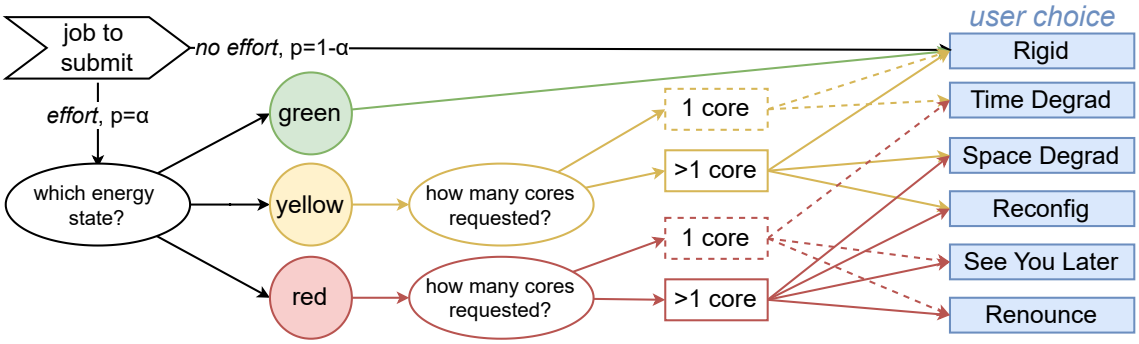


Figure IV.1: **Decision chart when a user submits a job.** The effort probability α represents the probability for users to consider adopting a sufficiency behavior. For a fixed energy state and number of cores, the available behaviors are equiprobable. For example, multicore job in red state: the four available behaviors have a probability 1/4.

See You Later The user decides to postpone the job submission by one hour. After that delay, she will take a new decision on that job.
For example, a user seeing that the current energy state is red could decide to come back and check the situation later. If the state is still critical when she comes back, she might want to postpone again, to submit anyway or apply any of the other behaviors.
➢ model: the submission time is postponed by one hour, and the job will be considered like a new submission.

It gives us a set of six available behavior: $\mathcal{B}$ = { Rigid, Time Degrad, Space Degrad, Reconfig, See You Later, Renounce }. For each energy state, we suppose that users will only consider a subset of $\mathcal{B}$:

- **green state:** there is enough energy, users submit normally.
  subset: { Rigid }
- **red state:** we assume that users will make an effort by choosing any of the other behaviors, depending on their context. Some will accept to reconfigure or degrade the job, if they can. Some will choose to come back later, when the energy state is hopefully better. And some will simply renounce the job, which was maybe not important enough.
  subset: { Time Degrad, Space Degrad, Reconfig, See You Later, Renounce }
- **yellow state:** we exclude the Renounce behavior, as we consider it too radical in regard to the criticality of the state. See You Later is also excluded because a yellow state is an intermediate state, so delaying jobs might push them to red states. However, we added Rigid behavior, as some users might choose to ignore the warning and submit normally.
  subset: { Rigid, Time Degrad, Space Degrad, Reconfig }

The modifications applied to the job submissions depending on the energy state are illustrated in Figure IV.1. Distinction had to be made between monocore and multicore jobs, as space degradations and reconfigurations are not possible on monocore jobs.

Finally, since we have no data on the popularity of each behavior, we will suppose that each time users take a decision, they use a uniform randomized choice. The algorithm used for each user each time they submit a job is presented in Algorithm 4.

Algorithm 4 Algorithm for user submission behavior.

```
// state ∈ [Red, Yellow, Green]
// α is the user effort probability
// cores is the number of cores for the job
function GET_BEHAVIOR(state, α, cores)
    With probability (1 − α) : return Rigid
    if state == Red & cores == 1 then
        return uniform(Time Degrad, Renounce, See You Later)
    else if state == Red & cores > 1 then
        return uniform(Space Degrad, Reconfig, Renounce, See You Later)
    else if state == Yellow & cores == 1 then
        return uniform(Rigid, Time Degrad)
    else if state == Yellow & cores > 1 then
        return uniform(Rigid, Space Degrad, Reconfig)
    else if state == Green then
        Return Rigid
```

2 Experimental setup

In this section, we describe our method by explaining our experimental campaign, its inputs and outputs.

2.1 Description of the experimental campaign

To answer research question (1), we use the data center model from Chapter II, that we extend with the energy state model from Section 1.3. Only one extra information needs to be available to users during the simulation: the energy state (red, yellow or green) at time of submission. This information can be retrieved from any energy production input, using the thresholds defining the states.

In order to vary user involvement, we introduce an additional parameter α, which represents the probability for a user to adopt a sufficiency behavior when in red or yellow state. This parameter is represented at the root of the decision chart in Figure IV.1. We will try four values, from low effort ($\alpha = 0.25$), medium effort ($\alpha = 0.5$), big effort ($\alpha = 0.75$), up to max effort ($\alpha = 1$).

We therefore propose the following experimental campaign. On a given workload and energy production input, we simulate the operation of the data center and its users. For each value of α, we launch 30 replicates, to minimize the effect of randomness. As weather and tasks input are always identical, the only difference between each of the replicates is the random choice of effort and then of the exact behavior, following Figure IV.1 and Algorithm 4. We also add four experiments, for comparison purposes:

- *full rigid*, where the jobs are submitted Rigid all the time (equivalent to $\alpha = 0$). This corresponds to the baseline.
- *full renounce red*, assuming that users would renounce all their submissions during red states. This provides us with an upper bound on achievable energy savings.
- *full degrad red* and *full reconfig red*, same as above with behaviors Degrad (Space by default, or Time if not possible) and Reconfig (when possible).

To answer research question (2), the whole experimental campaign is repeated, without taking into account the yellow state. They are instead treated by users as green states.

In the end, our experimental campaign consists of (4 values for alpha) × (30 repetitions) × (2 treatments) + (4 exp. for comparison) = **244 simulation runs**.

2.2 Experimental inputs

Like in the previous chapter, the experiments were launched with the simulation environment described in Chapter II, Section 2: Batsim (v4.1) and SimGrid (v3.31) for the infrastructure simulation and Batmen for the scheduler and user simulation. The scheduler is the same as in the previous chapter: bin-packing with server shutdown (see Chapter II, Section 1.5, Algorithm 3).

Our software Batmen was extended to support multibehavior feature (release 2.0). A new user class `DMUserMultiBehavior` was created to represent the energy-state-aware users, taking as input:

- a workload in Batsim JSON format,
- two lists of time intervals representing the red and yellow windows,
- the probabilities of choosing each behavior, for each situation (red multicore, red monocore, yellow multicore and yellow monocore),
- a seed for deterministic random number generation.

The IT workload and energy production traces are taken from ANR Datazero project, as well as the corresponding sizing for IT and power infrastructures. We describe them below.

IT workload. Like in the previous chapter, the workload is also adapted from the MetaCentrum trace[3]. The following filtering is performed:

1. Step 1: (excludes 87% jobs representing 86% core-hours)
   - Using only the period from June 1, 2014, to November 30, 2014. This part was taken because it is the longest period in the trace where no cluster was removed or added.
   - Removing clusters with GPU, because our simulation only simulate jobs running on CPU.
   - Removing clusters with more than 18 cores, because our simulated data center is composed of 18-core servers.
2. Step 2: (from the remaining jobs, excludes a further 5% jobs representing, since they are very large, 86% core-hours)
   - Removing jobs running on more than 18 cores, because our scheduler do not authorize multiserver execution.
   - Removing jobs running for more than 15 hours, to mitigate inertia in the system.

After the above filters, our workload is six-month long and contains 693066 jobs and 474 users.

[3] file `METACENTRUM-2013-3.swf` available at https://www.cs.huji.ac.il/labs/parallel/workload/l_metacentrum2/index.html

Energy Production Data. We consider electricity production from photovoltaic panels. The energy input data is generated from weather data (solar irradiation) provided by the website renewables ninja[4]. The reference point for the weather is 2019, in the city of Toulouse, France. Data is provided with one-hour time steps. Note that the weather and workload traces are not from the same year. However, we took care to align the days in the IT workload with the days in the weather, i.e., the workload of June 24, 2014 is replayed with the energy production of June 24, 2019.

For the experiments, we do not take into account energy storage systems. We simply consider that, in periods of low energy production, electricity has to come from other sources, be it batteries, power generator or the electricity grid. However, batteries and fuel cells are considered in the Datazero project, and were taken into account to produce a suitable sizing for both IT and energy platforms (see below).

IT and power platform. Both platforms were created by colleagues, ensuring that the volume of renewable production is sufficient to cover the energy needs from IT, taking into account the efficiency of batteries. The sizing technique used is similar to the one described in this article [56]. The simulated renewable sources consist of $a = 145\text{m}^2$ of solar panels with efficiency $\eta = 0.206$. The power produced at time t is obtained by the formula $\eta \times a \times irr(t)$, with $irr(t)$ the solar irradiation. The simulated IT platform is composed of 42 18-core servers, whose power constants are given in Table IV.1.

Table IV.1: **Power constants for the servers** and time to switch on (T_{son}) and switch off (T_{son}). Measurements in Gros Grid'5000 cluster

P_{idle}	P_{core}	P_{off}	P_{son}	P_{soff}	T_{son}	T_{soff}
62 W	4.52 W	20 W	110.5 W	76.5 W	164 s	6 s

Time and carbon footprint of the campaign. The experimental campaign was launched in one node (2 × 16-core Intel Xeon Gold 6130) of the scientific Grid'5000 platform[5], located in Grenoble, France. It took 7 hours to complete and 55 GB of storage space for the output files, drawing 2 kWh of electricity according to the machine's watt meter. Assuming a carbon intensity of electricity of 38 gCO2eq/kWh[6], this represents a carbon footprint of 76 gCO2eq.

Reproducibility. All the material to reproduce our experimental campaign and its analysis are available in a GitLab repository[7]. It contains the Nix file defining the software dependencies, the scripts to launch the experiments and the Notebooks to analyze the output data and produce the graphs included in this chapter.

2.3 Metrics

From the simulation outputs, we compute both energy- and effort-related metrics. They are defined below and reported later in the Results (Section 3).

[4] open-source weather data repository available at https://www.renewables.ninja/ [118, 119]

[5] https://www.grid5000.fr/w/Grid5000

[6] data from the French distribution operator RTE: https://www.rte-france.com/eco2mix

[7] experiment repository available at https://gitlab.irit.fr/sepia-pub/open-science/sufficient-behaviors-with-renewables (use the tag maelPhD)

2.3.1 Energy-related metrics

Batsim provides us with power consumption logs, thanks to the underlying SimGrid energy model. These logs are compared with the power production data, identical for all simulations. In case the renewable production is in excess compared to the power consumed, we say that we are in a phase of *overproduction*. On the contrary, when the production is insufficient to cover the power consumption, we talk about *underproduction*. These phases are illustrated in Figure IV.2.

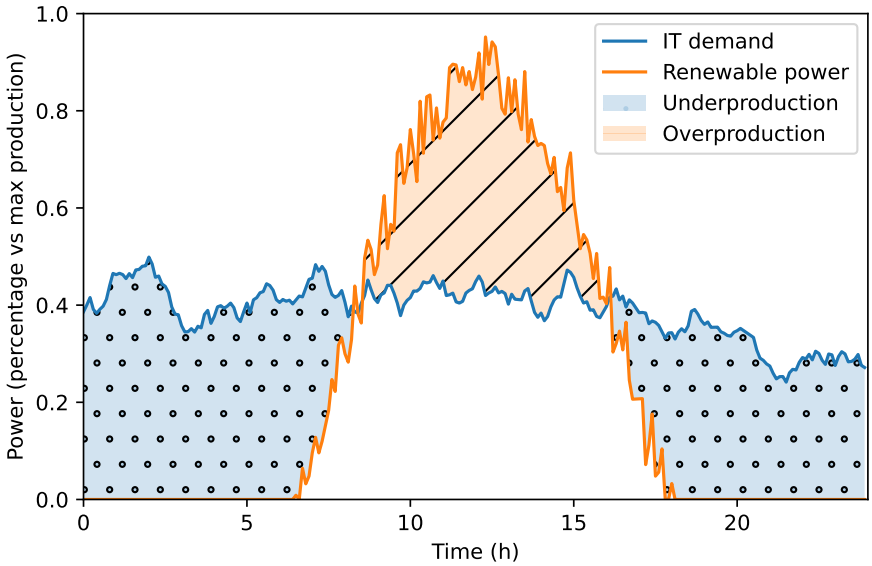


Figure IV.2: **Underproduction and overproduction.** Underproduction occurs when renewable power does not meet IT demand. Overproduction is when IT demand is lower than the renewable power.

We compute on each experiment the following metrics:

- **energy total**: the energy consumed by the data center overall;
- **energy red / yellow**: the energy consumed during red / yellow state windows;
- **overproduction**: the excess renewable energy produced in phases of overproduction, that was not consumed by the data center (orange in Figure IV.2). It is typically sold to the grid or stored in batteries.
- **underproduction** (a.k.a "brown energy"): the excess of energy consumed in phase of underproduction, that could not be produced by the renewable sources (blue in Figure IV.2). It has to be bought from the grid or drawn from batteries.

2.3.2 Effort-related metrics

In addition to energy, we want to report the effort made by the users when they adopt the various behaviors. The effort is a subjective quantity, that will not be experienced the same way by different users. All the same, we will report for each experiment:

- the number of jobs submitted without modification ("#unmodified");

- the number of jobs that were reconfigured, degraded, renounced and delayed by a See You Later ("#renounced", "#degraded", etc.).

Additionally, we aggregate all the above in a metric of *weighted effort*, where we associate to each behavior a weight, supposed to represent the inconvenience for the user. To do so, we follow the ranking of behaviors according to their "acceptability", that we proposed and justified in the previous chapter (Table III.2). We choose for the weights arbitrary values, reported in Table IV.2.

Table IV.2: **Weights for metric *weighted effort*** (without unit).

	Reconfig	See You Later	Degrad	Renounce
weight	25	50	75	100

The metric *weighted effort* is calculated as follows: with N the total number of job, n_b the number of jobs affected by behavior b and w_b the weight associated to behavior b (Table IV.2):

$$\text{weighted effort} = \sum_{\substack{b \in \{\text{Rigid,Reconfig,} \\ \text{See You,Degrad,Renounce}\}}} \frac{n_b \times w_b}{N} \tag{IV.1}$$

For example: if a job is delayed one time by a See You Later and then degraded, the effort for this job will be counted as $125/N$. Note that this is more than a single Renounce, which would be counted as $100/N$. In fact, we assume that it is less cumbersome for a user to simply renounce a job than to connect to the platform a first time, realize that the state is red, decide to come back later, realize that the state is still red and then decide to degrad.

To give an idea, summed over all jobs of a simulation, a *weighted effort* of 0 corresponds to submitting all the jobs without modification (100% Rigid) and a *weighted effort* of 100 corresponds to renouncing all jobs (100% renounce).

3 Results

The data presented in this section are obtained following the experimental campaign. They are based on Batsim scheduling and energy outputs in addition to custom user behavior logs, on which we computed the metrics described before.

3.1 State Window distribution

First, to get an understanding of the distribution of green, yellow and red state windows in the experiments, Figure IV.3 shows a typical week (top graph). The window distribution follows a day-night cycle with green state during the day, red state during the night and yellow state in-between. This reflects a common pattern for photovoltaic production. The sixth day is yellow due to the lack of sun. It is also possible to have one or more days that are completely red (no renewable production). We can see that the length of the yellow windows is small compared to green and red.

On the full experiment of nearly 6 months duration, the distribution of state window is given, aggregated by hour of the day, in the bottom graph of Figure IV.3. The day/night pattern is confirmed as green states are always starting after 6:00 and stopping before

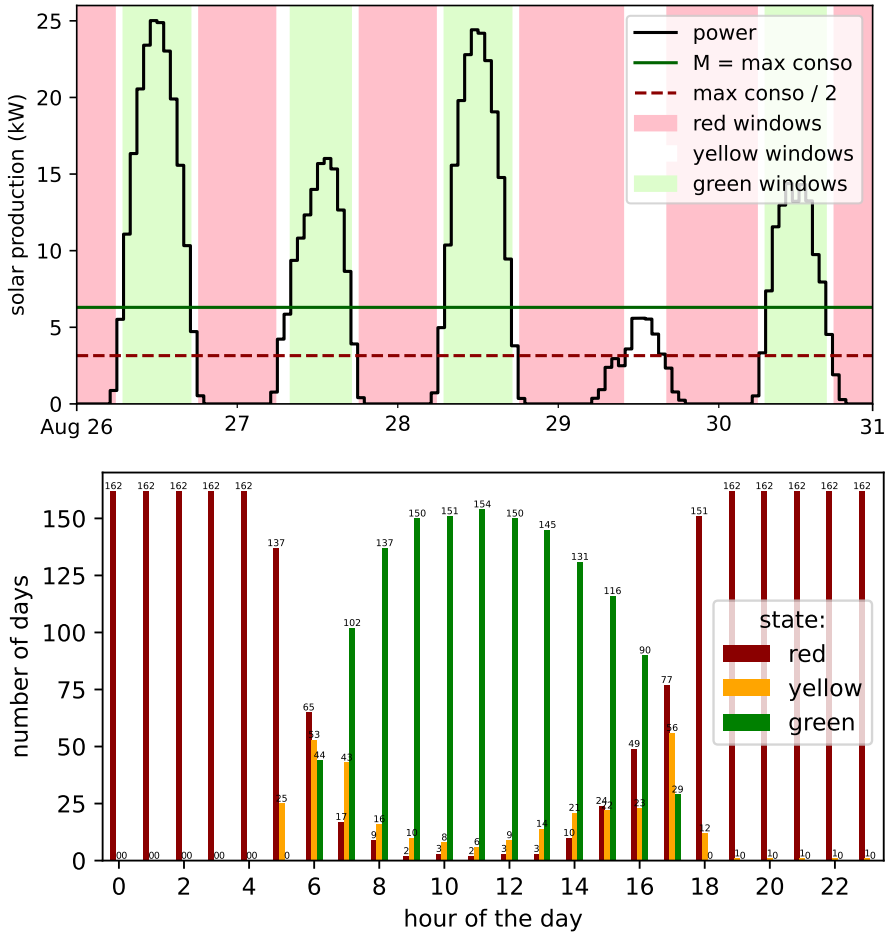


Figure IV.3: **Window state distribution in the inputs.**
Top graph: on a typical week. Bottom graph: over the 162 days of the input trace, in function of the time of the day. Top graph: data from 5 days of solar irradiation in Toulouse, August 26-30, 2019, one-hour time step. The two thresholds defining the windows are displayed as horizontal lines.

18:00, whereas red state are rare between 7:00 and 15:00. The exact distribution would vary depending on the season and geographic location of solar panels, but the overall shape should be similar.

Red state is the most common state, accounting for 56% of the experiment duration. Green state comes in second (36% of experiment duration). Yellow is the rarest state, appearing less than 8% of the time. Also, yellow states typically last at most one hour (85%), and a yellow state last 6 consecutive hours at maximum. Concerning red states, they typically last between 11 and 16 hours (80%). 10% of red state lasts less than 11 hours. Only once did a red state last up to 23 hours.

3.2 Energy consumption

The results concerning energy and user behavior are presented in Tables IV.3 and IV.4. For each metrics, the mean and standard deviation σ are computed on the $n = 30$ replicates of each experiment. The accuracy displayed in the tables corresponds to $3\frac{\sigma}{\sqrt{n}}$, or a confidence interval of 99.7%, assuming that the outputs are normally distributed.

Table IV.3 focuses on the energy-related metrics.

Table IV.3: **Mean energy metrics** in kWh, confidence interval 99.7%

user effort	y?*	energy red	energy yellow	energy total	overproduction	underproduction
full rigid ($\alpha = 0$)		8458	1145	15143	16532	8098
low ($\alpha = 0.25$)	no	8194 ± 4	1124 ± 1	14816 ± 5	16609 ± 3	7843 ± 4
low ($\alpha = 0.25$)	yes	8165 ± 6	1120 ± 2	14779 ± 8	16616 ± 4	7813 ± 6
medium ($\alpha = 0.5$)	no	7882 ± 7	1099 ± 2	14429 ± 9	16695 ± 4	7539 ± 7
medium ($\alpha = 0.5$)	yes	7827 ± 7	1089 ± 2	14361 ± 8	16705 ± 4	7480 ± 6
big ($\alpha = 0.75$)	no	7503 ± 7	1073 ± 2	13969 ± 10	16794 ± 4	7174 ± 7
big ($\alpha = 0.75$)	yes	7422 ± 8	1053 ± 2	13862 ± 11	16815 ± 6	7087 ± 8
max ($\alpha = 1$)	no	7019 ± 8	1047 ± 2	13400 ± 11	16910 ± 5	6717 ± 7
max ($\alpha = 1$)	yes	6915 ± 8	1020 ± 2	13253 ± 11	16943 ± 5	6602 ± 7
full renounce red		5620	944	11683	17308	5481

*: with yellow windows enabled?

Energy savings We see that by adopting sufficiency behaviors, users were successful to cut the energy consumption during critical periods: the energy consumed during red and yellow windows decreases with the effort α. The relationship is linear: a Pearson correlation analysis between α and energy red / yellow gives coefficients of -0.992 and -0.965, respectively.

This is also the case with the underproduction, which is the quantity that we seek to minimize. Here again, the relationship is linear (Pearson coefficient -0.991).

We see that the variability of our results is very small: the confidence intervals are close to the mean. This is due to the size of our workload (almost 700000 jobs) and the 30 replicates for each experiment, which greatly limit the effect of randomness.

Influence of yellow states Concerning the yellow states, their presence improves the energy-related metrics. In every behavior scenario, the version with yellow state yields better results for every energy-related metrics than the version without yellow states.

Remarks Firstly, it is important to note that even if *full renounce red* is the best scenario in terms of energy, it does not lead to zero energy consumption in red windows. This is due to the inertia in the system, well-discussed in the previous chapter (see Section III.4). *Full renounce red* removes all the fluid mass, but the residual mass due to jobs submitted before remains.

Secondly, we note that overproduction of energy is more than twice as big as underproduction. We have more excess than shortfall in production. More interestingly, overproduction is bigger than the total energy consumed. This reflects the fact that the electrical infrastructure was sized for the whole year, taking into account the efficiency of batteries and fuel cells. The overproduction in summer is meant to compensate the shortfalls in winter. Here, the experiments only cover summer and fall.

Lastly, we see that overproduction increases with α. This is an unwanted effect, since excess renewable power production will then have to be stored or sold to the electricity grid. It may also appear counter-intuitive because periods of overproduction are also green states. But it is in fact again a consequence of the inertia in the system. Degrad and Renounce tend to decrease the fluid mass, which, as we saw before, partly continues after the window (see illustration in Figure III.10). As a result, energy consumption after red windows, which is most of the time overproduction, is decreased by Degrad and Renounce. See You Later and Reconfig counter-balance this effect by moving load from underproduction to overproduction periods. Overall, we note that overproduction

Table IV.4: **Mean user effort metrics** in percentage of the total number of jobs in the workload (except the last column, without unit), confidence interval 99.7%.
Note that the sum of columns 3 to 7 is not equal to 100%, because a job that was delayed by a See You Later will be counted both as a delayed job and as its final behavior, which can be Rigid, Degrad, Reconfig or Renounce.

user effort	y?*	#unmodified	#renounced	#degraded	#reconf.	#delayed	weighted effort
full rigid ($\alpha = 0$)		100	0	0	0	0	0
low ($\alpha = 0.25$)	no	90.2 ±.02	3.4 ±.02	3.4 ±.02	0.4 ±.01	3.4 ±.02	7.8 ±.02
low ($\alpha = 0.25$)	yes	89.1 ±.02	3.4 ±.01	4.3 ±.02	0.6 ±.01	3.4 ±.02	8.5 ±.02
medium ($\alpha = 0.5$)	no	80.3 ±.03	7.4 ±.02	7.4 ±.02	0.9 ±.01	7.4 ±.02	17.0 ±.03
medium ($\alpha = 0.5$)	yes	78.2 ±.02	7.5 ±.02	9.3 ±.02	1.2 ±.01	7.5 ±.02	18.5 ±.02
big ($\alpha = 0.75$)	no	70.5 ±.03	12.3 ±.02	12.3 ±.02	1.4 ±.01	12.3 ±.03	28.0 ±.03
big ($\alpha = 0.75$)	yes	67.3 ±.03	12.3 ±.02	15.3 ±.03	2.0 ±.01	12.3 ±.04	30.3 ±.03
max ($\alpha = 1$)	no	60.7 ±.00	18.2 ±.03	18.2 ±.03	2.1 ±.01	18.2 ±.05	41.5 ±.03
max ($\alpha = 1$)	yes	56.4 ±.01	18.2 ±.03	22.2 ±.03	2.7 ±.01	18.2 ±.04	44.6 ±.02
full renounce red		60.7	39.3	0	0	0	39.3

*: with yellow windows enabled?

increases less than underproduction decreases, which is a reassuring result.

3.3 User effort

Table IV.4 focuses on user-effort-related metrics.

Increasing effort Similarly to energy-related metrics, we observe a strong correlation between the effort probability α and the user effort metrics. This is directly due to the definition of α in our model, which corresponds to the proportion of jobs that will be modified in red and yellow states. The metric *weighted effort*, as a linear combination of the others, is no exception. The Pearson correlation coefficient between α and *weighted effort* is 0.993.

Once again, the results feature a very small variability thanks to the size of our data.

We notice that Reconfig is the least used behavior, with only 2.7% reconfigured jobs at most. This is due to the large proportion (77%) of one-core jobs in our log, for which the reconfiguration is not available (see decision chart in Figure IV.1).

Among all the user effort scenarios, *full renounce red* is the one that requires the most effort, if we look at the number of renounced jobs. However, the scenario *max effort* with yellow states has a higher number of modified jobs, because it leads to the modification of all jobs in the red *and* yellow phases. With the weights of Table IV.2, *max effort* scenarios with and without yellow states result in a bigger weighted effort than *full renounce red*.

Influence of yellow states Overall, taking into account yellow states increases user effort. This follows directly from their definition: for the same effort scenario α, yellow states come in addition to the existing red states. As a result, the presence of yellow states increases the number of degraded and reconfigured jobs, and decreases the number of unmodified jobs. As behaviors Renounce and See You Later are not available in yellow states, the number of renounced jobs and delayed jobs is not modified.

4 Discussion

In this section, we make further analyses on the results, allowing us to answer the research questions. Then, we discuss the relevance of the weighted effort metric.

4.1 Trade-off between energy and effort

We plot in Figure IV.4 the energy gains in function of user effort. Both graphs use the energy metric underproduction, but Figure IV.4a expresses effort as the number of modified jobs, whereas Figure IV.4b uses the weighted effort metric. In both cases, we can see that user effort and gains in underproduction counter-balance each other, and the relationships between these two quantities are linear.

In the previous section, we observed that energy consumption in red and yellow phases was linearly correlated to the effort parameter α. This followed almost directly from the parameter's definition. In the present case, linearity between underproduction gains and user effort do not directly derive from the definition of α, but demonstrate that our 3-state energy approach was conclusive to transform efforts in red/yellow phases to brown energy savings.

As a result, we can answer our first research question:

(1) The sufficiency behaviors adopted by the users allow to reduce the underproduction, i.e., the energy consumption that could not be matched by renewable production. The energy savings are linear with the size of the effort, with a maximum effort giving an energy saving of -18.4% compared to no effort.
There is no threshold after which the effort provided does not result in a fair amount of energy savings. To obtain a balanced level between user effort and energy savings, one has to set either a maximum effort acceptable or a minimum energy saving wanted.

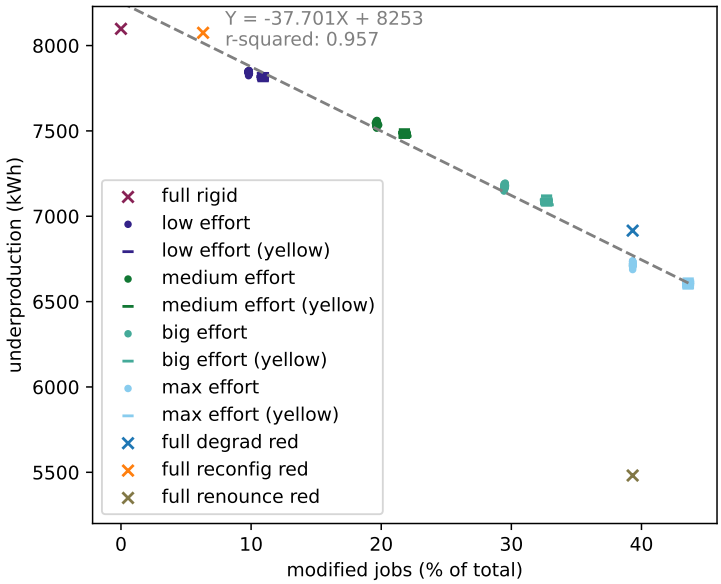
4.2 Relevance of yellow states

From Figure IV.4, we observe that adding yellow windows helps further reducing the underproduction, while also increasing the effort. It is not surprising, since scenarios with yellow windows are scenarios with red windows to which extra phases of user effort are added. A higher number of jobs gets modified, lowering their energy consumption (Reconfig and Degrad), hopefully anticipating for red phases to come. The relevant question to ask is: have the yellow states made it possible to reduce underproduction at a lower marginal cost in effort?

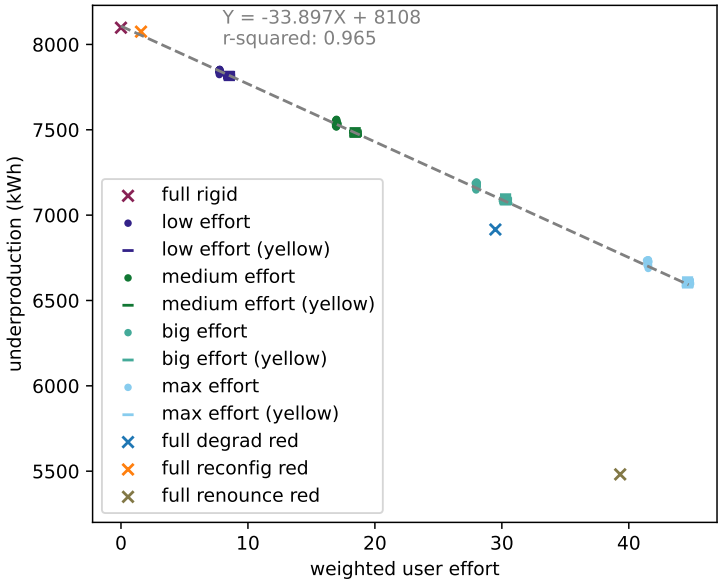
To answer this question, we plotted the gains in underproduction *per unit of effort* in Figure IV.5. What we see then, is that experiments with yellow windows yield lower gains per modified job (IV.5a) than their red-only counterparts. In other words: contrary to our expectations, yellow windows do not succeed to bring extra gains at a lower cost in effort, for this effort metric. However, if we take the weighted effort metric (IV.5b), we see no more difference between scenarios with and without yellow windows. The reason is probably that the behaviors adopted during yellow windows have a lower weight in the weighted effort metric.

We can now answer our second research question:

(2) On average, adding yellow states helps further reducing energy consumption in all the metrics considered. But yellow states can be considered as an additional effort,

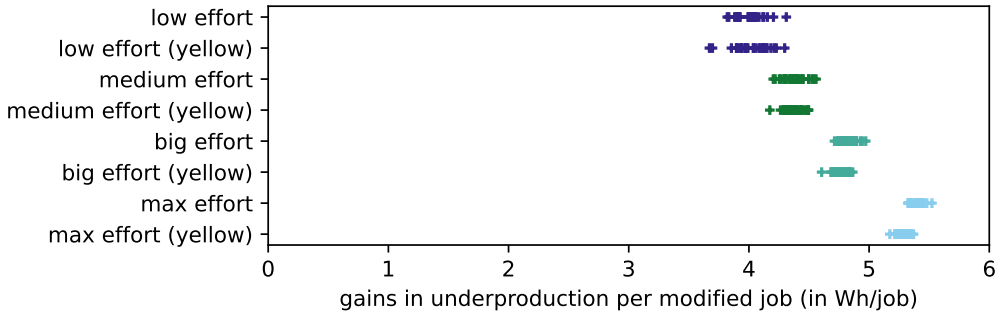


(a) effort expressed by number of modified jobs (Table IV.3, third column)

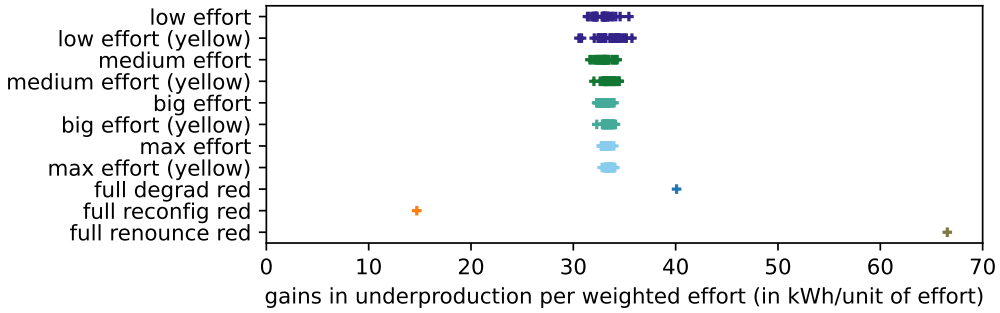


(b) effort expressed by weighted effort (Table IV.4, last column)

Figure IV.4: **Underproduction** (Table IV.3, last column) **in function of user effort.** Each point represents an experiment. Scenarios *full degrad red*, *full reconfig red* and *full renounce red* are given for comparison, and are *not* included in the linear regressions. *Full renounce red* gives a higher bound on energy saving achievable through user efforts (-32.3% compared to "full rigid").



(a) user effort expressed in number of jobs modified



(b) user effort expressed with the weighted effort metric

Figure IV.5: **Ratio between gains in underproduction** (compared to "full rigid") **and user effort.** A high ratio indicates good gains compared to the effort made by the users. "Full rigid" is not displayed as its ratio is undefined (0 modified jobs, 0 weighted effort). The other "full" scenarios are given for comparison.

and they result in additional savings of the same scale, if not slightly smaller, as other efforts made in red states.

4.3 User incentives and weighted effort metric

Interestingly, we note in Figure IV.5a that the marginal gains increase with α (low to max effort). This is an indication that "the more people who make an effort, the greater the impact of a user's additional effort". We will have to see if this observation is confirmed by other data (workload, scheduler) of if it is specific to our inputs.

In any case, since brown energy savings and effort are correlated, it is important to motivate users to adopt the sufficiency behaviors during the critical periods, for example through a financial compensation for the effort made. To this end, the weighted effort metric can prove very useful. In fact, we observe in Figure IV.5b that this metric is almost perfectly proportional to the gains. The coefficient of proportionality is $\gamma = 33.9$ kWh/unit of effort (slope of the line in Figure IV.4b). Renouncing to a job corresponds to a weighted effort of w_{renounce}/N (Equation IV.1). By multiplying by the coefficient of proportionality, it derives that renouncing to a job allows reducing the underproduction by $\gamma * w_{\text{renounce}}/N = 33.9 * 100/69306 = 0.00489$ kWh on average. This is consistent with the average length and size of jobs in the workload, and the power consumption of servers. We could imagine giving to the user a reward equivalent to the cost of the electricity saved that way.

5 Limitations

Our study has a number of limitations that are discussed in this section.

5.1 Limits of the model

First of all, the data center model being the same as in the previous chapter, it suffers from the same limitations (see III.5). Same goes with the user model:

User hypothesis for the modeling We assumed that users have some technical knowledge and control on the jobs they send. Otherwise, they could not apply the sufficiency behaviors. Depending on the type of data center, this hypothesis might not hold. For example, if a data center runs mainly automated jobs (e.g., automatic testing, critical services), there is no room for users to decide on the way to run the jobs.

Behaviors Degrad and Reconfig We also assumed that the jobs can always be degraded. However, this is only true for some jobs, e.g., convergence-based algorithms where the convergence criterion can be tuned or video transcoding where the quality can be lowered. Moreover, users might have to make timely changes in the application to modify the number of cores required (e.g., changing some parts of the code or configuration files), which makes Degrad and Reconfig not realistic for real-time response. Improvement of the model is needed to take this into account.

Behavior See You Later More specific to this study, the behavior See You Later can create some abnormalities in the submission behavior. First, job submission order is not preserved by this behavior. For example, if a user submits $job1$ at 13:30 and $job2$ at 14:00 in the original workload, a See You Later on $job1$ and a Rigid on $job2$ will lead to a change in the order of the two jobs. Moreover, it could lead to shifting submission time to a time of the day when the user is usually not connected to the platform. For example, we observe users submitting late at midnight when they typically stop submitting after 18:00 in the original workload.

5.2 Limits of the experiments

We see a number of shortcomings in our approach to answer the research questions:

Three-state energy model If the proposed feedback mechanism was conclusive to reduce energy consumption in underproduction periods, we see a limitation in the fact that it was based *only* on level of renewable production. Indeed, if production is low but IT demand is similarly low, there is no point in making an effort. It would be interesting to see if the results are better if the thresholds for energy states are defined on the real-time difference between production and consumption, rather than on production only.
Alternatively, we could have tried other thresholds to define the states, instead of the fixed 50%-100% proposed here, to see if the conclusions remain the same.
This leaves room for improvement for future works on the topic.

Equiprobability of every non-rigid behavior In the experimental campaign, the behaviors are drawn at random, assuming that each non-rigid behavior has the same probability (for a fixed window state and number of cores, see Figure IV.1). This has a number of drawbacks:

- it might not reflect the real popularity of each behavior,
- different users might have different preferences, and
- the choice of behavior is likely influenced by the nature of the job (criticality, size, difficulty to reconfigure, etc.).

However, there is no data available on the popularity of each behavior, and we have no information on the nature of the jobs in the input workload (i.e., which ones are critical or difficult to reconfigure). Consequently, doing a randomized campaign with equiprobable behaviors seemed the most reasonable method to test the potential of our approach experimentally. New studies are needed to explore how the mix of probabilities impacts both the results and the link between effort and gains.

Limited experimental conditions The experimental campaign presented in this paper explores a rather limited set of parameters. First, current input data only include one IT workload (MetaCentrum 2) which is characteristic of a HPC infrastructure. It would be interesting to see how the approach can be adapted to other types of data centers, like cloud infrastructures. Similarly, we used only one renewable energy trace (Toulouse), in the period from June 1, 2019, to November 30, which means that two seasons (winter and spring) are not included. Besides, we only looked at solar energy as a renewable source, and one sizing for the IT and electrical infrastructure. Finally, we tested the approach with only one scheduling policy: bin-packing with greedy server shut-down, which is rather naive and not state-of-the-art.

However, the focus of this contribution is to estimate the potential of *user behaviors*, and does not aim at evaluating the exact gains in all possible configurations. We argue that the results would be similar with other sets of workload/platform/scheduler. Since our code is open-source and our experiments reproducible, it would be easy to verify it in the future.

6 Conclusion

In this chapter, we build upon the five sufficiency behaviors introduced and characterized in Chapter III. We investigate the usefulness of these behaviors in the context of renewable energy management. A three-state energy feedback mechanism is introduced, to inform users on the status of electricity production: green when production is abundant, red when production is low, and yellow in-between. When submitting to the platform, users can see the energy state and decide to take action by adopting a sufficiency behavior. Our user simulation software Batmen is extended with a new user class implementing this approach.

Thanks to a reproducible experimental campaign with real-world inputs, we show that the approach is conclusive and allows reducing brown energy consumption. Energy gains are proportional to the efforts made by users. Yellow states allow to further increase the gains in exchange for a cost in effort similar to the one in red states. They can be useful at times when more effort is not possible.

With this, we conclude our contributions related to the analysis of sufficiency behaviors in data centers with simulation. Through the making, they made us realize two fundamental problems.

On the one hand, we cruelly lack knowledge on the type of efforts that data center users would be willing to make. Therefore, our simulations can only be based on assumptions. Collecting such data would involve social science methods such as interviews or questionnaire surveys. We lay a stone in this direction with Chapter VI.

On the other hand, analyzing the results of our experiments has shown to us the limits of the workload replay model in simulations, although widely used in the community. This model consists of replaying a historical workload by following the original timestamps of submission. However, nothing ensures that users would still have submitted at these dates if the system was behaving differently. This is particularly problematic in our case, where we simulate modifications of job characteristics at time of submission. It led for example to the burst of submission at the end of the window with behavior Delay in Chapter III, and to the non-preservation of the order of submission with behavior See You Later in this chapter. With the next chapter, we tackle this issue and provide both methodology and open-source code to achieve more realistic simulation.

Chapter V

Replay with feedback for more realistic simulations

Contents

Since the beginning of this manuscript, we have been using data center simulations to test our approaches. Simulations have the great advantage of being relatively simple to carry compared to experiments on real infrastructures. They also allow the researcher to try a large number of parameters and inputs at low cost. However, their drawback is that they rely on a set of models and assumptions. The assumptions might not always apply, and some models have not been carefully validated.

At the center of any simulation is the workload model. Sometimes, simulation works use synthetic workloads, generated from a mathematical model adapted to their use case. As explained in Chapter II, we chose in this thesis to use records from real infrastructures and replay the jobs submissions (see Chapters III and IV, with inputs from the Parallel Workload Archive). The method is simple: we simulate jobs of the same characteristics as in the recorded workload, and reproduce their arrivals at the same timestamps.

Whether the workload is replayed or generated, it is generally fully determined before the simulation, i.e., *jobs submissions are known in advance*. This is the case in recent works like Vasconcelos et al. using a synthetic workload to study distributed cloud federation [54], Wiesner et al. using a recorded workload from a production system to evaluate their renewable-aware scheduler [120] or de Nardin et al. using logs from an academic data center [121]. More generally, the vast majority of articles using simulation for performance evaluation of HPC-like systems uses a fixed-timestamp model.

While this makes the workload model simple and the results easy to compare, we argue that this is too strong a hypothesis leading to unreliable results. In fact, in reality, *users adapt their submission behavior to the feedback they get from the infrastructure*. For

example, imagine a real infrastructure that suddenly becomes twice slower due to a breakdown. The jobs will start accumulating in the queues and the users, seeing that their previous submissions are still pending, will slow down their rate of submission. On the contrary, if the infrastructure gets faster due to a more efficient scheduling or the addition of new servers, the users will tend to submit more and bigger jobs, a phenomenon known as "rebound effect".

The pattern of job arrival is in fact tightly linked to the specific infrastructure in which it is observed, and is the fruit of interaction between the users, the scheduling algorithm and the underlying hardware. Regrettably, this interaction is rarely accounted for in the literature.

In this chapter, we tackle this problem, making the best of our software suite to model user reaction to system performance, thanks to a method called "replay with feedback". We start by giving some background on the topic and explain the general idea in Section 1. The method is composed of two steps that are detailed in Section 2 and 3. After this, we design an experimental campaign comparing traditional replay and replay with feedback, that we describe in Section 4 and whose results are given in Section 5. To explain the results, we introduce three new metrics adapted to the dynamic adaptation of workload. In Section 6, we discuss several aspects of the results and our model. We close the chapter on the limitations of our approach (Section 7) and a conclusion section. These contributions have been published in the journal FGCS [J1].

1 Background and general idea

The problem described in introduction of this chapter is not new. It was in fact identified by Dror Feitelson and his PhD students in the years 2010 [122, 123]. Instead of replaying jobs with fixed submission times, they recommend doing *closed-loop* simulations [124], where the submitted workload adapts to the simulated performance of the system.

In particular, Feitelson introduced with Netanel Zakay the concept of "replay with feedback", which is the focus of this chapter. The main idea is to wait for the previous jobs to finish before submitting the new ones, instead of blindly following the submission times from the recorded trace. We will give a more formal definition in Section 3.

According to Zakay and Feitelson, "replay with feedback" follows a two-step approach:

(i) extracting the relevant patterns of user submission behavior from the recorded workload [125], and

(ii) using this information during the simulation, to replay users reaction to feedback [122].

Step (i) is explained in Section 2, and step (ii) in Section 3. Regrettably, we could find no code available to reproduce their results, nor detail on the simulation software used. This is why we chose to reimplement them with Batmen. At the end of this chapter, in Section 6.4, we come back to Zakay and Feitelson's method, and point out the difference with ours.

To the best of our knowledge, replay with feedback is only used once more in the literature, by Klusáček et al. They implemented Zakay and Feitelson's model inside the open source simulator Alea [126]. The feature is called "dynamic workload adaptation" and was notably used to test different schedulers before their deployment in a production system [114]. However, the model in their work is only used as a tool, and they provide no evaluation nor in-depth discussion on its effect on the simulation outcome.

2 Retrieving information from recorded workloads

The first step of "replay with feedback" is to process the recorded workload trace to extract the relevant patterns of user submission behavior that will be useful during the simulation. This is done through partitioning the trace into sessions (2.1), and infer their dependence relationship (2.2).

2.1 Session partitioning

We remind the notations from Chapter II (Definitions 4 and 5): a **recorded trace** is a list of **recorded jobs**, each containing at least the information (d_i, r_i, a_i, f_i) (duration, number of resources, arrival date, finish date). We suppose that for each recorded job we know the ID of the user that submitted it. Each j_i also has a walltime w_i, i.e., an upper bound on execution time given by the user after which the job gets killed. Walltimes can be used by the scheduler to take its decisions. To simplify the notations in the following, we split the recorded trace and renumber it per user:

Definition 12. *The **user trace** $\mathcal{T}_u$ is the list $(j_1, ..., j_N)$ of recorded jobs submitted by user u, ordered by submission time.*

In their work, Zakay and Feitelson argue that meaningful components of a recorded trace are *user sessions*, which they define conceptually as "periods of continuous work by a user" [125]. In other words, users have sessions of work in which they interact with the infrastructure — mainly by submitting new jobs — and periods of absence. With our notation:

Definition 13. *A **user session** in the user trace $\mathcal{T}_u$ is a list $(j_n, j_{n+1}, ..., j_m) \subset \mathcal{T}_u$ of consecutive recorded jobs, with $(n, m) \in \mathbb{N}^2$ and $n \leq m$.*

Unfortunately, we rarely have metadata on user activity associated to the recorded trace to help us detecting the session boundaries. Therefore, we will infer them from the information contained in the recorded trace. We call this operation *session partitioning*:

Definition 14. *A **session partition** of the user trace $\mathcal{T}_u$ is a set $\{s_1, ..., s_M\}$ of user sessions (with $M > 0$ the number of sessions), such that*

$$s_1 \cup ... \cup s_M = \mathcal{T}_u \text{ and } \forall 1 \leq i < k \leq M, s_i \cap s_k = \varnothing$$

There are many possible ways to do this partitioning based on the information at hand in the recorded trace. The simplest is to consider that each job constitutes its own session. Another way is to consider that several jobs are in the same session if they arrive close together, which means defining the sessions with thresholds on inter-arrival time between jobs. One could also use thresholds on think time between jobs (time that elapses between the termination of one job and the submission of another).

We refer to the original paper of Zakay and Feitelson [125] for the proposition and comparison of three such methods. I also developed the open-source Python script `swf-2userSessions`[1] performing the session partitioning of a recorded trace in the Standard Workload Format, in which the three methods are implemented. This script is in $O(n)$ on the number of lines in the input file, and has execution times in the order of the minute on a laptop computer with classical inputs (< 10 million lines).

[1] available at `gitlab.irit.fr/sepia-pub/mael/swf2userSessions`. The specific version tagged `replay_feedback2023` is used in this chapter.

2.2 Session graph

Relation *depends on* Once we have partitioned a user trace $\mathcal{T}_u$ into sessions, we are interested, for the replay, in the relationship between them. We define between sessions the partial order *depends on*:

Definition 15. *For two sessions $s = (j_n, ..., j_m) \subset \mathcal{T}_u$ and $s' = (j_{n'}, ..., j_{m'}) \subset \mathcal{T}_u$, we say that s' **depends on** s if all the recorded jobs of s finished their execution before the first job of s' was submitted, i.e., if $max_{n \leq i \leq m}(f_i) \leq a_{n'}$*

Think time between sessions From this definition follows our definition of *think time* between sessions. Conceptually, it corresponds to the time that a user had to think between the termination of all the jobs in a session and the submission of the first job of another:

Definition 16. *For two sessions $s = (j_n, ..., j_m) \subset \mathcal{T}_u$ and $s' = (j_{n'}, ..., j_{m'}) \subset \mathcal{T}_u$, we call **think time** the quantity $a_{n'} - max_{n \leq i \leq m}(f_i)$.*

Note that, by definition of the relation *depends on*, the think time between two depending sessions is always ≥ 0.

Session graph As a result, the set of sessions for a user forms a *weighted directed acyclic graph*, where the nodes are the sessions and the edges represent the relation *depends on*, weighted by the corresponding think time. See Figure V.1 for illustration.

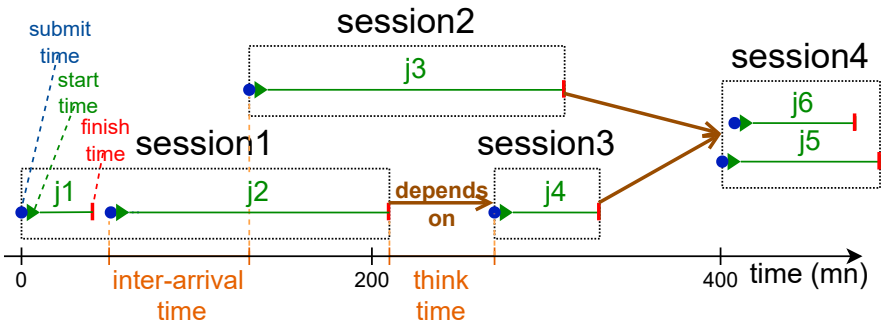


Figure V.1: **Illustration of a session graph with six jobs and four sessions.** Here, an inter-arrival time greater than 60 minutes between two jobs delimits a new session: j1 and j2 are in the same session since $a_2 - a_1 < 60$ mn, while j3 is in a different session since $a_3 - a_2 \geq 60$ mn.

In this representation, some sessions have no predecessor: sometimes only the first session and sometimes more (like session1 and session2 in the illustration). We complete the graph by adding a fictive session at the root of the graph and making all the sessions that do not have a predecessor dependent on it. For each edge added that way, we choose the recorded starting time of the session as *think time*, i.e., the submission time of the first recorded job of this session. The resulting session graph contains all the information needed for the replay.

3 Replay with feedback

In this section, we provide a definition of replay with feedback, then describe our method of replay using the session graphs previously described.

3.1 Feedback and rigid: two paradigms of replay

Replay with feedback is a new paradigm of using recorded workload data in simulations:

Definition 17. ***Replay with feedback*** *is a way of using a recorded workload in simulations to mimic the platform activity while accounting for user reactions to simulated system performance.*

In practice, users adapt to feedback in many ways. They change their dates of submission, submit bigger or smaller jobs, modify their software to fit the infrastructure or even leave the infrastructure to submit somewhere else. Taking into account all these behaviors in the replay is a very challenging task, and can potentially modify the workload significantly. For this reason, *we consider in this work only one type of user response, namely changes in submission times.*

To make a distinction between the *recorded* jobs and their simulated copy, we call the latter *replay* jobs and denote them $\hat{j}_i = (\hat{d}_i, \hat{r}_i, \hat{a}_i, \hat{f}_i, \hat{w}_i)$. The type of replay with feedback performed here preserves the jobs characteristics: computing load, number of resources and walltime. With the notation:

$$\forall 1 \leq i \leq n, \begin{cases} \hat{d}_i * \hat{r}_i * \hat{P} &= d_i * r_i * P \\ \hat{r}_i &= r_i \\ \hat{w}_i &= w_i \end{cases}$$

where P (resp. $\hat{P}$) is the performance of the server in the original (resp. simulated) infrastructure, in floating-point operations per second.

The traditional way of replaying jobs in simulations would also preserve the submission times, i.e., $\hat{a}_i = a_i$. We will denote it "rigid replay":

Definition 18. ***Rigid replay*** *simulates the arrival of jobs with the same characteristics and same submission time as in the recorded workload.*

3.2 Replay based on think times

The main idea behind our replay method is that it preserves the *think time between sessions* rather than the exact submission times of the jobs, thus reacting to the feedback provided by the (simulated) infrastructure. For example, if a job inside a session takes longer to finish in the simulation compared to the recorded trace, the following sessions in the session graph will be delayed accordingly. However, jobs *within* a session are neither delayed nor brought forward in reaction to feedback with our algorithm[2]. Consequently, all the information needed for the replay with feedback are embedded in the session graph of each user.

We say that a session **starts** when its first (replay) job is submitted. A session **finishes** when the last of its jobs finishes its execution. Without loss of information, we represent the submission times $\hat{a}_i$ of the replay jobs relatively to the start time of their session, i.e.,

$$\forall s = (j_n, ..., j_m) \subset \mathcal{T}_u, \forall i \in \{n, n+1, ..., m\}, \hat{a}_i = a_i - a_n$$

[2] this is different from Zakay and Feitelson, who introduce the notion of 'batches' within a session, which are groups of overlapping jobs. The relation *depends on* and the shifts during replay are defined for the batches. See Section 6.4.

Before going on with the description of the replay method, we need to introduce two additional definitions:

Definition 19. *A session $\hat{s} = (\hat{j_n}, ..., \hat{j_m})$ is **active**, at time t, if $\hat{s}$ has started and $t_{\hat{s}} \leq t < t_{\hat{s}} + \hat{a_m}$, with $t_{\hat{s}}$ the starting time of $\hat{s}$.*

Conceptually, $\hat{s}$ is active when the user is currently submitting from it.

Definition 20. *A session $\hat{s}$ is **free**, at time t, if it has not started and all the sessions it depends on have finished, i.e.,*

$$\begin{cases} t < t_{\hat{s}} \\ \forall \hat{s'} = (\hat{j_n}, ..., \hat{j_m}), \hat{s} \text{ depends on } \hat{s'} \implies max_{n \leq i \leq m}(\hat{f_i}) \leq t \end{cases}$$

*If $\hat{s}$ has not started but at least one session it depends on has not finished, we say that $\hat{s}$ is **dependent**.*

Consequently, the usual lifecycle for a session is: dependent → free → started (active) → started (all jobs submitted) → finished.

3.3 Replay method

We can now proceed to the explanation of the replay method. The method is combined with a discrete event simulation, in charge of simulating the Resource and Job Management System (RJMS). During the simulation, the RJMS simulator reproduces the behavior of the servers in the platform and sends events when something happens (e.g., a job terminates, a server finishes switching on) or when it is time to submit a job (callback). For its part, the replay method traverses the session graph for each user, by keeping track of

1. the list $\mathcal{A}$ of active sessions, and
2. the list $\mathcal{F}$ of free sessions.

At the beginning of the simulation ($t = 0$), $\mathcal{A}$ is empty and $\mathcal{F}$ contains the successors of the fictive root session. The replay method listens to the events sent by the RJMS simulator, and comprises mainly two functions: `wake_on_feedback`, called when a job terminates, and `job_to_submit`, called when it is time to submit a job. These functions are given as pseudocode in Algorithm 5.

Note that:

1. Our replay method does *not* necessarily preserve the original submission order of jobs. For example in Figure V.1, if j2 finishes earlier in the replay, j4 might be submitted before j3.
2. If the scheduler rejects a job during the replay, for instance because it requests more resources than the total number of cores in the platform, a job termination event is immediately sent. The finished time of this job is considered equal to its submission time, i.e., $\hat{f_i} = \hat{a_i}$.

4 Experimental comparison of feedback and rigid replay

In this section, we present an experimental campaign designed to compare replay with feedback (Definition 17) and rigid replay (Definition 18).

Algorithm 5 Replay method

```
function WAKE_ON_FEEDBACK                    ▷ in reaction to the termination of a job
    for all j ∈ {jobs finished recently} do
        s ← session(j)
        if j was the last job of s to finish then
            for all s' ∈ {successors of s} do
                dependencies(s').pop(s)
                if dependencies(s') is empty then
                    F.add(s')                          ▷ definition of free session
    update next callback time
function JOBS_TO_SUBMIT                      ▷ when it is time to submit one job (or more)
    for all s ∈ F do
        if s starts now then
            A.add(s)                           ▷ definition of active session
            F.pop(s)
    for all s ∈ A do
        submit all the jobs in job_list(s) with submission_time = now
        if job_list(s) is empty then
            A.pop(s)                           ▷ definition of active session
    update next callback time
```

4.1 Experimental setup

Software environment Like before, the simulations are run with the software environment described in Chapter II, Section 2: Batsim (v4.2), SimGrid (v3.32) and Batmen (v3.0).

The replay with feedback model described in Section 3 is implemented in Batmen. The superclass `FeedbackUser` implements the functions `wake_on_feedback` and `jobs_to_submit` described in Algorithm 5, along with the function `update_date_next_sub` in charge of calculating the next timestamp at which the user might have a job to submit. It is needed to inform Batsim on when to call Batmen to check if a job submission arrives. The child class `FBUserThinkTimeOnly` implements the function `close_session`, which is called every time a session finishes. It updates its following sessions in the session graph and the list of free sessions, according to the replay model.

Workload As inputs for the simulations, we use two recorded traces retrieved from the Parallel Workload Archive:

- KTH-SP2[3]: 11-month log from a 100-node IBM SP2, and
- SDSC SP2[4]: 24-month log from a 128-node IBM SP2.

KTH and SDSC logs contain respectively 28475 and 67667 jobs, for 214 (resp. 428) users. The submission log for each user was converted to session graphs, as explained in Section 2. We made the *session partitioning* based on a threshold on inter-arrival time. We tried two values for this threshold: 0 minutes ('arrival 0', in short '**a0**') and 60 minutes

[3] file `KTH-SP2-1996-2.1-cln.swf`, available at https://www.cs.huji.ac.il/labs/parallel/workload/l_kth_sp2/index.html

[4] file `SDSC-SP2-1998-4.swf`, available at https://www.cs.huji.ac.il/labs/parallel/workload/l_sdsc_sp2/index.html

('**a60**'). a0 gives sessions of only one job. Doing a replay with this delimitation is equivalent to *preserve the think time between jobs only*. We chose the other threshold of 60 minutes because it is the value used in the original paper [125]. The influence of this parameter will be discussed in Section 6.2.

IT platform We created two platform files adapted to the traces, with 100 (resp. 128) monocore homogeneous servers.

Scheduler From the information we could find online [127], IBM SP2 systems seem to be using some version of EASY-backfilling algorithm for scheduling (see Chapter II, Algorithm 2). For this reason and unless specified otherwise, we use such a scheduler in our experiments, called "EASY" in the remaining of this chapter. We will also try with a FCFS scheduler (Algorithm 1).

4.2 Experimental campaign

We design an experimental campaign to compare rigid and feedback replay methods. We are interested to see the effect of feedback when the system performances change, therefore investigating this question:

> Is replay with feedback satisfying to simulate a change in the infrastructure or scheduler?

A change in the infrastructure can be a change in the number of servers, server performance, interconnection, bandwidth etc. Whichever the change, the outcome will be that jobs execute faster or slower. Since the only feedback that matters to users in our replay method is the finish time of their jobs, we consider sufficient in this study to focus on two types of infrastructure change: number of servers and server performance. Consequently, jobs in the simulation are represented as compute-only (`parallel_homogeneous` in Batsim), without communication.

We run the workload several times, varying the scheduler and hardware infrastructure (6 different cases):

- **easy**: the baseline experiment, with EASY scheduler and a simulated platform representing the original infrastructure;
- **perf*2, perf/2**: multiplying or dividing by two the *performance* of the servers, in terms of floating-point operations per second, i.e., the jobs are executed twice (resp. half) as fast;
- **infra*2, infra/2**: multiplying or dividing by two the *number* of servers, e.g., for KTH log we tried with 200 (resp. 50) servers;
- **FCFS**: changing the scheduling algorithm to First Come First Served (Algorithm 1).

Each instance is run *with* the feedback model (delimitations methods **a0** and **a60**), and *without* (rigid). In total, $6 * 3 = 18$ simulations are run for each log.

Reproducibility All the experimental details and material to reproduce the graphs presented in this chapter are provided in forms of two notebooks[5]. Running the two notebooks on a recent laptop (Intel i5 11th gen) takes less than one hour, including downloading and processing the inputs, running the simulations and plotting the graphs.

5 Results

The results of the experimental campaign consist in complete records $(\hat{a}_i, \hat{f}_i - d_i, \hat{f}_i)$ of the timestamps of submission, start and finish time for each replay job $\hat{j}_i$. From these records, we compute several metrics. First, we give the makespan and waiting times (5.1), two common scheduling metrics. Then, we plot the distribution of submission times (5.2). Finally, we define and provide the results on three new metrics, better suited for feedback replay: mean lateness (5.3), relative lateness (5.4) and additional lateness (5.5).

5.1 Common scheduling metrics

In Table V.1, we provide the results of the experimental campaign on three common scheduling metrics: makespan, mean waiting time and max waiting time. We also calculate these metrics on the original recorded trace. The definitions of makespan and waiting time are reminded below, with our notations. Other usual scheduling metrics like turnaround time or slowdown can be found in the notebooks.

Definition 21. *The **makespan** is the time that elapses between the submission of the first job and the completion of the last:*

$$makespan = max(f_i) - min(a_i) \tag{V.1}$$

Definition 22. *The **waiting time** of a job is the time that elapsed between the submission of the job and the beginning of its execution in the infrastructure.*

$$waiting\ time(j_i) = f_i - d_i - a_i \tag{V.2}$$

Impact of feedback replay The results confirm what was said in introduction: the traditional replay model is not satisfactory to simulate a change in the infrastructure or in the scheduler. In the recorded log, the waiting times were of 0.18 days on average, and 11.34 days maximum for KTH (resp. 0.26 and 62.48 for SDSC). With the feedback model and *whichever the change we make in the infrastructure*, the mean waiting times are under 0.62 day for KTH (**perf/2 a60**) and 1.61 for SDSC (**perf/2 a60**). All the max waiting times remain lower than the original max waiting times.

In the case of rigid replay however, the picture looks different. We get waiting times of up to 141 days (**perf/2 rigid KTH**), resp. 508 days (**perf/2 rigid SDSC**). The mean waiting times are also significantly higher than the original in **infra/2** and **perf/2** experiments (up to two months for **perf/2 rigid SDSC**). It is unrealistic to think that the users would have waited on average all this time if the change actually occurred in the real infrastructure. Instead, they would have slowed down their pace of submission, which our model successfully accounts for.

[5]Experiment repository: gitlab.irit.fr/sepia-pub/open-science/expe-replay-feedback (use the tag maelPhD). The outputs are directly visible in the GitLab interface.

Table V.1: **Scheduling metrics** calculated on the recorded log and for all the experiments. Makespan and waiting times are expressed in days, with 2 decimal places.
For KTH infra*2, we read mean waiting times of 0.00 day. This is because of rounding, and these values are actually between 162 and 229 seconds.

		KTH			*SDSC*		
		makespan	**waiting time**		**makespan**	**waiting time**	
exp. name	**replay**		mean	max		mean	max
recorded trace	/	332.93	0.18	11.34	736.12	0.26	62.48
EASY	rigid	332.91	0.07	4.07	731.36	0.19	5.73
	a0	366.14	0.06	5.06	808.88	0.14	5.90
	a60	366.67	0.07	6.11	789.77	0.18	5.16
FCFS	rigid	333.10	4.51	11.79	794.26	14.82	63.96
	a0	457.89	0.29	4.95	1200.10	0.58	6.26
	a60	454.41	0.47	4.47	1065.66	0.88	5.51
perf*2	rigid	332.91	0.01	1.34	731.32	0.01	1.84
	a0	332.57	0.01	1.82	730.31	0.01	1.58
	a60	332.61	0.01	1.44	729.82	0.02	1.13
perf/2	rigid	471.85	31.84	141.34	1239.37	64.62	508.38
	a0	635.97	0.46	10.70	1506.26	0.92	15.54
	a60	630.28	0.62	10.26	1492.67	1.61	14.17
infra*2	rigid	332.91	0.00	0.54	731.36	0.01	1.28
	a0	332.63	0.00	0.81	729.81	0.01	1.04
	a60	332.65	0.00	0.56	730.02	0.01	1.35
infra/2	rigid	386.70	4.15	58.87	1167.94	37.43	437.28
	a0	472.93	0.27	7.43	1452.31	0.80	14.82
	a60	472.45	0.35	7.31	1446.13	1.20	15.93

Impact of infrastructure change Since the pace of submission slows down in reaction to a slower infrastructure (**FCFS**, **perf/2** and **infra/2**) with the feedback model, it should take more time for the same workload to be fully executed. This effect is clearly visible in the results: the makespan in experiments **FCFS**, **perf/2** and **infra/2** increases significantly more with feedback replay than with rigid replay, compared to the original makespan. However, we would also expect to see the opposite effect when the infrastructure is faster (**perf*2** and **infra*2**), which is not the case here. The makespans in these experiments with rigid, a0 and a60 replay models are very similar, close to the original makespan. This is due to the presence of large think times in the session graphs, for example because of new users arriving a few days before the end of the original record. We discuss these remaining sources of rigidity in the feedback model in Section 7.4 to give hints for improvement.

Impact of scheduler Note that the results also show that the scheduler in the real infrastructure and our implementation of EASY backfilling are not exactly the same. For example, the mean and max waiting times are significantly lower with our implementation (experiment **EASY rigid**). All the same, EASY seems closer to the original scheduler than FCFS, with which the mean waiting time explode (**FCFS rigid**). Interestingly, FCFS produces a *max waiting time* close to the original in both logs, suggesting that some jobs are probably submitted in pure FCFS order in the original scheduler. A detailed description of the scheduler originally used would be necessary to understand better, which

we could not find for these logs.

In the end, the usual scheduling metrics discussed here give us useful insights, but they show their limits to fully explain the effect of changing the replay model. For example, they don't capture to what extent the submission times are shifted compared to rigid replay. To that end, we introduce in the remaining of this section a new way of making the analysis, including new metrics.

5.2 Submission time distribution

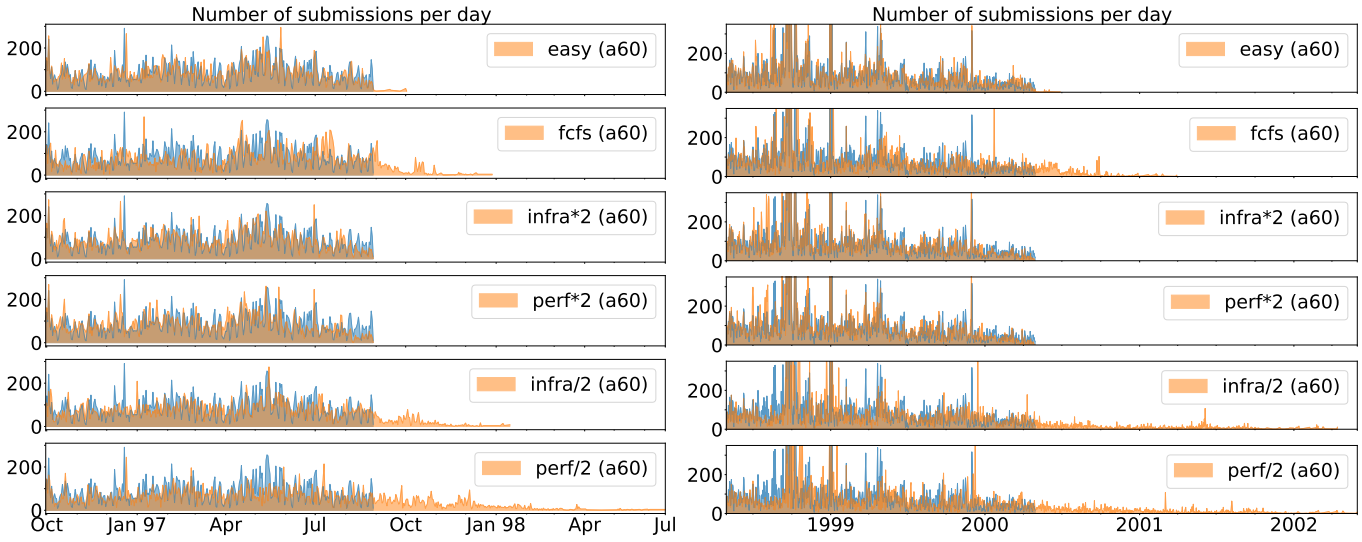
The replay with feedback model primarily impacts the *submission times* of jobs. Its effect is thus visible on the temporal distribution of job arrivals, plotted in Figure V.2 and in cumulative values in Figure V.3. The blue curve, corresponding to the rigid replay model, remains identical for all experiments: it corresponds to the original timestamps of submission in the recorded logs. With the feedback model, however, we observe that the submission distribution spreads with the specific infrastructure or scheduler used.

In experiments **perf/2**, **infra/2** and **FCFS**, we get confirmation that the simulated users submitted fewer jobs per day on average (orange curve under the blue curve in cumulated values in Figure V.3). In return, the length of the submission period has increased (horizontal span of the orange curve longer than the blue). On Figure V.2 we observe that, passed the original length of submission, the rate of submission decreases in trend. This is also visible in the cumulative graphs: the orange curve starts to slowly plateau where the blue graph ends. These are in fact end-of-simulation side effects: users finish submitting their backlog of jobs, without new jobs and users arriving. These effects are not relevant, and the analysis should instead focus on what happens in the simulations within the length of the original workload.

Experiments **perf*2** and **infra*2** need a closer look, as the effect of the infrastructure change is less visible in these cases. We can observe that the orange curve is slightly above the blue in the cumulative graphs, meaning that the rate of submission increased slightly. In return, the length of submission is not shorter, but we can notice that users submit fewer jobs per day at the very end, for example in the last month of KTH log. They are reaching the end of their pool of jobs to submit.

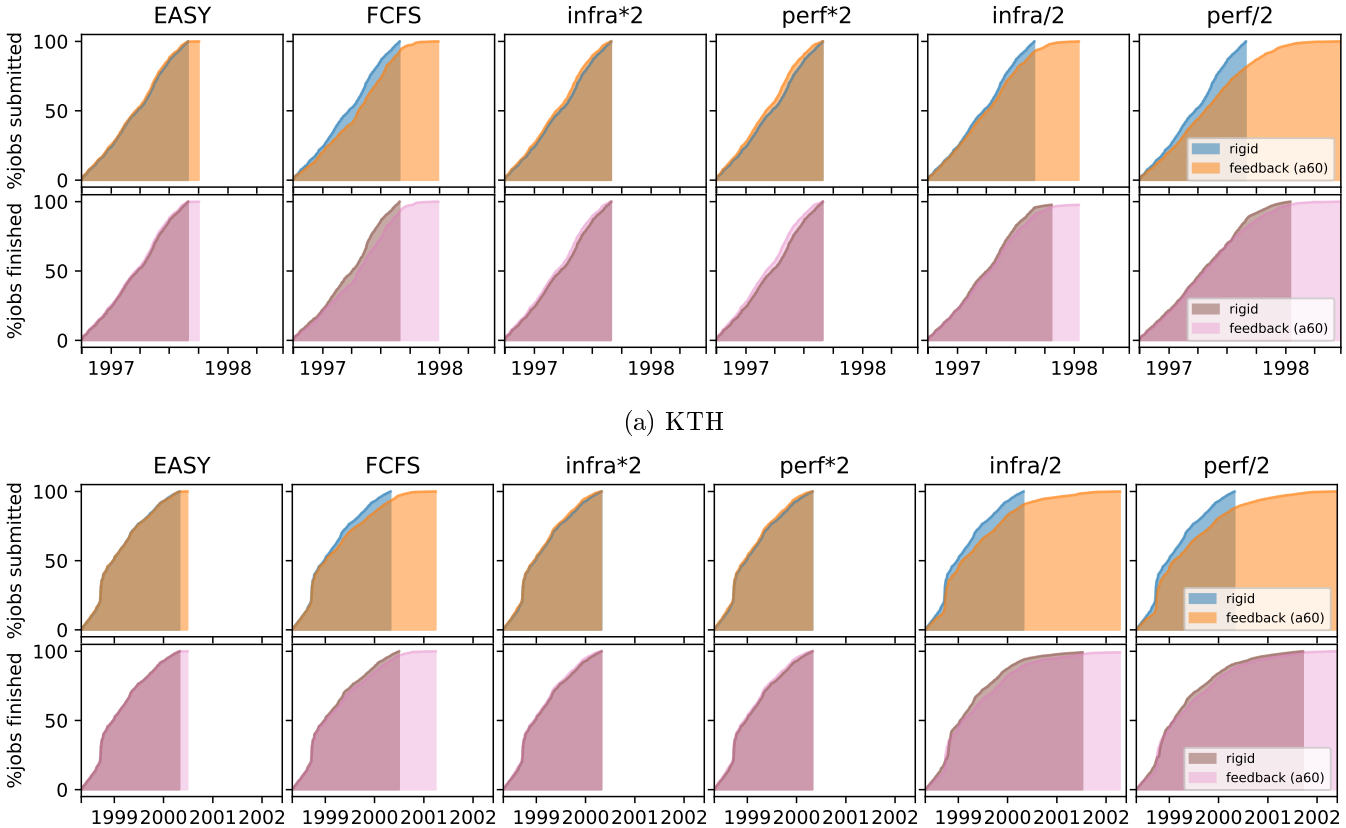
Regarding the schedulers, the graphs confirm that our implementation of **EASY** is closer to the scheduler used in the real infrastructures than **FCFS** is. Indeed, the rate of submission with rigid and feedback looks fairly similar in experiment **EASY**. This is an indication that the jobs get executed and finished around the same time (we remind that rigid replay preserves the *original timestamps* while feedback replay preserves the *think times*). With the scheduler FCFS, the patterns of submission in Figure V.2 look more disrupted. This is due to the absence of backfilling: big jobs are blocking the queue, delaying the execution of small jobs, and the users have to wait for their termination before submitting the next jobs that *depend on* them.

Finally, Figure V.3 also displays in its bottom graphs information about rate of job *terminations*. This time, the distributions with rigid replay (in brown) vary between the experiments, because contrary to the submission times, the finish times of jobs do get affected by the infrastructure or scheduler used. The makespans given in Table V.1 are reflected in the horizontal spans of these plots. We get to see that if the makespans in experiment **EASY** were larger with feedback compared to rigid, it is only due to a few jobs that got delayed. Indeed, the right part of the pink curve after the brown curve ends is essentially flat. Also, we note that the cumulative distributions of job terminations look fairly similar for all experiments, if we disregard the side effects in the end. This



(a) KTH (b) SDSC

Figure V.2: **Distribution of submission times** with rigid (blue) and feedback (orange) replay models.
Here we only plot feedback a60, but the picture with a0 is sensibly the same.



(a) KTH

(b) SDSC

Figure V.3: **Cumulative number of jobs submitted** (top) **and finished** (bottom)

is an indication that throughput (defined as average number of jobs finished per week) is relatively independent of the replay model. We come back to that in the Discussion (Section 7.4)

If we were able to better characterize the effect of the replay model thanks to the distributions of submission times, we lack reliable metrics to measure it quantitatively. We attempt to fill this gap in the following, by defining three new metrics: mean lateness, relative lateness and additional lateness.

5.3 New metric 1: mean lateness

First, we define the *lateness* of a job, a fundamental quantity that will allow us to define the three metrics:

Definition 23. *The **lateness** $\ell(i)$ of job j_i is the difference between its submission time in the replay and in the original record: $\ell(i) = \hat{a}_i - a_i$.*

Consequently, for a set of jobs $(j_0, ..., j_{n-1})$, we can compute our first metric, the **mean lateness**, denoted $\bar{\ell}$:

$$\bar{\ell} = \frac{1}{n}\sum_{i=0}^{n-1} \ell(i) = \frac{1}{n}\sum_{i=0}^{n-1}(\hat{a}_i - a_i) \tag{V.3}$$

Mean lateness can be calculated per user or on the whole simulation. It measures *how many days difference there are on average between the original submission times and those in the simulation.*

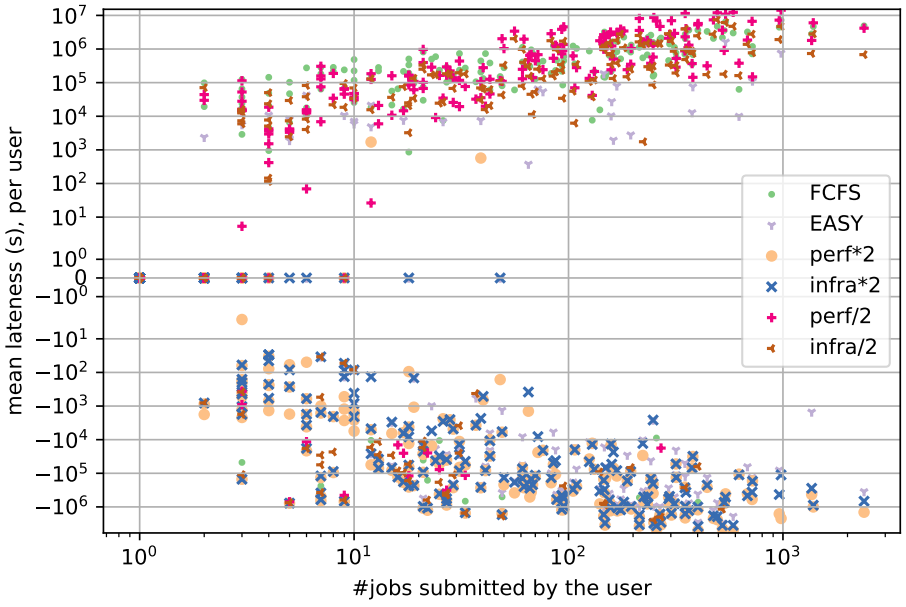
Mean lateness on all jobs Calculated on all the jobs in the simulation, *mean lateness* characterizes "the extent to which the orange curve is shifted to the right" in Figure V.2. Values of *mean lateness* for each experiment are given in Table V.2. We can read for example that jobs in experiment **perf/2 a60** are submitted 44 days later on average with KTH log, and 13 days *earlier* in experiment **perf*2 a60**.

We also notice that values are positive for experiments **FCFS**, **perf/2** and **infra/2**, indicating that jobs are submitted *later* on average, and negative for experiments **perf*2** and **infra*2**, a sign that jobs are submitted *earlier*. *Mean lateness* for experiment **EASY** are close to zero. This confirms our previous observations.

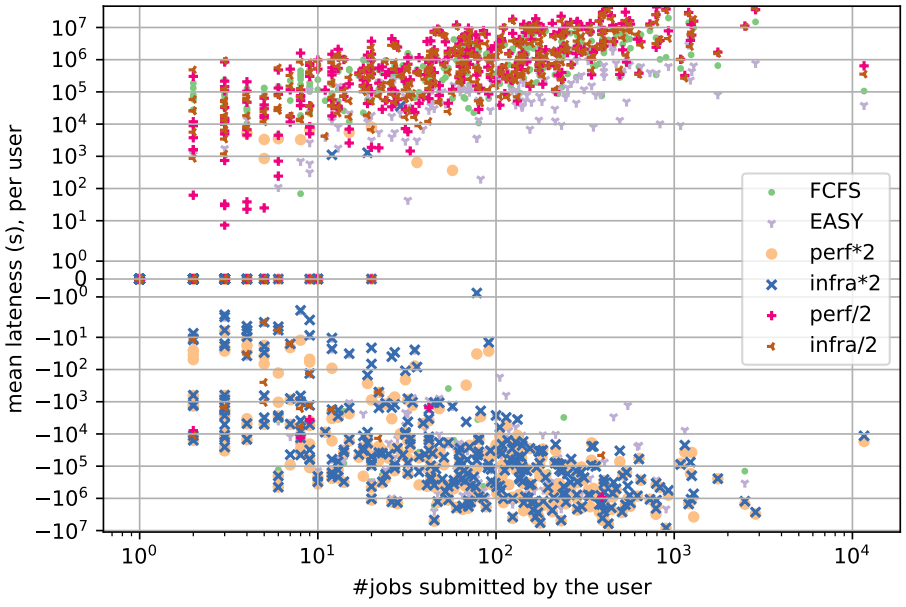
Mean lateness per user *Mean lateness* per user is plotted in Figure V.4. First, we see that values are very scattered. Depending on the user, *mean lateness* can be several orders of magnitude different. We can nevertheless make the same distinction between the experiments that have a positive lateness (**FCFS**, **perf/2** and **infra/2**), and experiments with a negative lateness (**perf*2** and **infra*2**).

More interestingly, we observe a drift to high values as the number of jobs submitted by the user gets higher. In other words, users that submit more jobs tend to have a greater (positive or negative) *mean lateness*. This makes sense, as the more the users submit jobs, the more they get to experience the feedback given by the infrastructure, hence the more they accumulate lateness. An implication of this drift, which is unfortunate, is that *mean lateness does not scale with the size of the workload.*

This makes the metric *mean lateness* unpractical to compare different workloads. That is why we introduce the two next metrics built to be independent on the number of jobs: *relative lateness* and *additional lateness.*



(a) KTH



(b) SDSC

Figure V.4: **Mean lateness per user**, for all experiments, with replay a60 (logarithmic scale). Each dot corresponds to one simulated user. A positive (resp. negative) value indicates that the user submitted later (resp. earlier) on average in the replay with feedback compared to the recorded log.

Table V.2: **New metrics calculated for all the experiments.** Units: mean lateness in days, additional lateness in seconds and relative lateness without unit.
Please note that lateness in a quantity that tends to accumulate for long chain of jobs. Taking the mean hides this distribution.

		KTH			*SDSC*		
expe	**replay**	**mean lateness**	**relative lateness**	**additional lateness**	**mean lateness**	**relative lateness**	**additional lateness**
EASY	a0	-3.36	0.99	-20.39	2.35	1.00	6.00
	a60	-4.47	0.99	-27.12	1.04	1.00	2.65
FCFS	a0	32.66	1.10	198.18	76.90	1.11	196.38
	a60	26.31	1.08	159.64	36.00	1.05	91.92
perf*2	a0	-12.40	0.96	-75.27	-11.04	0.98	-28.18
	a60	-13.31	0.96	-80.79	-11.55	0.98	-29.49
perf/2	a0	46.10	1.14	279.75	106.58	1.15	272.17
	a60	43.54	1.13	264.24	95.34	1.13	243.47
infra*2	a0	-8.65	0.97	-52.48	-8.58	0.99	-21.92
	a60	-9.32	0.97	-56.57	-9.23	0.99	-23.56
infra/2	a0	16.48	1.05	99.99	89.82	1.12	229.37
	a60	14.91	1.04	90.48	81.50	1.11	208.12

5.4 New metric 2: relative lateness

The metric **relative lateness** is the expression of mean lateness relatively to the length of the original workload. We want to see how significant the shifts in submission times in the replay are. Consequently, we define the length of the workload as the inter-arrival time between the first and the last job. It gives for relative lateness a dimensionless quantity:

$$\text{relative lateness} = 1 + \frac{\bar{\ell}}{a_{n-1} - a_0} \tag{V.4}$$

Values for this metric are given in Table V.2. A *relative lateness* > 1 corresponds to a *mean lateness* > 0, so a simulation where the submission times spread out over time. The maximum *relative lateness* in our experiments is reached by **perf/2 a0 SDSC**, with a value of 1.15. A way to interpret it is: "dividing the performances of the servers by two lead the users to accumulate a delay in their submissions, corresponding to 15% of the length of the workload".

5.5 New metric 3: additional lateness

Another way to make the metric independent on the number of jobs is to look at the "additional lateness" δ_i that accumulates with each new replay job $\hat{j}_i$:

$$\ell(i) = \ell(i-1) + \delta_i \tag{V.5}$$

Once again, the δ_i can be defined per user (the successive j_i would be the successive jobs submitted by one user) or on the whole simulation (the j_i would be all the jobs of the simulation, ordered by original submission time).

δ_i is a duration, that can take positive or negative values. Similarly to common scheduling metrics such as the waiting times, they fluctuate a lot with i. To understand the overall trend, one should look at their distribution.

However, taking the mean is not meaningful as the $\ell(i)$ would cancel out when we take the sum in Equation V.5: $\sum \delta_i = \ell(n-1) - \ell(0)$. Instead, to build a simple yet aggregated metric, we suppose that δ_i is constant equal to δ. Since $\ell(0) = 0$ with our replay model, we have by recurrence on i: $\ell(i) = i\delta$.

Injecting this in the definition of mean lateness gives:

$$\bar{\ell} = \frac{1}{n}\sum_{i=0}^{n-1} \ell(i) = \frac{\delta}{n}\sum_{i=0}^{n-1} i = \frac{\delta}{n}\frac{(n-1)n}{2} = \frac{\delta(n-1)}{2}$$

Thus, we propose the metric **additional lateness**, denoted δ, defined through the formula below:

$$\delta = \frac{2\bar{\ell}}{n-1} = \frac{2}{n(n-1)}\sum_{i=0}^{n-1}(\hat{a_i} - a_i) \tag{V.6}$$

We interpret this metric as the *additional delay that the users accumulate at each submission, in response to the feedback provided by the infrastructure.* Values of *additional lateness* on our simulations are given in Table V.2. For example, halving the server performances makes the users accumulate 280s of extra delay at each job submitted with KTH log, and 272s with SDSC (replay a0). On the contrary, doubling the performances makes the users submit an extra 75s earlier on average at each job for KTH, and 28s for SDSC.

5.6 Analysis of relative lateness and additional lateness results

Preliminary remark Looking at the definitions of the two metrics in Equations V.4 and V.6, we note that (relative lateness-1) and δ are roughly proportional to $\bar{\ell}/n$, assuming that there is an affine relationship between the length of the simulation and the number of jobs. This implies that *relative lateness* and *additional lateness* are linearly correlated. We were able to confirm it experimentally with our data: *additional lateness* and *relative lateness* feature a Pearson correlation coefficient of 0.997 with KTH log and 0.976 with SDSC, on the 80 data points of Figure V.8. Consequently, the analyses that can be made for one metric also apply to the other, and we will only present in the following the analyses for the metric *additional lateness*.

Which parameter influences the *additional lateness*? In our results (Table V.2), the parameter influencing the most the *additional lateness* is the infrastructure / scheduler. For a fixed log and replay method, we get very different values of *additional lateness* depending on the performances or number of servers or the type of scheduler. For instance in log **KTH** and replay method **a60**, *additional lateness* ranges from -80.79 to 264.24 days.

In second comes the specific log used for the replay. In our case, except for experiments **perf/2 a0**, **perf/2 a60** and **FCFS a0** where they are relatively similar, we observe a significant variability in *additional lateness* between the logs. The overall trends remain the same in both logs.

Finally, the change of replay method (a0 or a60) has the lowest influence on *additional lateness* in our results. A notable exception is experiment **SDSC FCFS**, where using a60 instead of a0 makes the *additional lateness* decrease significantly. A possible explanation is the presence of several flurries of very high activity by individual users in this log[6]. These flurries get grouped in a few sessions with a60, because they have close submission times. In the replay, they will therefore be submitted concomitantly. With a0, every job is in

[6] up to 11740 jobs submitted by the same user in less than 25 days, see "Usage Notes" in the page describing the log: https://www.cs.huji.ac.il/labs/parallel/workload/l_sdsc_sp2/index.html

a separate session that waits for its dependencies to finish before being submitted, which might lead to the increased delay with a rigid scheduler like FCFS.

***Additional lateness* per user** *Additional lateness* per user are plotted in Figure V.5. Unlike for *mean lateness*, the values are independent on the number of jobs submitted by the user: there is no drift compared to Figure V.4. However, they still depend on the specific user, with great variability. In fact, there are differences in *additional lateness* of more than 10'000 seconds between the 10th and 90th percentiles in all the experiments (Table V.5c). We also note that the median *additional lateness* per user are in the order of *hours* while they are in the order of *minutes* when aggregated at the level of the whole simulation (Table V.2). This means that despite the overall *additional lateness* being relatively low, the additional lateness experienced by most users is much more significant.

6 Discussion

In this section, we see how replay with feedback accounts for a change in infrastructure (6.1). Then, we study how the session delimitation method impacts the results (6.2). After that, we focus on the generalization of the new metrics by studying their scalability with regard to workload size (6.3). Finally, we point out the differences between our model and the original model from Zakay and Feitelson (6.4).

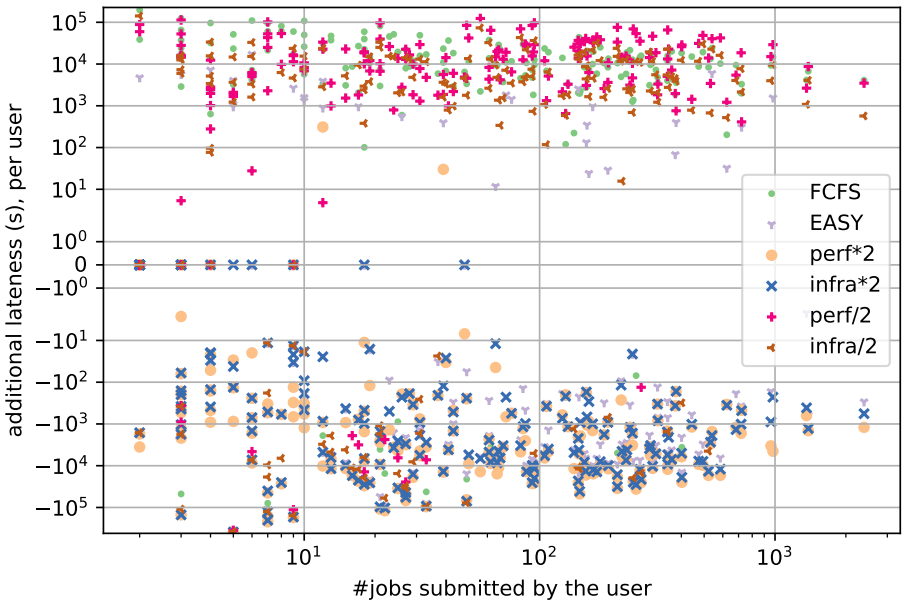
6.1 Influence of the change in infrastructure

Thanks to the replay model and new metrics, we are able to characterize the effect that a change in the infrastructure might have on user submission behavior. It will impact the submission times, shifting them forward or backward. Below is a ranking of the impact of the studied infrastructure change, from the earliest to the latest submission times in relation to the original times, based on Table V.2:

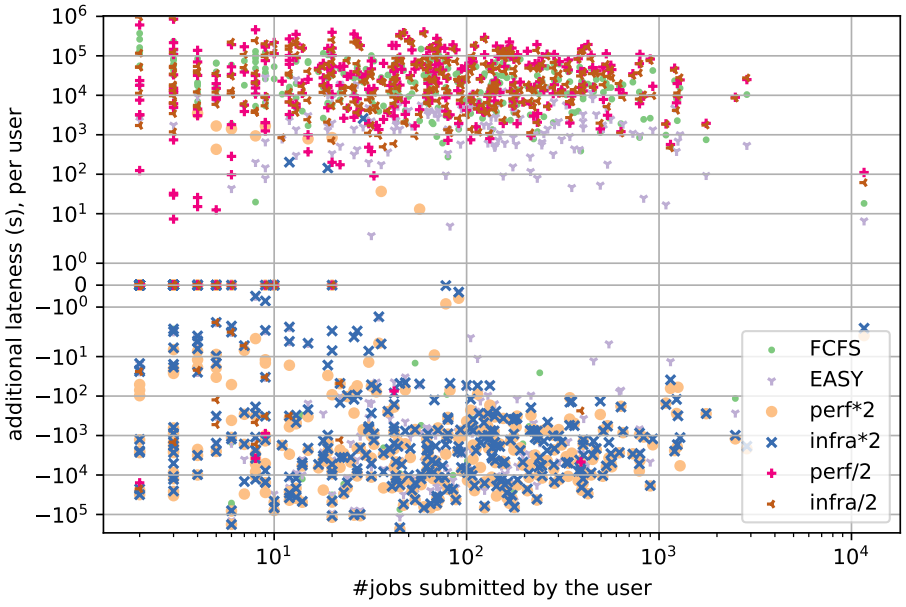
1. **perf*2** (earlier than original)
2. **infra*2** (earlier)
3. no change (**EASY**)
4. **infra/2** (later)
5. **perf/2** (later)

This ranking is verified by both logs, no matter the replay method (a0 or a60). For KTH log, the change to scheduler **FCFS** would rank between items 4 and 5 while for SDSC it would come between 3 and 4.

Importantly, we see that doubling/halving the server performance has a more significant effect in shifting the submission times than doubling/halving the *number* of servers (in absolute value). This effect is particularly visible with KTH log. In fact, **changing the performance directly affects the execution time of every job**. Changing the number of servers, however, has no effect on the execution times, but only **impacts the waiting times indirectly**. If the original infrastructure was already oversized, this change will have little effect. Note also that decreasing the number of servers below the maximum number of requested resources will cause some jobs to be rejected — hence considered immediately finished, for the replay method.



(a) KTH

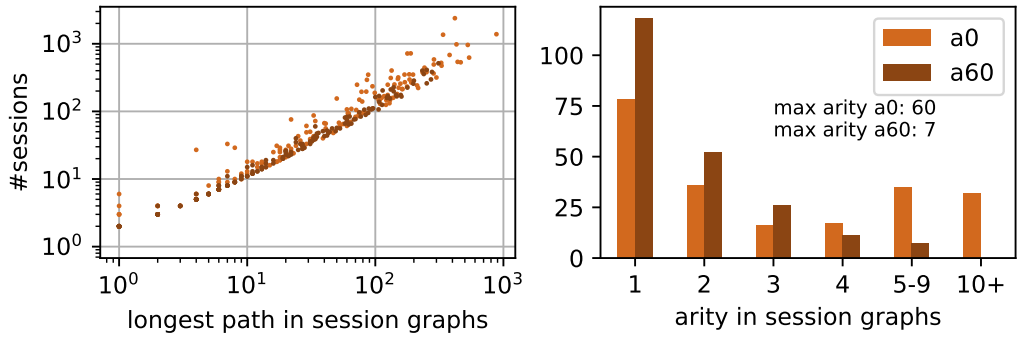


(b) SDSC

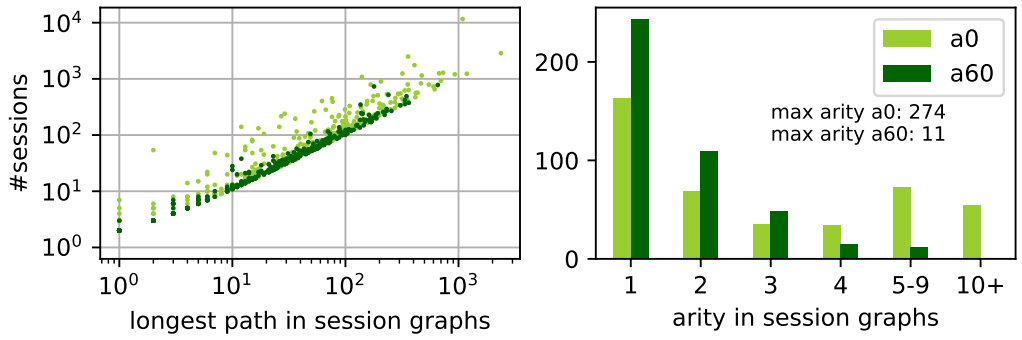
	KTH			*SDSC*		
	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
FCFS	-5	8775	44525	0	12141	67078
EASY	-17980	-699	918	-8898	0	4180
perf*2	-26247	-2186	-49	-26170	-1803	0
infra*2	-25321	-1553	-18	-21738	-1288	0
perf/2	0	10485	47168	0	15190	153976
infra/2	-8695	2586	19434	0	13424	119480

(c) Percentiles on the distributions of additional lateness plotted above (in seconds)

Figure V.5: **Additional lateness per user**, for all experiments, with replay a60.



(a) KTH



(b) SDSC

Figure V.6: **Longest path and arity distribution of user session graphs.**

6.2 Influence of the delimitation method

At the root of the feedback model is the partitioning of jobs into sessions (see Definition 14). We remind the reader that we use two methods in the experiments: a delimitation on inter-arrival of 0 minutes, corresponding to single-job sessions, and 60 minutes, which was an optimal parameter found by Zakay and Feitelson [125]. Inter-arrival times of more than 60 minutes led to sessions of unrealistic length in their study. The characteristics of our resulting session graphs are shown in Figure V.6.

Session graph structure We observe a large diversity in the size of the session graphs for the different users: some contain only one session, while others have thousands of sessions with the longest path inside the graph of several hundred sessions. These reflect the intrinsic differences between HPC users that use the platform for various motives and with different levels of activity.

Unsurprisingly, the use of delimitation a60 reduces significantly the number of sessions in the graphs, hence the longest paths. More notably, we observe that a60 reduces greatly the arity of the graphs: there is no graph with an arity greater than 11 with this delimitation method. In other words, there are less "sessions in parallel" for given user with a60.

Also, an analysis on the think times reveals that 75% of edges have a think time <10h and 35% are <1h for delimitation a0, while 50% of think times are <10h and around 13% are <1h with a60.

In short, **the session graphs produced by the two delimitations methods have very different structure.**

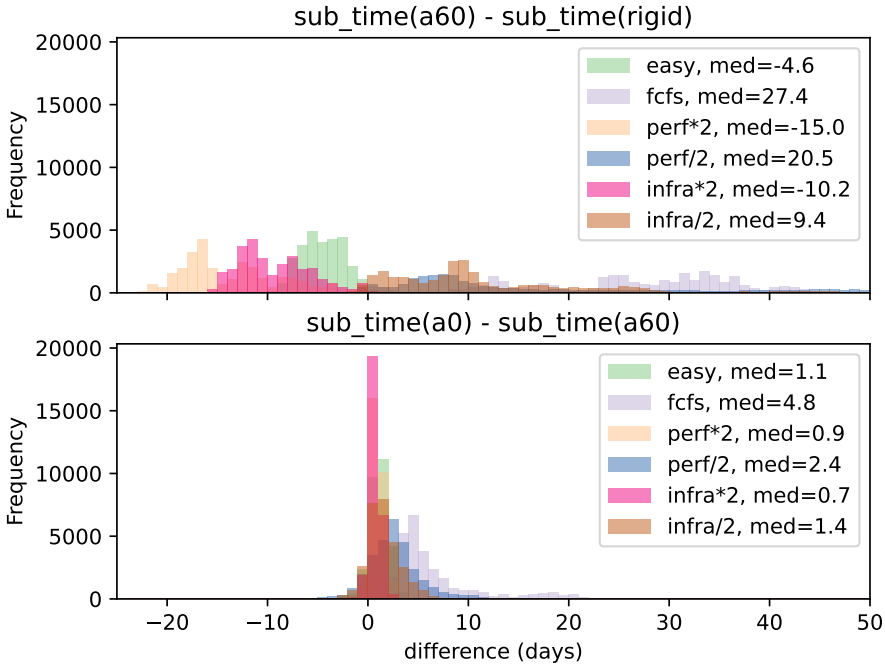


Figure V.7: **Distribution of the difference in submission timestamps** between rigid and a60 replay methods (top) and a60 and a0 (bottom), KTH log. Note: the top graph corresponds to the definition of lateness, and it confirms the ranking made in 6.1.

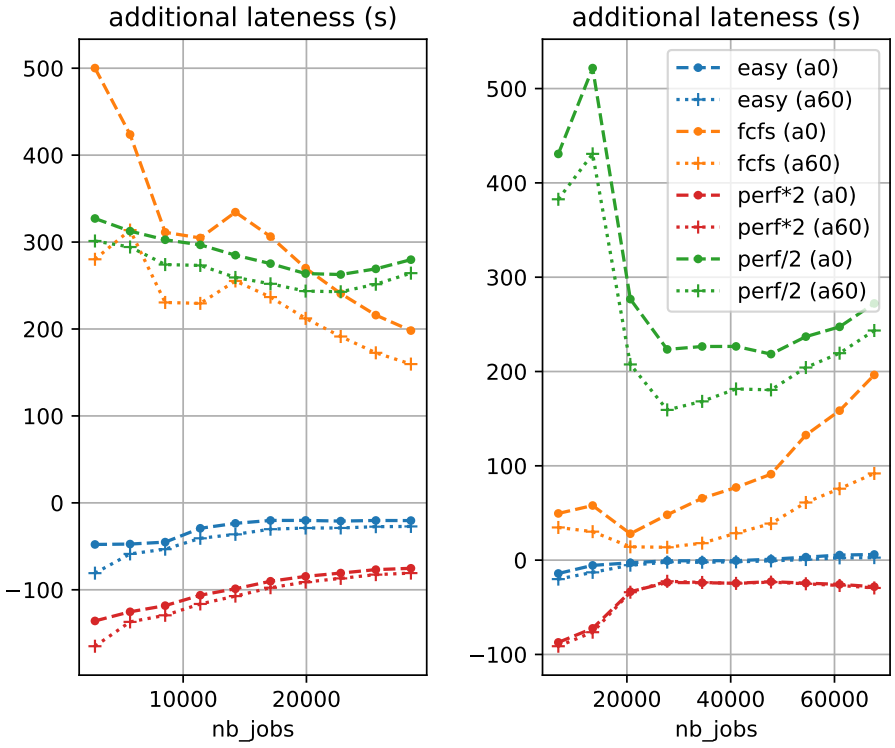
In the experiments All the same, and as we already mentioned in the previous sections, the two delimitations studied have little influence on the results. Scheduling metrics and our new metrics are roughly the same (Tables V.1 and V.2) and submission time distributions look very similar. However, if we look more carefully, we note that *mean lateness* (and hence *relative lateness* and *additional lateness*) are always lower with a60. To see that more in detail, we plotted the distribution of the difference in submission times between the different methods in Figure V.7.

As we can see, the difference between the submission time in a0 and a60 is positive for almost all jobs. **Delimitation a0 lead to slightly (a few days) later submission times than a60**. This effect is explained by the greater complexity of session graphs obtained with a0 that we explained above. Having more sessions and more dependencies results in less flexibility during the replay. If one job gets delayed in its execution, it will have more impact because it has more successors.

6.3 Scalability of relative lateness and additional lateness

The metrics *relative lateness* and *additional lateness*, that we introduce in Section 5, are influenced by several factors, including the simulated platform, scheduling algorithm, workload and delimitation method used to define the user sessions. Ideally, we would like them to be the signature of a particular infrastructure, with its specificities in terms of users, platform, scheduler, and to stay the same no matter what time window we are looking at. In other words, we would like these metrics to be independent on the *length of the workload*.

We already saw that the metric *mean lateness* depends on the length of the workload, since we observed a drift to high values as the number of jobs submitted by the users increased in Figure V.4. Figure V.5 was quite convincing in that regard, as it does not



(a) KTH (b) SDSC

Figure V.8: **Scalability of the *additional lateness* metric.**
For each point, a new simulation has been run with a subset of the workload as input: the subset contains only the n first jobs (ordered by submission time) of the original workload. Then, *additional lateness* is calculated on the output using Equation V.6.

show such a drift. To check the scalability of these metrics, we rerun the simulations on subsets of the original trace, containing the 10, 20, ..., 90% first jobs. The truncated traces undergo session partitioning and the corresponding session graphs are used as input for new simulations. The metrics obtained for each truncated trace are shown in Figure V.8.

The results are mixed. When only the infrastructure is modified, *additional lateness* seems to stabilize with the number of jobs as input. For experiments **easy** (no change in infrastructure) and **perf*2** for example, *additional lateness* increases at first with the number of jobs, but seems to plateau after 20000 jobs with both logs. The case of experiment **perf/2** is more problematic as *additional lateness* starts by decreasing until 20000 jobs but increases again thereafter, especially with SDSC log. This means that the delay caused by halving the performances does not only add up with time ($\ell(i)$ increases), but the additional delay for each new submission also increases ($\delta_i = \ell(i) - \ell(i-1)$ increases).

However, *additional lateness* is not scalable when the scheduler is modified (experiment **fcfs**). In this case, the metric does not seem to stabilize and behaves in the opposite way in both logs (decreasing for KTH and increasing for SDSC). It is hard to say what is intrinsic to the metric and what is due to heterogeneity in the input workloads.

To conclude, in our experiments, *additional lateness* scales rather well with the size of the workload for a change in infrastructure, but not for a change in scheduler. We therefore recommend anyone using this metric to do the same scalability checks.

6.4 Comparison with related works

In this part, we come back to the differences between our model and Zakay and Feitelson's [122]. Like them, we do a replay with feedback on submission times only. The method is based on think times, and on a session partitioning of the original workload. Compared to them, we only preserve the think times between *sessions*. Zakay and Feitelson introduce an additional notion of "batch", which are groups of overlapping jobs within a session. From there, they propose three methods:

1. 'adjusted': preserve the think time between *batches*. This is the method that is the closest to ours.

2. 'distribution-based': when a batch becomes free, submit it if the current time is in a "period of activity" of the user (working day, working hours). Otherwise, shift it to the next period of activity. This method requires assumptions on periods of activity, that they manage through a probabilistic model.

3. 'fluid': do the session partitioning, and preserve the session start and end times for users in the replay. The sessions will be the "periods of activity". The batches are submitted only during these periods, if they are free.

In this chapter, we did not reproduce their methods to compare our results to theirs. The reasons are twofold:

Firstly, in absence of a validation method, such a comparison would be inconclusive. For example, we expect their 'adjusted' method to show similar results than ours, but we would have no way to conclude which one is the most realistic. Similarly, 'distribution-based' artificially restores the seasonality (see 7.3) to the cost of an additional set of assumptions on periods of activity for users, making it difficult to know if it kept the fundamental features of the original workload trace.

Secondly, we disagree with the assumptions behind the 'fluid' model. We think that the *sessions* that are deducted from the recorded trace and the *periods of activity* are two separate notions. If a user does not submit any job one day, it does not necessarily mean that she was not working that day, but rather that she did not have anything to submit. If the performances of the platform were different, she might have had something ready to submit that day.

Instead of proposing a comparison based on hypotheses and beliefs, we preferred to implement the simplest model of replay with feedback, and provide a solid theoretical and software base for future contributions in the domain.

7 Limitations

The model of replay with feedback presented here allows accounting for effects that are invisible in traditional simulations. However, our work has some limitations, in both the experiments and the model, that we point out in this section.

7.1 Limits of the experimental campaign

The experimental campaign proposed in this chapter only includes two workloads, which are both quite old (recorded before the year 2000). They were carefully selected because they disclosed information on their scheduling policy and featured simple platforms (homogeneous with monocore servers). Similarly, we studied only two monocore schedulers

(EASY and FCFS). These were chosen as they are the most commonly used in the literature and correspond to the workloads.

Nevertheless, since our work focuses on the model of *replay* and not specific scheduling results, we argue that our campaign is sufficient to reach our conclusions, which would extend to other workloads and other schedulers. Furthermore, we remind that we took particular attention to make the experiments reproducible. Hence, it should be easy to re-run the campaign with any workload available in the Parallel Workload Archive.

The remaining of this section will focus on the model limitations.

7.2 Only one type of user response

First, let us recall that our method of replay with feedback focuses only on submission times. In reality, the response of users to system performances goes well beyond: they can react by submitting more or less jobs, modifying their applications, leaving the infrastructure etc. However, we chose to stick to feedback on submission times for three main reasons:

1. Even for feedback on submission time, we cruelly lack related literature and methods of validation (see 7.5). We found no literature on the other types of user response.

2. Multiplying the parameters that we change compared to rigid replay makes it harder to deeply analyze the effect of the proposed feature. We preferred to proceed by incremental steps.

3. Allowing for more types of user response might alter the input workload even further, to the point where it is not easy to know if it kept its fundamental structure.

7.3 Day/night variability

In real infrastructures, we observe a day/night and weekday/weekend variability in the user submissions. This variability is also present in our input data, as we can see in the top graph of Figure V.9, displaying the number of submissions per hour of the week in the recorded trace KTH.

Unfortunately, with our method of replay with feedback, the submission times get shifted around, and this variability is lost (graphs a0 and a60 in Figure V.9). This leads to both a loss of realism (e.g., users submitting at 3:00 in the night) and distorted results, since the workload becomes evenly spread out in time.

Such a variability could be fixed for example by adding assumptions on activity times for users (like in the 'distribution-based' user model [122], see 6.4).

7.4 Remaining rigidity in the feedback model

Our method, although better than rigid replay in this regard, is unsatisfying to fully capture user response to feedback from the infrastructure. In fact, as already mentioned, doubling the performances of servers only stretches the length of the submission period by 0.99 or 0.98 (see *relative lateness* in Table V.2). A more significant rebound effect would have been expected as a consequence of such a performance gain. An analysis of throughput and utilization in the different experiments, plotted below in Figure V.10, enlightens us on the reasons behind this limited rebound.

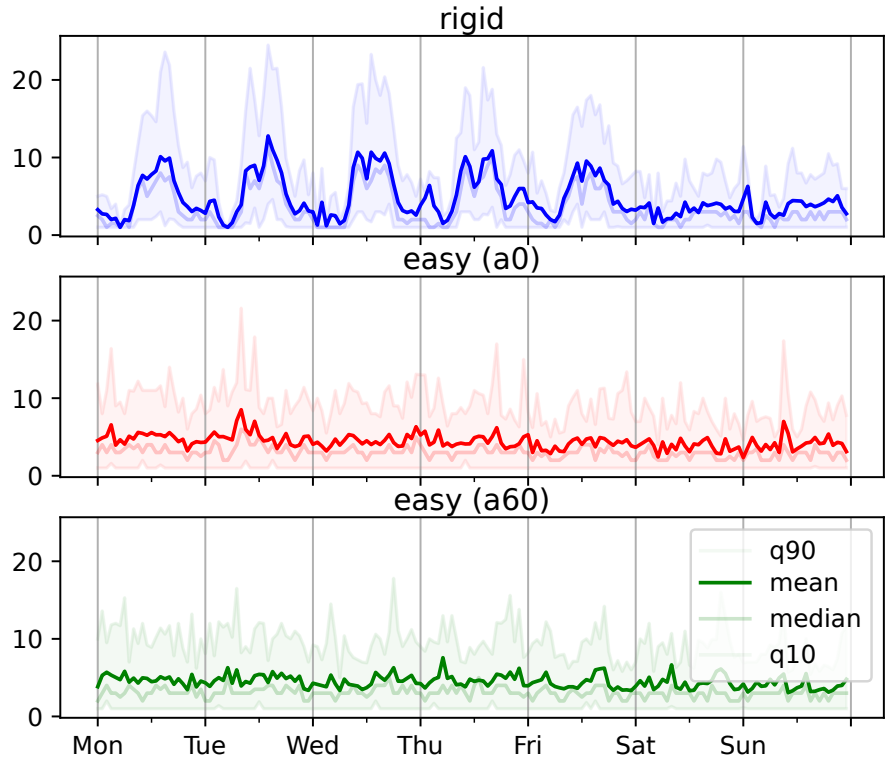


Figure V.9: **Number of submissions per hour, aggregated by week**, KTH log

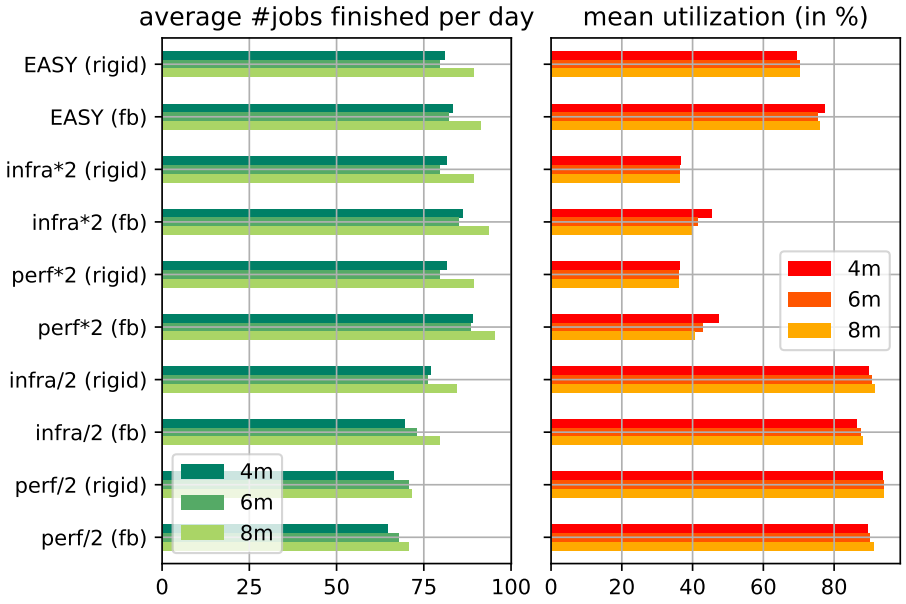


Figure V.10: **Throughput** (average number of job terminations per day) **and mean utilization** (average number of computing servers), KTH log. The metrics are calculated on a time window starting two weeks after the beginning of the simulation and with length 4, 6 and 8 months, to leave away beginning- and end-of-simulation edge effects.

Limited rebound in throughput and utilization As we notice, doubling the performances or number of servers has no effect on throughput with the rigid replay model (**rigid perf*2** and **infra*2**): it remains at around 80 jobs/day (left graph in Figure V.10). In fact, because the job arrivals are fixed, increasing the performance of the infrastructure only leads to lowering the mean utilization (right graph).

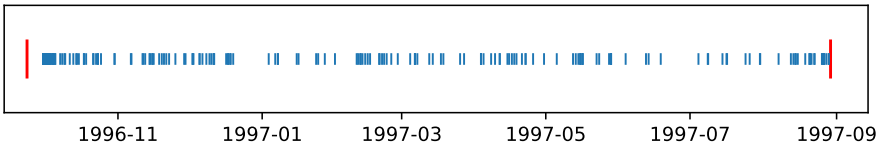
We would expect the feedback model to correct this effect. Unfortunately, this is only partly the case: the change in infrastructure leads to a higher throughput, but only 5 to 11% greater compared to rigid (**feedback infra*2** and **perf*2**). In parallel, the mean utilization drops from around 70% to 40-47%, only slightly above the level with **rigid infra*2** and **perf*2** (36-37%)

Similar effects can be observed with the experiments **perf/2** and **infra/2**, with, this time, a saturation of the platform (utilization >90%).

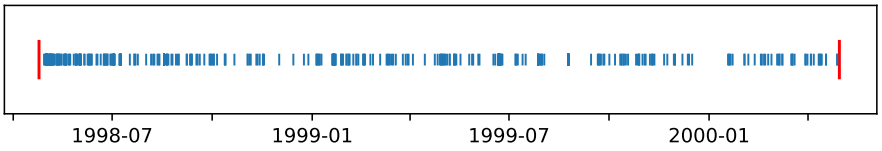
Explanations Our model does not allow us to achieve the level of realism needed to handle the case above. It still contains two sources of rigidity.

1. **Think times are constant in the model.** Even if the infrastructure yields the best performances, they will never be reduced. To understand, let us come back to the simple session graph drawn in Figure V.1. Better performance can reduce the length of the session boxes, i.e., the turnaround time of jobs. But the brown arrows representing the think times will always keep the same length. As a result, if a user stops submitting for one month and comes back afterwards, the first new job she submits will have a think time of one month, whether she actually needed all this time to "think" or she was absent for other reasons.

2. **The first job submitted by a user is always replayed at its original (recorded) timestamp.** In both KTH and SDSC logs, new users arrive in the platform up to a few days before the end of the record. We plotted their dates of arrival in Figure V.11. This explains why the length of the submission period is never significantly reduced when the performances are better.



(a) KTH



(b) SDSC

Figure V.11: **User arrivals in the platform.** Each vertical blue line represents the first time one user submits in the platform. The red vertical lines are the start and end time of the original log.

The two factors of rigidity above are the expected behavior of our model. To overcome these limits, the model would have to be extended with new assumptions. For example, Feitelson proposes to introduce a model of arrival/departure of users in later work [128]. Additionally, we could consider a user as a new user after a long think time.

7.5 On the validation of the model

As we saw, the current version of our replay with feedback model is perfectible. Despite some good properties compared to rigid replay, we cannot claim that it is more scientifically valid. In fact, verifying the validity of such a model is a challenging task, and we are not aware of any work attempting to do so. Instead, we can only provide ideas on the ways such a validation could be carried out:

- making a survey with users of grid/HPC infrastructures to understand what their behavior is in reaction to feedback (see Wolter et al. for an example of HPC user survey [129]),
- collecting data on a real infrastructure that underwent a major change and check if the model is able to predict the way the users adapted to this change (see Klusáček et al. for the analysis of Metacentrum log that underwent a major reconfiguration [114]).

We leave these avenues of research for future work.

8 Conclusion

In this chapter, we challenge the traditional way to simulate distributed systems by introducing a feedback loop in the workload model. Compared to using a pre-determined workload (historical record or generated) in the simulation, we let the workload adapt to the simulated performance of the system, in the same way that real users would adapt their pace of submission to the response they get from the infrastructure. This novel way of doing simulation, that we call "replay with feedback", was first proposed by Zakay and Feitelson [122]. It consists in using a historical workload as input and partitioning it into "sessions of work" for each user. During the simulation, we no longer preserve the original timestamps of submission, but rather the *think time between sessions*, i.e., the time that elapsed between the termination of all jobs in a session and the submission of the next one. Zakay and Feitelson introduce a notion of batch within session that are the sets of jobs whose execution overlap.

We complement their approach by providing a slightly different replay model, leaving aside the notion of batch to keep the model as simple as possible. Our model is implemented in our open-source software Batmen, making it easily reusable for implementing and studying other replay models. We apply our model and implementation by running a reproducible experimental campaign with two historical logs, with which we obtain similar results. The experiments show how replay with feedback can be used to predict the impact of a change in the infrastructure (computing performances, number of servers, scheduling algorithm) on user submission behavior and scheduling performance. In our case, decreasing the performances or number of servers make the users accumulate a delay at each submission compared to the baseline. On the other hand, increasing them instead made the users submit earlier on average. A change in performances has a more significant effect than a change in number of servers. Lastly, we introduce three novel metrics, independent of the specific replay model, to describe the effect of feedback on user submission. *Mean lateness* measures the average time difference between the original submission times and those in the simulation. *Relative lateness* gives an expression of this time difference, relative to the length of the simulation. Finally, *additional lateness* expresses the additional delay that the users accumulate at each submission, in response to the feedback provided by the infrastructure.

This work contributes to what is in our opinion a fundamental yet much under-researched topic within the field of distributed system simulation. It requires to rethink the way we do simulation and interpret the results. Performance of computer systems are not only about bandwidth or number of operations per second, but rather the utility that they bring to the humans using them. Not taking the human factor into account leads to large overestimates of potential gains, as it neglects the rebound effect inherent to efficiency.

Chapter VI

Case study: sufficient use of the cloud in a professional context

Contents

In the previous chapters, we focused on data center users who were directly interacting with the IT platform, i.e., the "direct users" in the definition of Chapter II (see II.1.1.1). We modelled different types of sufficiency behaviors and simulated their impact on the data center energy consumption.

However, most data center users are in fact *not* direct users. They are rather "indirect users", i.e., everyday Internet users, browsing websites and using cloud solutions, often without knowing where the servers are located or how much environmental footprint they represent. For cloud data centers, the indirect users represent the vast majority of end-users. It is therefore essential to reflect about what "digital sufficiency" means, in their case.

The positioning is twofold. On the one hand, it allows us to widen our scope by taking into account a larger share of end-users, and bring out new types of sufficiency behaviors. On the other hand, direct and indirect users are not completely uncorrelated. We can assume that for every service used by an indirect user there is one (or several) direct user, at the other end of the chain of intermediaries, taking care of provisioning enough resources to meet the service demand. Identifying the sufficiency levers for indirect users allows understanding what sufficiency behaviors direct users can (or cannot) adopt.

To investigate digital sufficiency for indirect data center users, we have to step out of our scientific discipline, Computer Science, and take a social science perspective. Understanding what is *sufficient* for users requires to meet them, collect their personal experience and question their *needs* for digital technologies.

In this chapter, we present such a study, that I made in collaboration with Prof. Patricia Lago from the Vrije Universiteit Amsterdam, and that we published in 2023 at the

conference ICT4S [C3]. We start by presenting the background and the three research questions of the study in Section 1. The details of our qualitative method and its execution are given in Section 2. Section 3, 4 and 5 provide the study findings, following the research questions. Our findings are discussed in Section 6, and the study limitations in Section 7. Finally, we conclude the chapter in Section 8, linking back the results to the rest of the PhD manuscript.

1 Description of the approach

I started to collaborate with Patricia Lago at the International Summer School on ICT for Sustainability 2021[1]. During this week and with other researchers, we started to look at the cloud computing model[2] and identify its shortcomings with regard to sustainability. In fact, the cloud allows access to everything, always and everywhere. It is often marketed as immaterial or even sustainable by tech companies, but may lead to "unsustainable patterns" like overconsumption or superfluous usages. We identified ten such patterns that we published in a paper at the workshop LIMITS [C2].

For the study described in this chapter, we decide to focus on **cloud usage in professional life**, specifically in the context of flexible work (see text box page 107). The aim is to build initial knowledge in the opportunities for digital sufficiency at work, by using qualitative research methods. We conduct three focus groups with a total of 11 participants working in two different companies and analyze the collected data with a thematic analysis. We look into the participants' daily work activities performed on a computer or phone (simply called "tasks") and investigate the following research questions:

(RQ1) Which tasks are cloud-based? Are there differences in different working settings?

(RQ2) What types of cloud-based tasks are perceived as *necessary*, and under which circumstances?

(RQ3) What is the perception of the benefits and challenges when superfluous activities are provided *on demand*?

2 Study design and execution

To address our research questions, we apply a mixed-method empirical research design that is intended to be conducted inside companies. For each participating company, it combines a preliminary interview with a knowledgeable person, with one or several focus groups with employees.

2.1 Preliminary interview

The first study step is a one-hour interview consisting of 11 questions with a top manager or IT responsible having a good overview of the company. The intention is to (i) identify the working culture and the company's vision for flexible work, and (ii) gather data on the IT infrastructure and solutions used to support work. The interviewee also helps identify

[1] Summer School organized remotely, program available at https://www.lorentzcenter.nl/international-summer-school-on-ict-for-sustainability-2021.html

[2] The term *cloud* in this chapter is to be understood in its broad sense i.e., "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources" [130], and for end-users (SaaS model).

Flexible work, cloud usage and environmental impact

Flexible work is, in a definition by the UK government, "a way of working that suits an employee's needs, for example having flexible start and finish times, or working from home" [131]. For some categories of jobs like research or consulting, new technologies, and especially mainstream cloud solutions (Google suite, Zoom, Microsoft Teams, etc.), have enabled people to work more flexibly by doing remote work. In his thesis, J. Stiles investigates the impact of cloud computing on work location decision-making with qualitative and quantitative research methods [132]. He identifies five categories of cloud usage for flexible work, namely (i) using messaging platforms across multiple devices, (ii) accessing shared company databases, (iii) video/audio conferencing, (iv) document-sharing and collaboration, and (v) online project management.
The COVID-19 pandemic accelerated even more the adoption of remote working. During the first global lockdown in 2020, many people were forced to work from their home, resulting in a growth of 15-20% of Internet traffic within a week (and 200% for remote working applications like VPN and video conferencing) [133].
In terms of environmental footprint, remote working is commonly admitted having a rather positive impact, by reducing physical commuting. Nevertheless, we see at least two reasons to be vigilant. First, the true environmental consequences of remote work become more complex when energy use at home, induced Internet traffic and other rebound effects are taken into account [134]. Second, digitalization in this area continues to develop at a fast pace. The calculation should be revised when innovations such as spacial video, digital smell or digital touch [135] come into being, with all their induced effects.

the types of roles and the specific people covering such roles within the company, and hence that are relevant to include in the focus groups. Finally, he or she acts as a champion to recruit such people.

2.2 Focus groups

The second and main step of the study consists in a series of focus groups organized as follows. The participants gather around a table and are given a sheet to fill out that they can use for individual note-taking. The facilitator has slides with the prompts to guide the discussion and a white board for collective note-taking. Each session lasts about two hours and follows an identical structure:

1. **Introduction and context.** The context of global greenhouse gas emissions and the increasing projected contribution of the ICT industry is presented by the facilitator. The concepts of "digital sufficiency" and "flexible working" are defined and the goal of the study is explained.

2. **Warm-up: smartphone usage.** The participants are asked to choose from a list the three *most frequent* uses of their smartphone. Then, they choose the three uses that they find the *most necessary* for them. Going around the table, the facilitator asks: "is there one task that you do frequently but is not on your necessary list?" This warming up exercise is used as an ice-breaker and an invitation to radical thinking. The discussion allows the facilitator to highlight the addictive nature of new technologies and the differences between *needs* and actual *usages*.

   From here, the data collection begins:

3. **Working week organization.** Going around the table, the participants describe the organization of their working week with regard to flexible working (work location, working hours, work-life balance).

4. **Cloud-based usages for flexible work** (RQ1). In the different working settings previously identified, the participants list the daily tasks they do for work on their computer or phone and identify the ones that are cloud-based. The discussion takes the form of a brainstorming, with the facilitator leaving them time to think and take individual notes before sharing to the group. During the discussion, the facilitator notes the tasks on the white board and links them to specific working settings.

5. **Tactics towards sufficiency** (RQ2). For each cloud-based daily task, the participants are invited to reflect: "could you do without? If yes, how? If no, why?". The discussion is left open and interactive, with the participants using their notes and the white board to recall the previously collected tasks and the facilitator making sure to cover most of them.

6. **On-demand activation** (RQ3). Finally, the participants are asked to focus on possible ways to embed sufficiency in their own work practice. They discuss the potential benefits and challenges for different cloud-based tasks.

7. **Wrap-up.** The facilitator closes the session and thanks the participants, after making sure that nothing was forgotten and answering potential questions.

The slide-deck for the focus groups contains 14 slides that were made visual by the use of colors, icons and minimal text. They are available in our replication package (see Section 2.5)

2.3 Study execution

We carried out a dry-run of the focus group session with five colleagues to collect feedback and check for potential problems. After this session, we made minor changes on the slide-deck and decided to help the participants keeping track of the conversation by distributing a pre-filled sheet for individual note-taking and taking collective notes on a white board.

Then, we rolled out the study between November and December 2022 in two companies: Company A (a small IT services and consulting company) and Company B (a large business services and consulting company), in the Netherlands. The participating companies were found within our network of industrial relations, and we had previous collaborations with them in sustainability-related projects.

We conducted a preliminary interview by videoconference with our contact in each company. Afterwards, these people kindly took charge of the recruitment of participants for the focus groups, by reaching out internally and making sure to get a representative panel of roles and responsibilities. They both participated themselves. The focus groups took place in-person on company premises. We carried one focus group at Company A with four participants and two focus groups at Company B with three and four participants, respectively. Table VI.1 provides the anonymized demographic information about the participants. Note the over-representation of home workers in our sample, specific to the Netherlands and the participating companies' culture.

Table VI.1: **The focus group participants.**

P#	Age	Gender	Occupation	FG#	work locations*
1	36-45	M	Consultant	A	**H**, O, C
2	26-35	M	IT developer	A	**H**, O
3	36-45	M	IT developer	A	**H**, O
4	56-65	M	Consultant	A	**H**, O, C
5	36-45	F	HR manager	B1	**H**, O
6	18-25	M	HR recruiter	B1	**H**, **O**
7	36-45	M	IT support	B1	**H**, O
8	26-35	M	Sustainability Procurement	B2	**H**, O
9	18-25	F	Sustainability Operations	B2	**O**, H
10	46-55	F	HR manager	B2	**H**, O, CW
11	46-55	F	Sustainability Procurement	B2	**H**, O

*we reported the work locations from which the participants declared to work more than one day a week (**bold**: more than two days a week), for a typical 5-day working week.
Abbreviations: H: Home, O: Office, C: Customer site, CW: Co-Working space.

2.4 Data analysis

The interview and focus group sessions were video-recorded. The focus groups were transcribed, producing 67 pages of text from 5 hours of discussions, constituting the primary data for analysis.

We performed a thematic analysis [136] on the data, with the help of the qualitative analysis software Saturate[3]. This tool allows to import the transcripts and associate codes to sentences or paragraphs. The codes can be sorted in categories and subcategories. We progressed by open coding, defining and modifying the codes through the analysis. I generated the initial codes from the transcripts and Patricia Lago reviewed them critically, indicating where she would have coded differently. Progressing by iterative steps, we produced a Codebook consisting of 248 codes sorted by three levels and by research question.

2.5 Replication package

For the sake of transparency, we provide a replication package in Zenodo [137] containing the material used in the study. The package includes:

- the interview guide for the preliminary interview,
- the focus group slides and note-taking template, and
- the Codebook containing all the extracted codes along with their definition, number of occurrence in the transcript and description on when the code is applicable.

For privacy reasons, and upon agreement with the participants, the video recording and transcripts are not included, and the data is anonymized.

[3]free browser-based qualitative data analysis software available at www.saturateapp.com

3 Findings 1: cloud-based usages for flexible work

In this and the next two sections, we expose the outcome of the focus groups, following the outline that has arisen from the thematic analysis. Along the way, we highlight a number of "suggestions towards sufficiency" that emerged from the analysis of the results.

To start, our participants' cloud experiences in relation to flexible work are presented here in three parts: the list of their digital activities (3.1), the differences observed between different work settings (3.2), and the impact that the cloud and COVID-19 have had on the way we work (3.3).

3.1 List of tasks

The daily work activities performed by our participants on a computer or phone (simply called "tasks") are summarized in Table VI.2. We report only general-purpose tasks, cross-cutting different functions inside the company, and exclude function-specific tasks, e.g., coding, CV-screening, employee advising. Please refer to the Codebook [137] for the complete list. Such a list, emerging from the field, is an important basis to reflect upon sufficiency in everyone's context of flexible work.

When asked "which of the tasks are cloud-based?" (RQ1), our participants actually struggled to find some that did *not* rely on the cloud. As P10 expressed it: "everything is cloud-based in the work that we do". For this reason, the answer to RQ1 is nuanced. All the same, we could identify three categories:

1. **tasks that require constant access to the cloud**, e.g., meeting online (P4) or processing internal requests on an internal tool (P6, P7);
2. **tasks that only need access to the cloud from time to time** to synchronize, e.g., coding using GitHub (P2) or collaborative editing on a shared file (P7); and
3. **tasks that do not use the cloud at all**, e.g., modifying a local file (P1) or having physical meetings (P5).

Some tasks fall either in the first or the second category, depending on the technical solution used. For example, P1 explained that for some email accounts he uses a web version, and for some others, a desktop version.

➢ **Suggestion towards sufficiency**
Strategies to reduce cloud usage could be adopted, where possible, to move from the first to the second category, or from the second to the third one.

3.2 Work settings

Are the work activities different depending on the work setting? The first observation is that all the participants are *mobile* workers: all of them work from home and from the office at least one day a week each (see Table VI.1). They are also mainly *remote* workers: 10 out of 11 have home as their main location (in fact, for Company A, working from home except on Fridays is the common practice).

During the focus groups, the participants were asked to describe their work settings and potential differences between work settings in the tasks they perform. Five work locations emerged: home, the company office, a customer site, the car and the train. Besides, some participants (P8, P9, P10) are mobile between different locations where their company has

Table VI.2: **List of general-purpose digital usages for flexible work** identified by the focus group participants.

Task*	#	Description
email	9	Reading and sending emails
messaging	6	Communicating through instant messaging
planning	5	Making appointment, organizing one's week
online meeting	5	Participating in a meeting through a videoconference tool
phone	4	Communicating through a phone call
reviewing	4	Carefully examining data or documents to find out whether changes or improvements need to be made
project management	4	Organizing a (personal or collective) project with an online tool
data analysis	4	Examining and making sense of data, most of the time with a spreadsheet
preparing presentation	3	Structuring a presentation, creating support material
giving presentation	3	Presenting, remotely or physically, with or without the help of support material
gathering info (internal)	3	Searching and browsing for information on an intranet
gathering info (external)	3	Searching and browsing for information on the Internet
writing time	2	Declaring one's time spent on each project or department, using internal tools
writing documents	2	Producing virtual documents
watching video	1	Watching a video, streamed or not
taking notes	1	Writing down information about something happening, for future notice
online training	1	Learning new skills for a particular job through online resources
online presence	1	Providing a substitute to being in the same room, e.g., with video on and mike only when needed
creating visuals	1	Producing graphs, schemes, flowchart etc. for strategy planning
brainstorming	1	Interacting with other people to suggest many ideas on a topic
attending digital event	1	Participating in an online event, through a videoconference, video-streaming or dedicated tool

*function-specific tasks (e.g., coding, CV-screening, employee advising) are excluded
column #: number of occurrence of the task in the focus groups discussions (decreasing order)
last column: description of the task based on the transcripts and online dictionaries

an office. The overall impression is that *their work locations are completely interchangeable* (P1, P2, P4, P9, P10): "I do exactly the same here than at home" (P2). Small differences remain however, for example the case of transports, as P4 pointed out: "only the case of transport, that's a different situation".

We enumerate the reasons driving the choice of a work location, divided in three themes (3.2.1). And, once at a specific location, specificities of the location in terms of tasks performed (3.2.2).

3.2.1 Reasons to choose a location

The choice of work location can be guided by the task nature, by personal preference or by absence of alternative.

Task driven The most important driver seems to be the task locality: "there are appointments where I just need to be on a certain location" (P1). It can as well be the access to specific gear: "at home office I have a huge screen that's really helpful if you're coding" (P1). P5 also mentioned confidentiality reasons to work from home when she has to deal with employee matters.

Opportunistic driven The reason to go to the office that came back often is the opportunity to see colleagues. For example, some teams in Company B have introduced the "team day": "our team tries to come together and actually see each other once in a while, so we decided that Tuesday was the best day for that" (P5). Otherwise, in this highly remote environment, home is the default (P1, P4, P6). Other reasons are personal attraction to certain locations (P8, P10, P11), or appeal for change of scenery (P9). P5 also mentioned the problem of availability of office space as a reason to stay at home. The opportunistic nature of the choice of location was well summarized by P1:

> "for me flexible work means [...]: being there where you need to be, followed by being there where it is most efficient or most suitable, the easiest place of working" (P1)

No choice Finally, in some cases, there is no choice because there is only one alternative, e.g., for the colleagues of P10 which are on a fully-remote contract.

➢ **Suggestion towards sufficiency**
Awareness of the reasons to choose a location allows planning ahead for quality work in an appropriate environment, thus focusing on the essential, e.g., by grouping together tasks that require the same location.

3.2.2 Choice of tasks driven by location

The participants also revealed differences between work locations, in terms of the tasks they choose to perform there. The most discussed example is the train (differences for other locations can be found in the Codebook). Similarly to the car, the train as a work location stands out because the worker only stays for a limited time and does not have all the usual comfort (space, good Internet connection). On top of that, in a train, one does not want to disturb the other passengers.

These have various implications depending on the participants. Some said they never work from the train (P8). Some reported using the travel time to focus on individual

tasks: answering emails (P1, P5, P10, P11), preparing a presentation (P11), doing data analysis (P10), reading up (P11) or chatting (P11). Finally, some tasks were identified as never done in a train, namely having a meeting (P1, P4, P11), giving a presentation (P1), for obvious reasons of discretion, or screening CVs (P6), for confidentiality reasons.

If the train as a place to work has many disadvantages, one participant reported enjoying working there a lot, being focused and efficient:

> "I take the train at 6:30 and I have those two hours. I'm so productive between 6:30 and 8:30! I do so much! That's why I love it, actually. Because I have no other emails. I just focus on what I have to do. I'm not disturbed." (P11)

➢ Suggestion towards sufficiency

Working from constrained environments allows progressing on individual tasks while reducing one's digital usage to the essential.

3.3 Impact of cloudification

New technologies have profoundly modified the way we work and the pace of work. Along the study, we collected evidence that more recently, the over-availability of work tools brought by the cloud has further reshaped the organization of work. This phenomenon has been amplified by the lockdowns and social distancing measures imposed during the COVID-19 pandemic. Even if we did not specifically ask the participants, this topic often came in the discussions. The participants reported that work has accelerated (3.3.1), has become fragmented (3.3.2) and involves less human contact (3.3.3). They also felt the impact of this new work paradigm on wellbeing (3.3.4).

3.3.1 Acceleration of work

P4 perceived an increase in productivity: "that's a revolutionary time now, how you work. [...] for us it changed a lot the last few years. I think the productivity is higher". Since the pandemic, people started working from home more (P4, P5): "I remember we would come to the office be default at least 4 times a week and now it's more the other way around" (P5). There are more online meetings (P5, P7) and more digital events (P8, P11). P11 mentioned the new common practice of having 'back-to-back meetings' i.e., series of online meetings straight after one another, symptomatic of contemporary way of working.

3.3.2 Fragmentation

The 24/7 availability model results in the workers multitasking a lot:

> "Cloud definitely help in the availability... whether it's people's availability, whether it's availability of data, whether it's the availability of systems that you use. I mean, it's all there all the time, which is kind of also... brings to risk that you are doing everything at the same time all the time, right?" (P7)

As P1 put it, "there is a lot of fragmentation" (P1): the different projects and tasks of the participant are broken in small parts and spread out over the working week, as opposed to having them contained in dedicated days.

3.3.3 Less relationship

Unsurprisingly, a by-product of working more often remotely is to lose human contact. We remind that "seeing colleagues" was an important reason to work from the office for the participants (see 3.2.1). P5 declared trying to do more physical meetings now, to build a relationship with the people she works for. In focus group B2, the second facilitator and P11 pointed out that the informal circuit, where one could quickly ask a question to a colleague, has been lost and needs to be replaced by booked meetings.

3.3.4 Impacts on wellbeing

The perception of the impacts of contemporary way of work on well-being was mixed. On the one hand, participants appreciated the flexibility of work location and work hours, the time saved in transports or the hierarchical differences that became less visible with online interactions (P10). On the other hand, many participants underlined that working remotely is tiring (P4, P6, P8, P11): "it's mentally draining to keep meeting online" (P6). In P4's opinion, it can be stressful, unhealthy or even lead to burn-out. Remote working tends to blurry the personal/professional boundary, as P10 explained: "when you're preparing for dinner or so you're still going on about that last call and think 'oh I need to write this' ". In the end, the post-COVID-19 hybrid system leads to situations where a participant shared having the impression of accumulating the disadvantages of both on-site and remote working:

> "What I find the worst is this mix that I now encounter. There is a lot of things happening on site, in my case. I do have half a day of online meeting followed by half a day of onsite meetings. That's... You have the disadvantages of traveling onsite, and the disadvantages of being in online calls for the other half of the day... So you lose all the advantages." (P1)

4 Findings 2: distinguishing the necessary from the superfluous

In this section, we tackle RQ2. The participants were invited to reflect on their tasks: "Could you do without? If yes, how? If no, why?" Same as for RQ1, the answer to RQ2 is not black or white. Through the focus groups we intended to get an understanding of the reasons why some tasks are perceived as necessary (52 excerpts collected) or superfluous (26 excerpts).

We will only present here the results for the task "online meeting", symptomatic of contemporary work style. Our motivations for this choice are the amount of data collected for this task and the number of hours the participants spend in online meetings (more than 50% of their working time for most of them, sometimes as high as 80%). Besides, online meeting is probably among the most data-intensive of the identified tasks (Table VI.2). As usual, interested readers are invited to look at the online Codebook [137].

4.1 Why do we meet online?

The reasons why online meetings or one of its features are deemed necessary by our participants were clustered in four themes, presented below.

➢ **Suggestion towards sufficiency**
Through the four themes, we give an understanding of the nature of the services rendered by this task, in other words, what to focus on for a sufficient use.

4.1.1 Substitute a physical meeting

The participants have online meetings for the same reasons they would have had an in-person meeting. P6 noted that meetings are an essential part of team-work: "[Company B] is a team-based organization, there is no other way to collaborate than through meetings" (P6). P5 strives to have calls with the people she advises, "to give some personal attention" (P5). Finally, P10 simply wants to keep human contact: "I really need the time to talk to people. So can I do it more efficiently? Probably yes, but then I would lose the click with ..." (P10).

4.1.2 External constraints

Some participants (P5, P8, P11) work with collaborators spread out in the country, sometimes abroad. When they need to meet, they do it by videoconference. Besides that, doing the meeting online can be imposed by the schedule. P1 described a day in which he had three meetings in three different cities and explains that he could not physically have done them without joining some of them online. Finally, a topic that came back several times in the three focus groups is the weight of *expectations* on availability: it is assumed that everyone can drop in an online meeting at any time, so people feel compelled to follow that norm. Overall, the three reasons above result to be experienced as external constraints, on which the participants have no influence.

4.1.3 Convenience

Remote participation in meetings can also be a *deliberate* choice. It allows saving travel time (P1, P4), making it easier to arrange (P6), or not excluding someone that cannot attend in-person (P6).

4.1.4 Camera- and screen-sharing

During the online meeting itself, some features can be enabled or disabled, e.g., the camera. Two participants (P10, P11) expressed a strong feeling: they want to see their interlocutors, all the more since online meetings are becoming a common practice. Camera helps understand nonverbal communication, like facial expressions (P10). Moreover, it was perceived as necessary for important meetings: "If the setting is bigger, with a more formal session, with an agenda and multiple people involved also from other teams, it has to be always with the camera on" (P7). At the same time, P5 admitted that having always the video on is culture-dependent, and noticed a difference in other countries.

4.2 An excess of meetings

Having online meetings is essential in the eyes of the participants, but not all of them are perceived as necessary. In fact, some seem rather useless, as P6 pointed out ironically: "There's a bunch of meetings about meetings and pre calls for the meeting and then different meetings to evaluate the meetings. It's a lot of... yeah... meetings" (P6). Too many meetings, but also meetings that last longer than what they need to: "sometimes I actually need the full half-hour, and sometimes it's done in 15 minutes. So it's surprising how long

the meetings do take in the end" (P5). Several participants (P1, P4, P7) shared having recurring online meetings on their agenda as a replacement of dropping by a colleague when needed. Everyone in the focus groups seemed to relate. This phenomenon of blocking recurring online meetings is also a side effect of the difficulty to find a slot. These meetings are "not driven by any content or agenda" (P7), and end up sometimes being inefficient:

> "In the early days we had corridor meetings. I see someone by the coffee machine and we have a talk. We miss that. So what we do to replace that: we organize meetings to have that kind of conversations. But it's not really efficient because you do it with too many people, and it takes too long" (P5).

Even when a meeting happens physically, an online broadcast may be set up for the people that could not join in person (P5). The norm has become 'online by default', to the point that "you really have to explicitly mention 'this is an *in-person* meeting' because otherwise people would expect that there is a virtual option" (P7). On top of that, some participants noticed having many different clients for the same purpose: "Teams, Skype, Google Meet, Zoom, I forgot the name of the 3 or 4 that I sometimes have to use" (P1). All these applications require an account, to be downloaded, installed, updated.

➢ Suggestion towards sufficiency

While online meetings are essential to the work of the participants, there seems to be substantial room for improvement in finding the *sufficient* quality and amount.

4.3 Addressing the superfluous: tactics towards sufficiency

From the transcripts of the focus groups, we extracted 48 distinguished tactics that emerged from the conversations (some mentioned by the participants as being already applied in their everyday practice of working flexibly, some others as suggestions of future strategies). The tactics were organized into themes. The result is shown in Table VI.3, ordered according to the goal of the tactic itself (written in column 'GOAL').

As illustrated in the table, we identified three ways to classify each tactic:

- **Who should apply it?** Tactics are meant for either humans (Human Oriented - 'HO'), a computing system itself (System Oriented - 'SO'), or for the context where systems and humans operate, e.g., the organization, the environment (Context Oriented - 'CO'). For example, tactic `offline_with_regular_sync` should be applied by humans, while `raise_awareness` aiming to create a sufficiency mindset in the whole organization, must be adopted by the context.

- **Where should it be applied to?** The identified tactics can be applied to either human behavior (see column 'NON TECH'), or technical artifacts (column 'TECH'). For example, tactic `set_mailbox_quota` is clearly a technical feature, while tactic `disable_camera` requires a change in human behavior, hence is non-technical.

- **What types of effects does it have?** The effects of the identified tactics have been found as either visible to humans and addressing the superfluous (hence implementing sufficiency - column 'SUFF'), or transparent to humans and addressing efficiency (column 'EFF'). For example, tactic `self_regulation` removes distractions, hence promotes sufficiency, while tactic `app_aggregator` supports a more efficient work experience.

Table VI.3: **Extracted tactics towards sufficiency**

TACTIC	DEFINITION	GOAL	HO	SO	CO	NON TECH	TECH	SUFF.	EFF.
raise_awareness	Inform, sensibilize, share tips&trick on the issue	adopting sufficiency			✓	✓		✓	
sufficiency_in_company_culture	Integrating digital sufficiency principles in the company culture	adopting sufficiency			✓	✓		✓	
train_for_sufficiency	Explaining sufficiency and how to implement it	adopting sufficiency			✓	✓		✓	
local_if_not_shared	Use local, off-cloud solutions if the work doesn't need to be shared	avoiding overconsumption	✓			✓		✓	
use_local_resources	Using resources (manuals, documents, ..) that are locally available rather than the Internet	avoiding overconsumption	✓	✓		✓		✓	
use_messenging_instead_of_email	Using an instant messaging app instead of sending an email	avoiding overconsumption	✓			✓		✓	
avoid_device_and_data_duplication	Preventing oneself from having several devices with the same purpose or several copies of similar data	avoiding overconsumption	✓			✓		✓	
give_more_freedom_to_employees_to_use_their_devices	Not restricting employees too much in the usage of their professional devices so that they do not need to have separate ones for private use	avoiding overconsumption			✓	✓		✓	
separate_pro_and_perso_environments	Having a way to separate business from private matters on the same device can help use the same device for both	avoiding overconsumption	✓				✓	✓	
respect_do_not_disturb_status	If someone has the status 'do not disturb', not sending messages or calling that person	avoiding social pressure	✓			✓		✓	
respect_blocked_time_in_agenda	If someone has blocked focus time in his/her agenda, not insisting on scheduling a meeting on that slot	avoiding social pressure	✓			✓		✓	
set_mailbox_quotas	Having a (low) quota on the mailbox size	countering data accumulation		✓			✓	✓	
auto-disappearing_messages	Deleting messages automatically after a certain time in messaging apps	countering data accumulation	✓				✓	✓	
data_lifecycle_management	Archiving or deleting old or useless data	countering data accumulation	✓				✓	✓	
shared_agenda	Having a single and shared agenda could avoid a lot of back and forth communication	decreasing useless comm.	✓				✓	✓	
don't_send_unnecessary_email	Preventing oneself from sending an email that is not necessary	decreasing useless comm.	✓			✓		✓	
use_reaction_functionnality_instead_of_a_message	Using the reaction functionality in instant messaging instead of sending a textual answer	decreasing useless comm.	✓			✓		✓	
geographical_separation	Having separate locations for work / non-work	improving quality of life			✓	✓		✓	
integrated_environment	Having all features in a same solution to avoid moving data between the solutions	improving user experience			✓		✓		✓
app_aggregator	Receiving messages coming from different sources in a single app allows everyone to use the solution they prefer	improving user experience	✓				✓		✓
self_regulation	Preventing oneself from being disturbed	increasing focus	✓			✓		✓	
block_focus_time_in_agenda	Planning ahead for periods of uninterrupted work on a specific topic, hence making oneself unavailable to others	increasing focus	✓			✓		✓	
define_channel_for_urgent_matters	Informing collaborators of an way to be reached for urgent matters in order to disconnect from other channels	increasing focus	✓			✓		✓	
disable_notifications	Turning the notifications off	increasing focus	✓			✓		✓	
in_person_meetings	Doing a meeting in person rather than online	increasing focus	✓			✓		✓	
prioritize_notifications	Tuning the notifications to give more priority to urgent matters	increasing focus	✓				✓	✓	
set_do_not_disturb_status	Using the status functionality of messaging app to signal one's availability	increasing focus	✓			✓		✓	
stacking_notifications	Being notified at regular intervals instead of in real time	increasing focus		✓			✓	✓	
use_a_focus_app	Making use of an application helping to organize and keep focus	increasing focus	✓				✓	✓	
use_noise_cancelling_headphones	Using noise-cancelling headphones to help focus	increasing focus	✓			✓			✓
work_in_early_morning	Working at times where one gets less disturbed	increasing focus	✓			✓		✓	
switch_off_client	Closing an application, window or tab	increasing focus	✓			✓		✓	
disable_camera	Turning the video off in an online meeting	lowering tech	✓			✓		✓	
accept_low_tech	Working on oneself to accept tools that are *good enough*	lowering tech	✓			✓		✓	
low_video_quality_by_default_in_meeting	Lowering the video quality in meetings by default	lowering tech	✓				✓	✓	
lower_video_quality_in_meetings	Lowering the video quality in meetings	lowering tech	✓			✓		✓	
replace_cloud_with_phone	Replacing the use of the cloud by phone calls or SMS	lowering tech	✓			✓		✓	
use_pen_and_paper	Using pen and paper instead of a digital tool	lowering tech	✓			✓		✓	
cancel_recurring_meeting_if_no_update	Cancelling the next session of a recurring meeting when it has no purpose	rationalizing meetings	✓			✓		✓	
prepare_before_meeting	Getting ready before a meeting can reduce meeting duration and increase focus	rationalizing meetings	✓			✓		✓	
replace_meeting_by_an_email	Sending an email instead of having a meeting	rationalizing meetings	✓			✓		✓	
shorten_default_meeting_duration	Setting a lower value for the default meeting duration	rationalizing meetings	✓				✓	✓	
space_out_recurring_meeting	Deciding to meet less often with a person or a group one has a recurring meeting with	rationalizing meetings	✓			✓		✓	
online_meeting_saves_travel	Having a meeting online rather than making people travel	saving emissions	✓			✓		✓	
email_attachment_as_link	Enclosing big files as links rather than attachments	saving resources	✓			✓			✓
auto-scaling	Resource management technique to automatically adapt server capacity	saving resources		✓			✓		✓
automatically_killing_unused_applications	Having a smart feature detecting and killing unused applications on one's computer	saving resources		✓			✓		✓
offline_with_regular_sync	Performing the task offline and only synchronizing occasionally with the online version	saving resources	✓			✓		✓	

Abbrv.: HO: Human-Oriented, SO: Software-Oriented, CO: Context-Oriented, SUFF.: sufficiency, EFF.: efficiency

Note that this classification was proposed following discussions between us co-authors, but certain categories could certainly be further debated.

Finally, we observe that the goals of the identified tactics are of two kinds, pursuing a positive outcome (e.g. sufficiency, focus, quality of life) or counteracting a negative one (e.g. overconsumption, social pressure, useless communication).

5 Findings 3: benefits and challenges of going offline

5.1 Services on-demand rather than always on

Towards the end of the focus group sessions, the participants were asked to reflect on their perception of challenges and benefits associated with having cloud services provided *on-demand*, as opposed to being *always on* (RQ3). This would be achieved for example by applying the tactics `offline_with_regular_sync`, `switch_off_notifications` or `use_pen_and_paper`, to only name a few.

On the benefits, three participants (P6, P7, P10) stressed the opportunity to have "more space to focus on a certain task" (P7). Others liked to simply stop being distracted by notifications (P4, P7). Shutting down pieces of software running in the background would also benefit energy consumption (P4).

On the other hand, being partially offline comes with the risk of missing critical emails (P7, P10), not keeping collaborators updated with the latest version (P9, P10), not being reachable (P3, P11) and not benefiting from automatic backup (P9). The difficulty is also that most things in the participants' jobs happen online (P1) and disconnecting momentarily would require to catch up afterward (P11). Finally, P10 admitted not seeing any challenge as long as the on/off feature is controlled by the user, to which P11 added that the feature should also not have impacts on other people.

In the focus group sessions, we only skimmed over the benefits and challenges encountered by the participants, as we wanted them to reflect primarily on their experiences and think about how sufficiency would influence their work habits. In addition to the reflections above, we notice that main benefits and challenges are already expressed in the goal of the identified tactics, for example "improving quality of life". Of course, the tactics bring co-benefits that require further work.

5.2 Barriers to the adoption of digital sufficiency

The biggest challenge of sufficiency is not technical, but rather societal. This study revealed a number of barriers to adopting a more sufficient digital behavior during flexible work. First, reducing user experience to the minimum goes against the most common practice in companies today. P7 pointed out that the use of the cloud has been pushed by his company for years, and that there are discussions about upgrading the laptop cameras for improving the online meeting experience. Moreover, several participants commented on the weight of expectations from the peers. The socio-professional context pushes the workers to always be available and use the latest technologies. We also noticed a low level of awareness on the environmental impacts of digital technologies, both from the participants themselves, but also from the companies that do not show any concern at this point. Finally, while striving to find opportunities for sufficiency, the participants sometimes felt powerless (P1, P5, P9-P11): "I think we're sort of... condemned" (P1). They then tend to turn to efficiency measures, as those seem more attractive:

> "If we see that this is necessary to have the cloud, shouldn't we try to address

> how efficient it is in terms of sustainability? [...] So maybe the source of electricity for instance for the data center, to ensure that they are 100% renewable or decarbonized." (P8).

The three tactics with the goal "adopting sufficiency" are a good way to counteract these barriers. They aim to shift the culture towards promoting quality of work rather than quantity. Doing so, they target the "goals of the system" which is, according to Donella Meadows, among the most powerful leverage points to intervene in a system [138].

6 Discussion

The inputs from the participants of the focus groups allowed us to draw a very rich picture of the interactions between the cloud, flexible work and sufficiency, and the resonance this has in their practice. In this section, we will summarize the main takeaways of the study, then provide additional comments on the tactics towards sufficiency, before discussing the limitations of this work.

6.1 Main takeaways

Compared to our expectations when designing the study, we were surprised by three results:

(i) almost all the tasks performed by the participants are cloud-based,

(ii) their tasks are mostly independent of their work location, and

(iii) workers identified fewer tasks than expected as superfluous.

Results (i) and (ii) showcase an almost total cloud adoption and flexibility inside the two participating companies. We can say that the cloud met its promises of ubiquitous and convenient access to computing resources in these two very digitalized companies. Results (ii) and (iii) challenged our initial idea of looking at tasks one by one, in different work settings, to find ways to address the unnecessary. We also noticed that despite our efforts, it was difficult for the participants to adopt critical thinking. This might be explained by the novelty of the concept of sufficiency, but also the complexity of questioning one's own practices and the system in which they are embedded.

Yet, the study was still successful to uncover a broad list of strategies towards digital sufficiency, that we consider as the main outcome of this study. These are

- the 48 "tactics towards sufficiency" collected through the focus group discussions and reported in Table VI.3, and
- the 5 "suggestions towards sufficiency" (text boxes in this chapter) that we, authors, propose in light of the results.

6.2 On the tactics towards sufficiency

First, we want to remind that the list in Table VI.3 is an initial list, emerging from the three focus groups and our analysis. More work would be required to consolidate that list, and the proposed categories. Nevertheless, we make the following observations:

Category "who should apply it?" Most tactics (38 out of 48) are *human-oriented*, and only 5 are *software-oriented*. This is not surprising, as participants were invited to think in terms of their individual habits. Still, 6 tactics are *context-oriented*. It could indicate that practitioners are adopting a more systemic mindset by thinking in terms of the needed shift in the organizational culture and the global ecologic implications, which is a promising result.

Category "where should it be applied to?" 14 tactics are technical, which means that they modify the operation of systems (software, hardware), and the 34 others apply to humans, by requiring a change in behavior. This shows that digital sufficiency is mainly non-technical (all the more if we notice that 4 of the technical tactics are actually *efficiency*, and not sufficiency tactics, see paragraph below). But again, the results can be biased since we asked the participants to reason mainly on their personal experience ("could YOU do without?").

Category "what types of effects does it have?" Most of the identified tactics implement sufficiency, which was the goal of the study. All the same, 6 of them are efficiency measures, which are relatively irrelevant to our study, and which could have been removed from the list. We still decided to keep them, because they show that there is a confusion between efficiency and sufficiency among the participants, but also because it is sometimes difficult to clearly decide between the two. For example, "automatically killing unused applications" is without hesitation an efficiency measure, since it optimizes the systems while being transparent to users. But "enclosing big files as links rather than attachments" can be a sufficiency measure if the users have to create the link themselves, or an efficiency measure if this is done automatically (which was the case in the context of the focus group, hence the classification). Since the concept of sufficiency is quite new in the literature, we would be interested in other opinions on this classification, i.e., if all the tactics classified as such are truly sufficiency tactics.

6.3 Comparison with digital sufficiency definition

Coming back to the original definition of digital sufficiency by Santarius et al. [8] (see Chapter I, Part 3.2), we note that our tactics mostly concern one stakeholder: the private user. It is far from the many other actors they take into consideration, namely producer/developer, seller/provider, organizational user, policy regulator and civil society. Nonetheless, we believe that the identification of the tasks and the questioning of the essential needs of end-users is the starting point for the consideration of sufficiency by the rest of the system. Besides, the identified tactics successfully encompass three of the four dimensions of digital sufficiency (see Table VI.4).

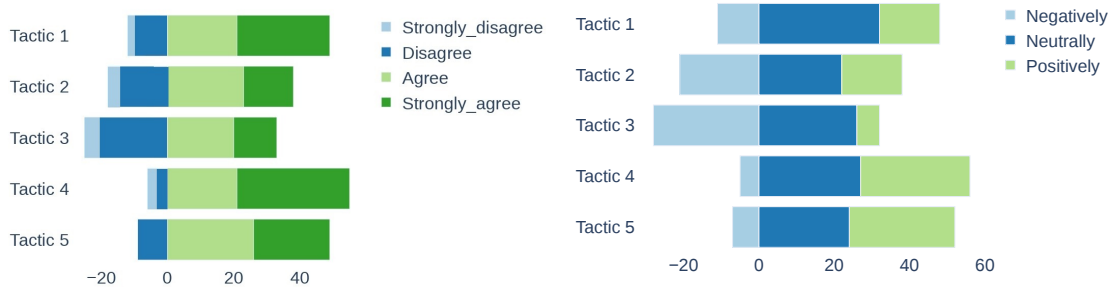
6.4 Quantifying the impact of tactics

So far, one important information is missing from our collection of tactics: there is no indication of their actual potential to decrease environmental damage or their impact on the people applying them. This prevents the list from being actionable, since we would not know which tactics to prioritize. This problem arises from the fact that our study is only qualitative, and not quantitative.

In order to bridge this gap, Patricia Lago and I supervised Maliha Nawshin Rahman, a student doing her master's thesis on the topic. We provide below a few results issued from her work [139], that was accepted for publication at the conference ICT4S 2024 [C5].

Table VI.4: **Comparison of our tactic classification** (Table VI.3, column TC) **with the digital sufficiency dimensions** [8]

Digital sufficiency dimension: *definition* [8]	**TC**
User sufficiency: *users apply digital devices frugally and make use of ICT in a way that fosters sufficiency oriented lifestyles*	Human Oriented
Hardware sufficiency: *producing and designing hardware for longevity, repairability, and with the least possible resource and energy demand*	NA
Software sufficiency: *software development and implementation that ensures long-term functionality and the lowest possible data traffic and hardware utilization for task performance*	System Oriented
Economic sufficiency: *ICT-borne improvements are used to nurture public and common good instead of economic growth*	Context Oriented



(a) "I would consider applying each of these tactics in my daily work life"

(b) "How would each of these tactics affect your daily work productivity?"

Figure VI.1: **Preference on applying tactics, and effect on work productivity** (questionnaire survey with 61 responses) [139]
Tactic 1: using off-cloud version of an application if the work doesn't need to be shared,
Tactic 2: performing a task locally with regular cloud synchronization,
Tactic 3: turning off camera or lowering video quality in meetings,
Tactic 4: enclosing email attachment as link,
Tactic 5: closing an application, window or tab when it is not needed anymore.

Impact on users First, we surveyed practitioners on their perception of tactics, through a questionnaire survey. We collected answers from 61 participants, coming from different professional sectors including IT, Consultancy, Marketing, Academia and Finance. A majority of them (42 out of 61) declare following a "hybrid" work practice, working both remotely and on site.

The questionnaire results on a subset of five tactics are reported in Figure VI.1. It is interesting to note that theirs answers to the first (Figure VI.1a) and the second (Figure VI.1b) questions are very similar. They follow the same pattern of 'strongly disagree' / 'disagree' versus 'agree' / 'strongly agree' (respectively 'negatively' versus 'neutrally' / 'positively'). This suggests that willingness to adopt the tactics and perception of how they will affect work productivity are correlated. In other words, *practitioners would consider applying tactics when they do not affect their work productivity.* For this subset of tactics, the most popular is Tactic 4, followed by Tactics 1, 5, 2 and 3.

Table VI.5: **Impacts of applying the tactics.** Preliminary attempt to quantify, with the help of the literature.

	Energy saved		Saved	Ref.
	Client-side	Server-side	data traffic	
Tactic 1	0.3-1W*	0.25W**	all	[140]
Tactic 2	0.3-1W*	∅	2-3 orders of magnitude***	[140]
Tactic 3	4W†	∅	∅	[141]
Tactic 4	∼0	?	∼0	rough estimate
Tactic 5	no statistical evidence	∅	1 order of magnitude	our measurments [139]

∅ indicates "out of scope"
*word processing in google drive, difference between power consumption of netbook in offline and online edition (Table VI [140])
**assumption used in the article (Section IV.D.6 [140])
***slopes in Figure 4 [140]
† mean values in Table 7 [141], converted to Watts

Impact on energy consumption In parallel, we tried to quantify the impact of applying the tactics on energy consumption. For this, we used data from the literature and performed direct measurements. The results are reported in Table VI.5. We found that the literature was very scarce on the topic, and that it is very difficult to find comparable figures, since energy consumption vary greatly with the type of hardware used. Tactics may allow energy savings on three tiers: (i) the device (e.g., laptop, tablet), (ii) the network (e.g., router, core network) and (iii) the servers on which the cloud services are located. Data on client-side are the easiest to obtain, through direct measurements (watt-meter or software tools like Intel RAPL). Network energy consumption can not be directly measured in most cases. However, it is possible to measure data traffic as a proxy. Finally, server-side information is extremely hard to obtain, as most cloud services are proprietary and function as a black box. We have to rely on rough estimates. All these reasons explain the many missing data in Table VI.5, which should therefore be handled with precaution.

From the data we have, it looks like Tactic 3 (switching off the camera) has the most impact, followed by Tactics 1 and 2 (local VS cloud-interactive). Tactics 4 (email attachment) and 5 (closing tabs) come last. Interestingly, this order is almost opposite to the impact on users as reported in Figure VI.1: they ranked Tactic 4 highest, followed by Tactics 1 and 5. This is an indication that, on this subset of tactics, there is no "free lunch": the bigger the energy savings, the more efforts it requires from the practitioners. We also noticed that practitioners have a wrong intuition on the energy saving potential of each tactic, which was a question included in the survey, but not reported here. They ranked Tactics 4, followed by 3 and 5 as "most beneficial to reduce energy or resource consumption", which is quite different from the results above. This misconception was observed in a similar study [83], and is probably due to a mix of factor: intangibility of digital footprint, constantly evolving industry making it difficult to be up-to-date, perceived sacrifice of each tactic, skepticism toward change, etc.

7 Limitations

In this section, we come back to the main study presented in this chapter, and point out its threats to validity.

We ensure the anonymity of our participants, their companies, and their collaborators. Hence, we keep their identifying details confidential, under the human ethics guidelines governing this study. Accordingly, the verifiability of our results should derive from the soundness of the research method. We therefore describe in Section 2 the study design we followed throughout our investigation, and transparently provided all the material we could disclose in a replication package (see Section 2.5). In addition, we reference in our findings direct quotes from our participants as much as possible. Despite our best efforts, we are aware of potential limitations to the validity of our results, that we discuss below.

Misinterpretations As this is a qualitative study involving humans, there is a risk of misunderstanding from the participants, and misinterpretations from the researchers. To mitigate this threat, we designed the study upfront, and run a dry-run session with colleagues to uncover possible issues. We provided introductory material in the focus groups to explain motivation and context, and to align the participants to a common understanding and terminology. We also embedded a small exercise for the participants to get acquainted with the concept of sufficiency in their own everyday lives.

Moderator bias During the focus group sessions, the moderator is in charge of presenting the study and asking the questions, which comes with a risk of introducing bias. To mitigate this threat, special care was taken upfront to design questions as open and neutral as possible. During the sessions, we involved a third researcher who would support the moderator in overcoming the subjectivity. They adopted a neutral stance, avoiding to use targeted questions or to introduce bias by giving examples. In addition, to ensure that no important information went unaccounted for, we ended each focus group session with an open-ended question. There, participants could add anything they forgot, and address open remarks.

Data analysis To ensure the quality and reliability of data analysis, the focus group transcripts were analyzed and coded first by a researcher, then critically reviewed by the second. Any inconsistencies were discussed to check data interpretation, reduce biases in analyzing the data, and formulate the results.

Generalizability This is an exploratory study and, as such, we do not claim generalizability. Rather, we intend to build initial knowledge on how digital sufficiency can be pursued. However, we recognize three possible validity threats, here, namely (i) the relatively small quantity of data collected (3 focus groups, 11 participants), making it possible to miss a major trend; (ii) the fact that the study was carried out in the Netherlands, a country that is very advanced in both digitalization and flexible work; and (iii) the type of work performed by our participants, mostly work on computers, potentially overrepresenting their flexibility and dependence on the cloud. For these reasons, our results cannot be considered generalizable to all types of work and contexts. However, it could serve as a starting point for further works to build upon.

8 Conclusion

In this chapter, we expose a qualitative study exploring the question *"how much and which cloud usage is sufficient?"* in the context of flexible work. We present data collected by running three focus groups in two companies in the Netherlands (11 participants), following a thematic analysis. The results provide a preliminary picture of the nature of our digital professional needs, perceived benefits and challenges if we choose to go offline sometimes, along with a first list of concrete actions to focus on the essential while limiting environmental footprint (the "tactics towards sufficiency"). Overall, the study shows that our participants are "always *on*": the cloud dominates their professional activities. The notion of digital sufficiency resonates a lot with them, even though they need concrete suggestions.

Link with the sufficiency behaviors As mentioned in the chapter introduction, the context of this study differs from the rest of the contributions of this PhD. The first difference is the scope: here, we focus on *indirect users* of data centers (in the sense of the definition in II.1.1.1) rather than direct users in the other chapters. The second difference is the trigger for sufficiency behavior. In the previous chapters, we always studied external events as a trigger for user action: a peek electricity consumption or a phase of low renewable production. Here, we study sufficiency *in the absolute* and not in response to any particular event. We believe that this vision it more faithful to the original concept of sufficiency and digital sufficiency (see I.3), i.e., a voluntary reduction of one's consumption of IT with the goal to reduce its adverse environmental effects.

Ideally, we would like to link the sufficiency tactics of indirect users to the corresponding sufficiency behaviors of direct users, following the chain of intermediaries between end-users and the actual hardware. This would enable to test their potential on real-life scenarios thanks to data center simulations. However, given the challenges of quantifying the impact of user actions, due the lack of scientific literature and the complexity of software stacks (see out attempt in 6.4), it would be difficult, at this stage, to achieve such a mapping.

All the same, here is preliminary attempt. We can say that all the tactics from Table VI.3 with goal "avoiding overconsumption", "countering data accumulation", "decreasing useless communication", "lowering tech", "rationalizing meetings" and "saving resources" are suitable to be linked with the sufficiency behaviors. A few examples:

- "Turning the video off in an online meeting" is a lever of degradation, probably Space Degrad since the meeting duration remains the same.
- "Setting a lower value for the default meeting duration" would then be a lever of Time Degrad.
- "Cancelling the next session of a recurring meeting when it has no purpose" is a Renounce behavior.
- Finally, "deciding to meet less often with a person or a group one has a recurring meeting with" can be considered as Delay.

We wish for more efforts in quantification of online behaviors, through more academic studies, data transparency and automatic reporting, for our decisions and recommendations to be guided by accurate data.

Conclusion and perspectives

Agriculture feeds, and without food, we die — even with virtual reality goggles.

Fabrice Flipo [70]

This chapter concludes the work performed during this PhD thesis and attempts to provide an answer to the questions raised in Introduction. We suggest thereafter perspectives for future works on the topic, trying to order them roughly in order of difficulty. We start with short-term perspectives, that could be done with relatively low effort thanks to the methods and implementations developed in this thesis, and finish with longer-term perspectives that would require more significant efforts in modelling and implementation.

9 Conclusion

The use of data centers is increasing globally, and so is their environmental impact, despite continuous hardware and software improvements. This is due to the rebound effect, which tends to transform efficiency gains into increased demand. To counteract this effect, scientists and international bodies are starting to consider measures of *sufficiency*, aiming at decreasing the total environmental impacts rather than the relative impacts per unit of good or service. Sufficiency can be applied to all sectors of the industry, and has been declined to IT under the term "digital sufficiency". It invites us to rethink our use of technologies to what is deemed *enough* in regard to the planetary boundaries, and involves all stakeholders along the supply chain. In this thesis, we focused on the "user sufficiency" dimension of digital sufficiency, applied to data centers.

Q1: How to accurately model the interaction between direct users and the data center? Most works in the research field of Distributed Systems do not model the users directly. Users are rather taken into account through various performance metrics. In contrast, we propose a model for data center users submitting jobs to the infrastructure. Their jobs can be replayed from real-world traces or generated from a mathematical model. The originality of our model is that it allows accounting for *user reaction in response to internal or external events*. In the works presented, we focus on external events linked to the availability of electricity and internal events linked to the performance of previously submitted jobs, but more types of events could be envisioned.

Our model is implemented in Batmen, an open-source plugin for the scientific simulator Batsim that I developed during this thesis. Batmen use workload inputs which are standard in the community, splitting them per user and submitting the jobs dynamically under the course of the simulation. This way, it enables a more fine-grain representation of user behaviors.

In particular, we propose, implement and study a method of replay with feedback. Compared to traditional replay, simulated users wait for the termination of their previous jobs before submitting the new ones, just as real users would. Our method still comprises some limitations and would need to be scientifically validated, which proved to be out of scope for this thesis. Nevertheless, we discuss the methods that could be used for such a validation and argue that this contribution remains an important step towards more accurate simulations of distributed systems.

Q2: Which "sufficiency behaviors" can be adopted by direct data center users, and how does user effort translate into footprint reduction? We define "sufficiency behaviors" for direct data center users as voluntary modifications of the characteristics of their job submissions in order to decrease the demand in the data center. We propose five such behaviors, namely delaying (Delay) or renouncing (Renounce) the job submission, reconfiguring its resource request (Reconfig), degrading it spatially (Space Degrad) or temporally (Time Degrad). A mathematical representation of these behaviors is provided according to our data center model.

The sufficiency behaviors are then characterized individually through simulation on real-world inputs. The results allow us to classify them according to their energy saving potential, impact on scheduling metrics and effort required from users (see Table III.2). Renounce is unsurprisingly the most efficient behavior, with 16% energy savings on average on a 4-hour time window, if applied by all users during this time window. However, it is also the behavior that asks the biggest sacrifice to users. At the other end of the spectrum, Delay and Reconfig require the least effort from users. Delay is very adapted for short-term gains, but leads to a peak of submission in the future which significantly affects the energy and scheduling metrics then. On the long term, Reconfig is almost a zero-sum game. Finally, the two Degrad behaviors yield interesting results, as they cut off computing load. Time Degrad is not adapted in the short term, but it consistently lightens the load in the future. This characterization also reveals the inertia of sufficiency behaviors, that do not act on the jobs already running in the infrastructure.

We investigate thereafter the behaviors' usefulness in the context of renewable energy management. A three-state energy feedback mechanism is introduced to inform the users on the status of electricity production: green when production is abundant, red when production is low, and yellow in-between. Thanks to a reproducible experimental campaign with real-world inputs, we show that the approach is conclusive and allows reducing brown energy consumption. For example, if users accept to apply the sufficiency behaviors on 50% of their job submissions at periods of low production, brown energy can be reduced by 8% overall. The energy savings are linear with the effort.

Q3: What are the opportunities for digital sufficiency in cloud usage? Since sufficiency entails rethinking our basic human needs, we believe it is essential to involve the social sciences in any work on this subject. That is why we decided to carry out a qualitative study, and interview practitioners about their digital needs. Overall, our study reveals that our participants are "always *on*": the cloud dominates their professional activities. All the same, we extracted from the discussions a list of 48 so-called "tactics towards sufficiency" in cloud usage for work, which can be seen as the counterparts of the sufficiency behaviors for indirect data center users. The tactics are concrete actions to limit one's consumption of IT, and bring co-benefits such as improved focus or work-life balance. The notion of digital sufficiency resonates a lot with our participants, even though they need concrete suggestions.

10 Short-term perspectives

Evaluation of the sufficiency behaviors with replay with feedback In Chapter V, we introduce the method of replay with feedback. I argue that this method leads to more realistic results than rigid replay and should therefore be used generally for all simulation work in HPC. Using this method is even more important in studies focusing on user behaviors, like ours. Consequently, the experimental campaigns presented in Chapters III and IV would gain from being rerun in combination with the feedback model. This would allow getting rid of some unrealistic behaviors observed in the results, like the burst of submission at the end of the demand response window with behavior Delay. To understand, we illustrate the effect of Delay behavior with and without the feedback replay model in Figure A.1. Without feedback model, a simulated user would submit all her jobs at the end of the window, even if the jobs had dependencies between them. With the feedback model, this user would better follow the original submission pattern.
➢ *implementation: the sufficiency behaviors from Chapter III with the feedback replay model are already implemented and tested in the latest version of Batmen*[4]*. One would only have to adapt and relaunch the reproducible experimental campaign.*

Unfortunately, since our replay with feedback model still contains some limitations and did not undergo rigorous scientific validation (see corresponding long-term perspective further down), it would be difficult to justify why we use it instead of the widespread rigid replay method, and which particular parameters we should choose.

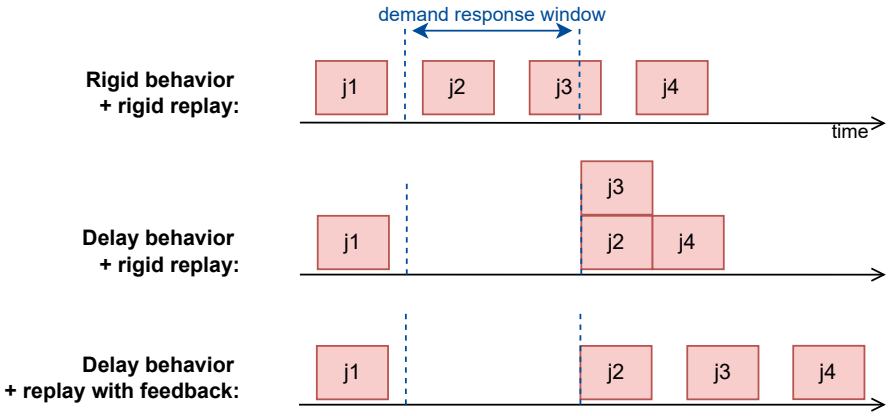


Figure A.1: **Illustration of the effect of the replay method on sufficiency behavior Delay.**

Studying other behaviors To the best of our knowledge, our contribution is, to date, the one studying the most user levers together. However, there are still other behaviors that could be included. First of all, the sufficiency behaviors presented in this thesis could be made parametric. In their current form:

- Delay and See You Later postpone the submission to the end of the demand response window, or by $\Delta t = 1$ hour, respectively;
- Time Degrad multiplies the job execution time by a scaling factor $x_T = 0.5$;
- Space Degrad and Reconfig apply a scaling factor $x_S = x_R = 0.5$ to the number of resources required, and do not take speedup into account.

[4] see Batmen commit ea7fd480

The values of these parameters Δt, x_T, x_S and x_R have been chosen to be as simple as possible for this initial exploration of sufficiency behaviors. Future works could consider a distribution of them and/or make them user-dependent.
➢ *implementation: everything is ready to allow the modification of these parameters, including the speedup model.*

Another behavior of interest, not studied in this thesis, is to allow users to checkpoint their running jobs. We could imagine that sustainability-concerned users would accept to stop some of their jobs, for example when renewable energy is scarce, and resume them later. This would allow the potential energy savings to go beyond the "fluid mass" introduced in Chapter III.
➢ *implementation: it would require to create a new user behavior, and implement preemption in Batsim, for example by killing the current job and dynamically resubmitting another one with the remaining computing load.*

Information provided to users In our model, data center users take action in response to specific feedback. In Chapter III, the trigger for sufficiency behavior is being in a demand response window. In Chapter IV, we use the three-state energy feedback mechanism, defined on energy production. While these models have the advantage of being simple and actionable, they might not be the best suited depending on the objective. For example, if the objective is to minimize brown energy consumption, i.e., the energy that does not come from renewable sources, energy states could rather be defined on *instantaneous power consumption compared to production*, with red states when consumption is above production, green state when production is in excess and yellow states in-between.
➢ *implementation: the energy state cannot be computed before the simulation as it is done currently. We would need to have access to the platform's instantaneous power consumption*[5] *and calculate the energy state each time a user wants to submit a job.*

Additionally, the model could include forecasts to *anticipate* demand response windows or energy shortages. For example, yellow states could be used to that end, as a pre-warning before a red state. Users would be encouraged to submit jobs that would finish before the red state, in order to limit energy consumption in that period.
➢ *implementation: we would need a model for production forecasting, such as the API* https://solcast.com/

Leveraging user behaviors inside the scheduler Throughout this thesis, we focused on user behaviors alone as a lever for energy saving. I believe that there is potential in taking these efforts into account also inside the scheduler. For example, allowing the scheduler to kill jobs, checkpoint them, or suspend the waiting queue would make it possible to go beyond the fluid-residual limit, as discussed before. Other user-scheduler collaborations can include malleable applications [142], predictive server shutdown, DVFS etc. In this case, decisions could be taken on behalf of the users, with a mechanism of contract with the data center operator specifying the degradation they are willing to accept (see related work on green SLA in Chapter I Part 4.1). Nevertheless, we argue that the users should be informed of these decisions and that they should be involved as much as possible, otherwise the techniques become once again efficiency (and not *sufficiency*) measures.

Quantifying the impact of tactics and linking with sufficiency behaviors In Chapter VI, we collect many "tactics towards sufficiency" in the context of cloud usage for

[5] see QUERY event in Batsim protocol https://batsim.readthedocs.io/en/latest/protocol.html#query

flexible work. It is a qualitative study, intended to create knowledge on the topic. We briefly introduced the work of Maliha Nawshin Rahman who started to quantify some of them. Future work could create evidence of the actual benefits for digital sufficiency of applying the tactics, i.e., apply them and measure their impact on resource efficiency, productivity and flexibility. In addition, the link between them and the sufficiency behaviors could be made explicit, in order to also test their potential with data center simulations.

11 Long-term perspectives

Validating the model of replay with feedback In my opinion, one of the most useful and promising research direction would be to carry out a scientific validation of the model of replay with feedback, and answer the question "are simulations with replay with feedback more realistic than simulations with rigid replay?" We already provided hints on how such a validation could be done in Part V.7.5. For now, only some isolated works have used a feedback model in performance evaluation, for example Klusáček et al. with the simulator Alea 4 based on GridSim [126] or Schlagkamp in generative simulations [123]. Would the method be successfully validated, it would allow it to be used more widely in the community, or even become standard for performance evaluation, hopefully helping to reach more realistic conclusions.

Note that the best version of the replay with feedback method is probably not the one presented here, and future work is needed to study other replay models that would:

- account for day/night and weekday/weekend variability of submission,
- consider a user as a new user after a long think time,
- include a model of arrival/departure of users,
- account for other user response to feedback, like change in requested resources.

The work of Feitelson et al. is once again essential in that endeavor [143], as well as, we hope, the open-source tools and metrics that we developed in this thesis.

Going beyond energy In this thesis, the experiments studying the sufficiency behaviors look *in fine* almost only at energy metrics, despite our endeavors at the beginning of the manuscript to explain higher-order effects and other impacts indicators. This is due to the low maturity of the community on the topic leading to the absence of methods and metrics to expand the scope. All the same, we tried to include a discussion on "acceptability" in Chapter III, user effort metrics in Chapter IV and willingness to adopt / impact on productivity in Chapter VI. Future research is needed to include a LCA dimension on the data center as a whole, in order to investigate whether sufficiency behaviors are indeed effective to slow down the growth of demand and the building of new infrastructures associated to it.

We would also like to see studies looking at the actual effect of eco-feedback (like the three-state energy model) on data center users to trigger action. This would allow to empirically define the proportion of them who are ready to renounce, degrade, etc. This requires to conduct quantitative surveys among users and has been done for example in the software engineering field [87].

Collective dimension of sufficiency In this thesis, we study human behaviors in relation to data centers and their environmental impact. We would like to point out that our

intention is *not* to demonize all data center or cloud usage. These paradigms offer powerful opportunities for a low-carbon transition as long as their use is controlled, and maybe limited to certain sectors. We are *not* arguing either that all the burden for a sustainable transition lies with the individual, which would be reductive [144]. On the contrary, as noted before, digital sufficiency involves many more stakeholders, and we consider our work as a first step in that direction. Besides, we faced during the focus groups in Chapter VI the question of the *agency* of IT users, i.e., their capacity to deliberately initiate actions to influence the course of events [145]. We could see that their agency is constrained for many reasons (personal, professional, cultural, societal, etc.).

As a result, sufficiency must be thought as a *collective* effort. Future works on sufficiency in data centers should consider more stakeholders than the mere end-users. We want to understand how companies, cooperatives, associations, local governments etc. can foster the adoption of sufficient lifestyles. This can only be achieved by involving scientists from other disciplines such as Sociology, Economy or Political Sciences.

Reorienting research in Computer Science I will conclude with a last and more personal perspective. As I already had the occasion to recall, the state of the environmental crisis is alarming. Yet, public and private research in Computer Science continue to accelerate, bringing new gadgets over and over to the market. Digital technologies are great tools for optimization, but they do not choose their use cases. They can optimize indifferently public transportation schemes than oil and gas extraction [146]. As researcher Vlad Coroama puts it:

> *Digitalization is a **lubricant** of societies and economies. But **friction** is good from an environmental perspective.*[6]

I think it is important to take a step back and ask ourselves what kind of world current research is leading us into. Luckily, these reflections are starting to take place, for example within the research communities ICT4S[7] or EcoInfo[8], but also at a bigger scale, as demonstrated by the organization of the "eco-responsible computer sciences" conference at the headquarters of the French research organization CNRS[9]. In France, the group "reflection" of the Labo1point5 initiative[10] tackles this issue head-on by proposing that, in addition to measuring the direct footprint of our research laboratories, we reflect on the very subjects of our research and their relevance to current crises. I am involved in the think tank related to Computer Science, and we are currently rolling out a campaign of semi-structured interview of computer science academics to collect their ideas and perceptions about what should (and should not) research focus on.

Countries have been racing to build the first exaFLOPS supercomputer, companies are rushing into artificial intelligence, the Internet of Things, the metaverse... Are these the top priorities in a world with limited resources and ever-increasing inequalities? I don't think so. On the contrary, I believe that the solutions to the environmental crisis are already there, and they are *not* high-tech. They are simple and local, maybe not optimal, but made in a way that creates human links. The only thing missing is the real political will to use them.

[6] in a talk about his paper [6] at ICT4S 2021 summer school

[7] https://conf.researchr.org/home/ict4s-2023

[8] https://ecoinfo.cnrs.fr/

[9] program (in French): https://www.ins2i.cnrs.fr/fr/cnrsinfo/conference-les-sciences-informatiques-ecoresponsables

[10] https://labos1point5.org/

Publications and communications

Peer-reviewed journals

[J1] Maël Madon, Georges Da Costa, and Jean-Marc Pierson. "Replay with Feedback: How Does the Performance of HPC System Impact User Submission Behavior?" In: *Future Generation Computer Systems* 155 (Jan. 2024), pp. 66–79. ISSN: 0167-739X. DOI: 10.1016/j.future.2024.01.024.

Peer-reviewed conferences and workshops

[C1] Maël Madon, Georges Da Costa, and Jean-Marc Pierson. "Characterization of Different User Behaviors for Demand Response in Data Centers". In: *Euro-Par 2022: Parallel Processing*. Ed. by José Cano and Phil Trinder. Lecture Notes in Computer Science. Cham: Springer International Publishing, Aug. 2022, pp. 53–68. ISBN: 978-3-031-12597-3. DOI: 10.1007/978-3-031-12597-3_4.

[C2] Klervie Toczé, Maël Madon, Muriel Garcia, and Patricia Lago. "The Dark Side of Cloud and Edge Computing: An Exploratory Study". In: *Computing within Limits*. LIMITS, June 2022. DOI: 10.21428/bf6fb269.9422c084.

[C3] Maël Madon and Patricia Lago. ""We Are Always on, Is That Really Necessary?" Exploring the Path to Digital Sufficiency in Flexible Work". In: *ICT4S 2023: International Conference on ICT for Sustainability*. Rennes, France: IEEE, June 2023, p. 11. DOI: 10.1109/ICT4S58814.2023.00012.

[C4] Jolyne Gatt, Maël Madon, and Georges Da Costa. "Digital Sufficiency Behaviors to Deal with Intermittent Energy Sources in a Data Center". In: *ICT4S 2024: International Conference on ICT for Sustainability*. Stockholm, Sweden, June 2024.

[C5] Maliha Nawshin Rahman, Maël Madon, and Patricia Lago. "Sufficient Use of the Cloud for Work: Practitioners' Perception and Potential for Energy Saving". In: *ICT4S 2024: International Conference on ICT for Sustainability*. Stockholm, Sweden, June 2024.

Communications

- 27 March 24: *Comportements de sobriété pour des utilisateurices d'un data center alimenté aux énergies renouvelables.* Presentation at GreenDays2024, Toulouse, France. http://perso.ens-lyon.fr/laurent.lefevre/greendaystoulouse2024

- 7 March 24: *Sobriété numérique dans les data centers.* Presentation at ROADEF'2024, Amiens, France. https://roadef2024.sciencesconf.org/

- 8 Dec 23: *Numérique Eco-Responsable*. Seminar at Histoire Humanités Numériques Informatique, Toulouse, France. https://hhn.hypotheses.org/programme-2023-2024
- 27 Nov 23: *J'ai orienté ma recherche vers les sciences informatiques écoresponsables*. Panel discussion at the conference "les sciences informatiques écoresponsables" organized by CNRS Sciences Informatiques, Paris, France. https://www.ins2i.cnrs.fr/fr/cnrsinfo/conference-les-sciences-informatiques-ecoresponsables
- 4 Jul 23: *Involving the users to mitigate the environmental impact of data centers*. Remote guest speaker at EIT Summer School, Rennes, France (online). https://numerinnov-hub.eu/universite-de-rennes-summer-schools-2023/
- 28 Mar 23: *Vers une utilisation sobre des centres de données : les sciences sociales à la rescousse de l'informatique*. Presentation at GreenDays'2023, Lyon, France. http://perso.ens-lyon.fr/laurent.lefevre/greendayslyon2023/programme_greendays2023.html
- 16 Feb 23: *Towards sufficient use of data centers: simulation work and qualitative research*. Seminar at Low carbon and sustainable computing seminar series, Glasgow, UK (online). https://www.gla.ac.uk/schools/computing/research/researchthemes/lowcarbon/
- 27 Jan 23: *Vers des simulations HPC plus réalistes : rejouer le comportement de soumission utilisateur*. Presentation at Journées non thématiques du GDR RSD, Lyon, France. https://gdr-rsd.fr/journees2023/
- 23 Nov 22: *Involving the users to mitigate the environmental impact of data centers*. Guest presentation at Research Cocktail series at VU Amsterdam, Amsterdam, The Netherlands.
- 8 Apr 22: *Characterization of user behaviors for demand response in data centers*. Seminar at DataMove INRIA research team, Grenoble, France (online). https://team.inria.fr/datamove/talks/
- 17 Jun 21: *From the embodied emissions of radio base stations to involving the user in environmental-aware clouds*. PhD seminar at Société Informatique de France, online. https://sifdoctorants21.sciencesconf.org/

Postface

Statistics on writing this manuscript

Writing a PhD manuscript is a long journey. For me, it was undoubtedly the stage of my thesis that I was the most apprehensive about. The document being written in LaTeX, I used my usual programming environment: VSCodium and useful extensions as IDE (Integrated Development Environment) and Git for versioning and organizing my writing.

The git repository `maelPhD` was created on October 9, 2023. As I write these lines, it counts 154 commits (average 2.4 commits per active day, 0.9 per all days) by one author, 19 tags and 9872 lines of code[11]. Figure A.2 plots the evolution of the number of added lines of code in the repository. It can be taken as proxy of the writing activity. We notice several peaks: October 13, December 15 and January 29. They correspond to commits where I copied the sources from the corresponding article onto the manuscript repository. We can also see three periods of no activity: Nov 1–Nov 16, Dec 25–Jan 11 (Christmas break) and Mar 12–Apr 2 (after the manuscript submission). Overall, we notice a relatively slow start, with 6500 lines of code added in the first four months, followed by a steady writing period in late January and February.

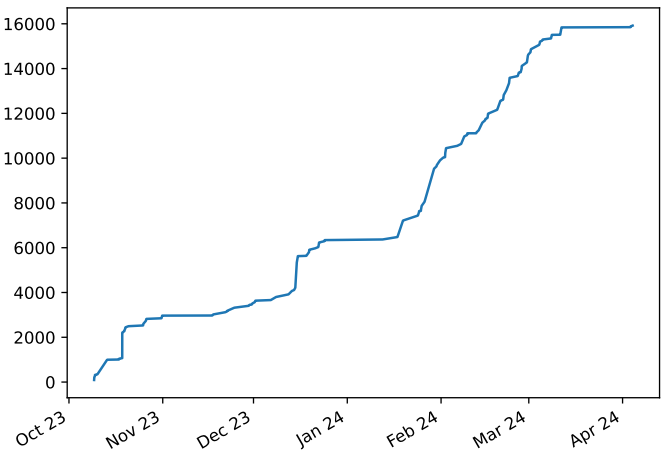


Figure A.2: **Cumulated added lines of code in `maelPhD` git repository**

The distribution of commits per day of the week and hour of day is plotted in Figure A.3. Apart from a few exceptions, I am happy to realize that I managed to stick to common working hours during this period — at least as far as commit times are concerned ;-).

[11] the statistics and graphs shown here are generated with GitStats https://gitstats.sourceforge.net/

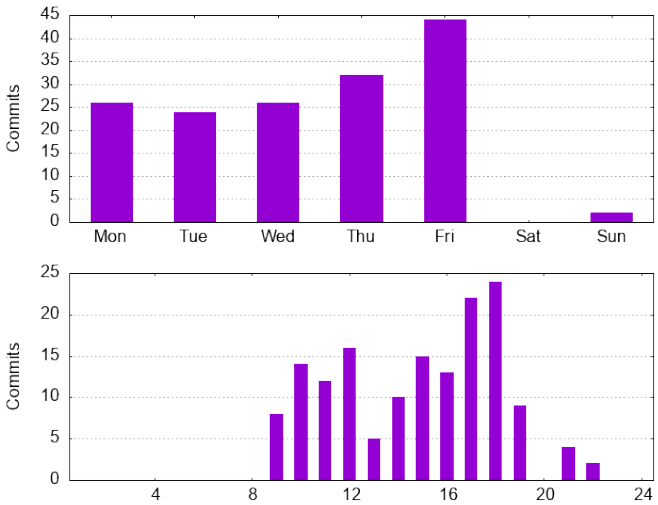


Figure A.3: **Number of commits per day of the week** (top) **and hour of day** (bottom)

Carbon footprint of this thesis

Like any human activity, performing the work presented in this thesis involves the consumption of natural resources and the emission of greenhouse gases. During my PhD, I was involved in the "ecological transition group" of my laboratory[12] in which we endeavored to estimate the carbon footprint of our research activities and build actionable scenarios to reduce it. Our approach is part of the wider Labo1point5 initiative[13], which has supported us in this process.

For the sake of transparency, I try to estimate below all the carbon emissions attributable to this PhD:

- **Office work**: For three years, I used an office in Toulouse, a laptop, a screen etc. To estimate the associated impacts, I use my laboratory's reference carbon footprint: 2.1 tCO_2e per person per year[14].
  30% of my lab's footprint are due to commuting and 32% to business trips. Since I make specific calculations for these two items in the following, I will retain only the 38% remaining, which include electricity, heating, digital equipment and current expenses. This results in 2.1 * 0.38 = 0.795 tCO_2e per person per year.

  Over the three-year period of my PhD: **2394 $kgCO_2e$**.

- **Commuting to work**: I came to work everyday by bike, 8 km back and forth, for three years → **24 ± 18 $kgCO_2e$**[15].

- **Business trips**: My PhD gave me the chance to travel for conferences, summer schools, national congresses or project meetings. Since the destinations were always

[12] https://www.irit.fr/missions/transition-ecologique/

[13] https://labos1point5.org/

[14] reference carbon footprint calculated for the year 2019, with the tool GES 1point5 from by Labo1point5, available at https://www.irit.fr/missions/transition-ecologique/nos-actions/le-bilan-carbone-de-lirit/

[15] using the "commutes simulator" from Labo1point5 https://apps.labos1point5.org/commutes-simulator

in Europe, I only took the train. Below is the list of trips with their associated carbon footprint in $kgCO_2e$, considering back and forth train travel from Toulouse[16]:

- SICT'21 summer school (Louvain-la-Neuve, Belgium): 32 ± 19;
- Europar'22 (Glasgow, Scotland): 55 ± 33;
- Research stay in Amsterdam, The Netherlands: 39 ± 23;
- ICT4S'23 (Rennes, France): 4 ± 1;
- ICT4S'24 (Stockholm, Sweden): 80 ± 48;
- around 10 missions in France, for project meetings and national congresses: 50 ± 10 (we consider Toulouse-Paris as an average trip).

Total: $\mathbf{260 \pm 134\ kgCO_2e}$.

- **Experiments**: The experimental results presented in this thesis arise only from simulation work. The simulations use efficient and community-proven tools, and we endeavored not to make our campaigns unnecessarily large. As such, the experimental part of this thesis has a low electricity footprint.
  Apart from my laptop whose production and usage impacts are already included in the "office work" item, I used the French research computing infrastructure Grid5000 for simulation campaigns (mainly 18-core machines from cluster "Gros" in Nancy, whose power constants are given in Table IV.1). According to Grid5000 logs, I used 1125 core-hours over the period of my PhD, i.e. 62.5 hours on machine Gros. Assuming a generous 150 W power consumption for the CPU and another 150 W for the rest (memory, network, disk, cooling), the electricity footprint of my Grid5000 usage totals 18.750 kWh. With the low French carbon electricity mix (32 gCO_2e/kWh in 2023[17]), this results in only $\mathbf{0.6\ kgCO_2e}$.

- **PhD defense**: For the PhD defense, three members of the jury came physically to Toulouse and two attended remotely. This represented three train travels in France.

  Total: $\mathbf{8 \pm 3\ kgCO_2e}$.

In the end, the carbon footprint of this thesis can be estimated to 2.7 tCO_2e. A representation of its distribution on the aforementioned items is shown in Figure A.4. Note that these figures should be taken with great precaution as they encompass large uncertainties. All the same, we can note that the carbon footprint per year (0.9 tCO_2e), is lower than my lab's average (2.1 tCO_2e). This is due to the choice of the bike as a commuting mode and the train as a business trip mode.
Still, this figure remains large. For comparison, it is estimated that each person should reduce its greenhouse gas emissions down to 2 tonnes per year by 2050, in order to limit global warming below 2°C[18]. My professional activities would already take almost half of this budget, leaving the small remaining part to all the rest (housing, eat and drink, leisure activities etc.). To be fair, professional activities are technically not taken into account in this budget, as they would be endorsed by the employer (in my case, the university) and redistributed to the consumers as part of their consumption (public services).

[16] using the "travels simulator" from Labo1point5 https://apps.labos1point5.org/travels-simulator
[17] https://www.rte-france.com
[18] https://en.2tonnes.org/

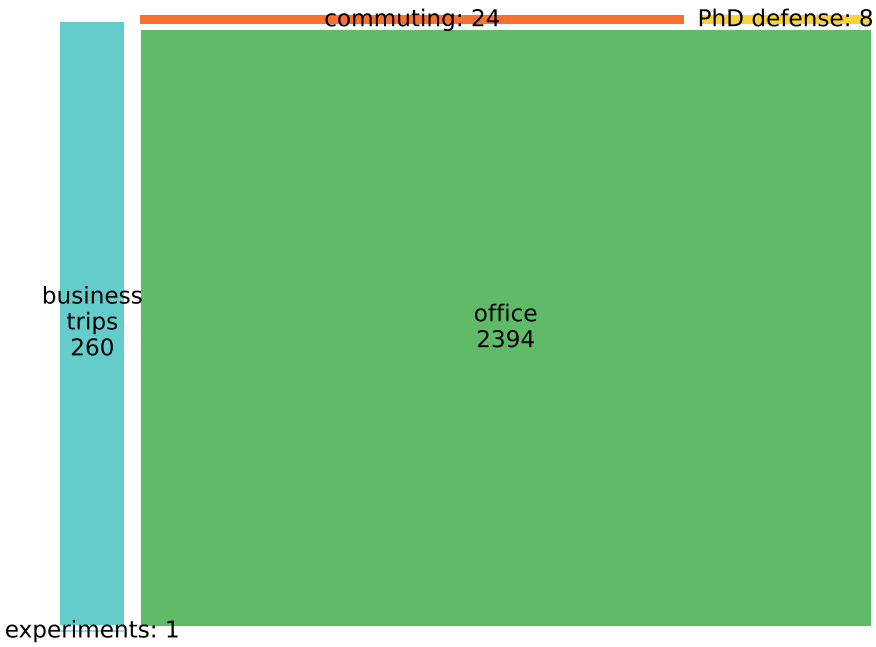


Figure A.4: **Estimated carbon footprint of this PhD thesis, in kgCO_2e**

Bibliography

[1] Katherine Richardson et al. "Earth beyond Six of Nine Planetary Boundaries". In: *Science Advances* 9.37 (Sept. 2023). DOI: 10.1126/sciadv.adh2458.

[2] Charlotte Freitag, Mike Berners-Lee, Kelly Widdicks, Bran Knowles, Gordon S. Blair, and Adrian Friday. "The Real Climate and Transformative Impact of ICT: A Critique of Estimates, Trends, and Regulations". In: *Patterns* 2.9 (Sept. 2021). ISSN: 2666-3899. DOI: 10.1016/j.patter.2021.100340.

[3] ADEME and ARCEP. *Evaluation de l'impact environnemental du numérique en France et analyse prospective*. Note de synthèse. Ademe et ARCEP, Jan. 2022, p. 18. URL: https://librairie.ademe.fr/consommer-autrement/5226-evaluation-de-l-impact-environnemental-du-numerique-en-france-et-analyse-prospective.html.

[4] Eric Masanet, Arman Shehabi, Nuoa Lei, Sarah Smith, and Jonathan Koomey. "Recalibrating Global Data Center Energy-Use Estimates". In: *Science* 367.6481 (Feb. 2020), pp. 984–986. DOI: 10.1126/science.aba3758.

[5] Lorenz M. Hilty and Bernard Aebischer. "Ict for Sustainability: An Emerging Research Field". In: *ICT Innovations for Sustainability*. Springer, 2015, pp. 3–36. DOI: 10.1007/978-3-319-09228-7_1.

[6] Vlad Coroama and Friedemann Mattern. "Digital Rebound – Why Digitalization Will Not Redeem Us Our Environmental Sins". In: *ICT4S 2019: International Conference on ICT for Sustainability*. Lappeenranta, Finland, 2019, p. 10. URL: https://ceur-ws.org/Vol-2382/ICT4S2019_paper_31.pdf.

[7] IPCC. *Climate Change 2022: Mitigation of Climate Change. Summary for Policymakers*. Tech. rep. 6. IPCC, 2022. URL: https://www.ipcc.ch/report/ar6/wg3/downloads/report/IPCC_AR6_WGIII_SPM.pdf (visited on 07/15/2022).

[8] Tilman Santarius, Jan C. T. Bieser, Vivian Frick, Mattias Höjer, Maike Gossen, Lorenz M. Hilty, Eva Kern, Johanna Pohl, Friederike Rohde, and Steffen Lange. "Digital Sufficiency: Conceptual Considerations for ICTs on a Finite Planet". In: *Annals of Telecommunications* (May 2022). ISSN: 0003-4347, 1958-9395. DOI: 10.1007/s12243-022-00914-x.

[9] Cécile Decker. *Pourquoi la sobriété numérique est au cœur des offres de TeleCoop*. Jan. 2021. URL: https://telecoop.fr/blog/pourquoi-placer-la-sobriete-numerique-au-coeur-des-offres-de-telecoop (visited on 12/26/2023).

[10] Footprint Data Foundation. *National Footprint and Biocapacity Accounts*. 2023. URL: https://data.footprintnetwork.org (visited on 02/08/2024).

[11] Richard Heinberg and Daniel Lerch. "What Is Sustainability?" In: *The Post Carbon Reader: Managing the 21st Century's Sustainability Crises*. Healdsburg, CA: Watershed Media. 2010.

[12] Monique Grooten and Rosamunde EA Almond. *Living Planet Report-2018: Aiming Higher*. Gland, Switzerland: WWF international, 2018. ISBN: 978-2-940529-90-2. URL: https://www.wwf.org.uk/sites/default/files/2018-10/wwfintl_livingplanet_full.pdf.

[13] IPCC. *Climate Change 2021 – The Physical Science Basis*. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change. Cambridge: Cambridge University Press, 2021. ISBN: 978-1-00-915789-6. DOI: 10.1017/9781009157896.

[14] Johan Rockström et al. "Planetary Boundaries: Exploring the Safe Operating Space for Humanity". In: *Ecology and Society* 14.2 (Nov. 2009). ISSN: 1708-3087. DOI: 10.5751/ES-03180-140232.

[15] Kate Raworth. *A Safe and Just Space for Humanity. Can We Live within the Doughnut?* Oxfam, Feb. 2012. ISBN: 1-78077-059-6. URL: https://primarysources.brillonline.com/browse/human-rights-documents-online/a-safe-and-just-space-for-humanity-can-we-live-within-the-doughnut;hrdhrd98240069 (visited on 11/12/2019).

[16] ISO. *ISO 14040: Environmental Management - Life Cycle Assessment - Principles and Framework.* 2006. URL: https://www.iso.org/standard/37456.html.

[17] Fehmi Görkem Üçtuğ and Tayyar Can Ünver. "Life Cycle Assessment-Based Environmental Impact Analysis of a Tier 4 Data Center: A Case Study in Turkey". In: *Sustainable Energy Technologies and Assessments* 56 (Mar. 2023), p. 103076. ISSN: 2213-1388. DOI: 10.1016/j.seta.2023.103076.

[18] Anders Andrae and Tomas Edler. "On Global Electricity Usage of Communication Technology: Trends to 2030". In: *Challenges* 6.1 (Apr. 2015), pp. 117–157. ISSN: 2078-1547. DOI: 10.3390/challe6010117.

[19] Jens Malmodin, Nina Lövehagen, Pernilla Bergmark, and Dag Lundén. "ICT Sector Electricity Consumption and Greenhouse Gas Emissions – 2020 Outcome". In: *Telecommunications Policy* (Dec. 2023), p. 102701. ISSN: 0308-5961. DOI: 10.1016/j.telpol.2023.102701.

[20] Lotfi Belkhir and Ahmed Elmeligi. "Assessing ICT Global Emissions Footprint: Trends to 2040 & Recommendations". In: *Journal of Cleaner Production* 177 (Mar. 2018), pp. 448–463. ISSN: 09596526. DOI: 10.1016/j.jclepro.2017.12.239.

[21] Francis Charpentier, Bernardo Martins, and Pascal Bourcier. "Estimating the Carbon Footprint of ICT Using Input-Output Analysis: Dealing With Overcounting and Other Challenges". In: *2023 International Conference on ICT for Sustainability (ICT4S)*. Rennes, France: IEEE, June 2023, pp. 154–163. ISBN: 9798350311099. DOI: 10.1109/ICT4S58814.2023.00024.

[22] Katayoun Rahbar, Chin Choy Chai, and Rui Zhang. "Energy Cooperation Optimization in Microgrids With Renewable Energy Integration". In: *IEEE Transactions on Smart Grid* 9.2 (Mar. 2018), pp. 1482–1493. ISSN: 1949-3061. DOI: 10.1109/TSG.2016.2600863.

[23] Noémie Périvier, Chamsi Hssaine, Samitha Samaranayake, and Siddhartha Banerjee. "Real-Time Approximate Routing for Smart Transit Systems". In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 5.2 (June 2021), 24:1–24:30. DOI: 10.1145/3460091.

[24] Hans Nijland and Jordy van Meerkerk. "Mobility and Environmental Impacts of Car Sharing in the Netherlands". In: *Environmental Innovation and Societal Transitions*. Sustainability Perspectives on the Sharing Economy 23 (June 2017), pp. 84–91. ISSN: 2210-4224. DOI: 10.1016/j.eist.2017.02.001.

[25] William Stanley Jevons. *The Coal Question; an Inquiry Concerning the Progress of the Nation and the Probable Exhaustion of Our Coal-Mines*. Macmillan, 1866.

[26] Steffen Lange, Florian Kern, Jan Peuckert, and Tilman Santarius. "The Jevons Paradox Unravelled: A Multi-Level Typology of Rebound Effects and Mechanisms". In: *Energy Research & Social Science* 74 (Apr. 2021), p. 101982. ISSN: 2214-6296. DOI: 10.1016/j.erss.2021.101982.

[27] Yuke Zhu and Mudan Lan. "Digital Economy and Carbon Rebound Effect: Evidence from Chinese Cities". In: *Energy Economics* 126 (Oct. 2023), p. 106957. ISSN: 0140-9883. DOI: 10.1016/j.eneco.2023.106957.

[28] Kelly Widdicks et al. "Systems Thinking and Efficiency under Emissions Constraints: Addressing Rebound Effects in Digital Innovation and Policy". In: *Patterns* 4.2 (Feb. 2023), p. 100679. ISSN: 26663899. DOI: 10.1016/j.patter.2023.100679.

[29] Christina Bremer, George Kamiya, Pernilla Bergmark, Vlad C. Coroama, Eric Masanet, and Reid Lifset. "Assessing Energy and Climate Effects of Digitalization: Methodological Challenges and Key Recommendations". In: *nDEE Framing Paper Series* (2023). URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4459526 (visited on 02/16/2024).

[30] Md Abu Bakar Siddik, Arman Shehabi, and Landon Marston. "The Environmental Footprint of Data Centers in the United States". In: *Environmental Research Letters* 16.6 (May 2021), p. 064017. ISSN: 1748-9326. DOI: 10.1088/1748-9326/abfba1.

[31] Patrick Bresnihan and Patrick Brodie. "Data Sinks, Carbon Services: Waste, Storage and Energy Cultures on Ireland's Peat Bogs". In: *New Media & Society* 25.2 (Feb. 2023), pp. 361–383. ISSN: 1461-4448. DOI: 10.1177/14614448221149948.

[32] Anna Bessin, Floris de Jong, Patricia Lago, and Oscar Widerberg. "News from Europe's Digital Gateway: A Proof of Concept for Mapping Data Centre News Coverage". In: *Advances and New Trends in Environmental Informatics 2023*. Ed. by Volker Wohlgemuth, Dieter Kranzlmüller, and Maximilian Höb. Progress in IS. Cham: Springer Nature Switzerland, 2024, pp. 161–175. ISBN: 978-3-031-46902-2. DOI: 10.1007/978-3-031-46902-2_9.

[33] Sukhpal Singh Gill and Rajkumar Buyya. "A Taxonomy and Future Directions for Sustainable Cloud Computing: 360 Degree View". In: *ACM Computing Surveys* 51.5 (Dec. 2018), 104:1–104:33. ISSN: 0360-0300. DOI: 10.1145/3241038.

[34] Salil Bharany, Sandeep Sharma, Osamah Ibrahim Khalaf, Ghaida Muttashar Abdulsahib, Abeer S. Al Humaimeedy, Theyazn H. H. Aldhyani, Mashael Maashi, and Hasan Alkahtani. "A Systematic Survey on Energy-Efficient Techniques in Sustainable Cloud Computing". In: *Sustainability* 14.10 (Jan. 2022), p. 6256. ISSN: 2071-1050. DOI: 10.3390/su14106256.

[35] C. A. Silva, R. Vilaça, A. Pereira, and R. J. Bessa. "A Review on the Decarbonization of High-Performance Computing Centers". In: *Renewable and Sustainable Energy Reviews* 189 (Jan. 2024), p. 114019. ISSN: 1364-0321. DOI: 10.1016/j.rser.2023.114019.

[36] Kyong Hoon Kim, Rajkumar Buyya, and Jong Kim. "Power Aware Scheduling of Bag-of-Tasks Applications with Deadline Constraints on DVS-enabled Clusters". In: *Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid '07)*. May 2007, pp. 541–548. DOI: 10.1109/CCGRID.2007.85.

[37] Tom Guérout, Thierry Monteil, Georges Da Costa, Rodrigo Neves Calheiros, Rajkumar Buyya, and Mihai Alexandru. "Energy-Aware Simulation with DVFS". In: *Simulation Modelling Practice and Theory*. S.I.Energy Efficiency in Grids and Clouds 39 (Dec. 2013), pp. 76–91. ISSN: 1569-190X. DOI: 10.1016/j.simpat.2013.04.007.

[38] Anne-Cécile Orgerie, Laurent Lefèvre, and Jean-Patrick Gelas. "Chasing Gaps between Bursts: Towards Energy Efficient Large Scale Experimental Grids". In: *2008 Ninth International Conference on Parallel and Distributed Computing, Applications and Technologies*. Dunedin, New Zealand: IEEE, 2008, pp. 381–389. ISBN: 978-0-7695-3443-5. DOI: 10.1109/PDCAT.2008.80.

[39] Damien Borgetto, Michael Maurer, Georges Da-Costa, Jean-Marc Pierson, and Ivona Brandic. "Energy-Efficient and SLA-aware Management of IaaS Clouds". In: *Proceedings of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet*. E-Energy '12. New York, NY, USA: Association for Computing Machinery, May 2012, pp. 1–10. ISBN: 978-1-4503-1055-0. DOI: 10.1145/2208828.2208853.

[40] Belen Bermejo, Carlos Juiz, and Carlos Guerrero. "Virtualization and Consolidation: A Systematic Review of the Past 10 Years of Research on Energy and Performance". In: *The Journal of Supercomputing* 75.2 (Feb. 2019), pp. 808–836. ISSN: 1573-0484. DOI: 10.1007/s11227-018-2613-1.

[41] Belén Bermejo, Sonja Filiposka, Carlos Juiz, Beatriz Gómez, and Carlos Guerrero. "Improving the Energy Efficiency in Cloud Computing Data Centres Through Resource Allocation Techniques". In: *Research Advances in Cloud Computing*. Ed. by Sanjay Chaudhary, Gaurav Somani, and Rajkumar Buyya. Singapore: Springer, 2017, pp. 211–236. ISBN: 978-981-10-5026-8. DOI: 10.1007/978-981-10-5026-8_9.

[42] Violaine Villebonnet, Georges Da Costa, Laurent Lefevre, Jean-Marc Pierson, and Patricia Stolf. ""Big, Medium, Little": Reaching Energy Proportionality with Heterogeneous Computing Scheduler". In: *Parallel Processing Letters* 25.03 (Sept. 2015), p. 1541006. ISSN: 0129-6264. DOI: 10.1142/S0129626415410066.

[43] Jannis Klinkenberg, Anara Kozhokanova, Christian Terboven, Clément Foyer, Brice Goglin, and Emmanuel Jeannot. "H2M: Exploiting Heterogeneous Shared Memory Architectures". In: *Future Generation Computer Systems* 148 (Nov. 2023), pp. 39–55. ISSN: 0167-739X. DOI: 10.1016/j.future.2023.05.019.

[44] Emmanuel Jeannot, Guillaume Pallez, and Nicolas Vidal. "IO-aware Job-Scheduling: Exploiting the Impacts of Workload Characterizations to Select the Mapping Strategy". In: *International Journal of High Performance Computing Applications* (2023), p. 1. DOI: 10.1177/10943420231175854.

[45] Jacqueline Davis, Daniel Bizo, Andy Lawrence, Owen Rogers, and Max Smolaks. *Uptime Institute Global Data Center Survey 2022*. Tech. rep. 12. New York: Uptime Institute, Sept. 2022, p. 33. URL: https://uptimeinstitute.com/resources/research-and-reports/uptime-institute-global-data-center-survey-results-2022.

[46] Ali Habibi Khalaj and Saman K. Halgamuge. "A Review on Efficient Thermal Management of Air- and Liquid-Cooled Data Centers: From Chip to the Cooling System". In: *Applied Energy* 205 (Nov. 2017), pp. 1165–1188. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2017.08.037.

[47] Hongyang Sun, Patricia Stolf, and Jean-Marc Pierson. "Spatio-Temporal Thermal-Aware Scheduling for Homogeneous High-Performance Computing Datacenters". In: *Future Generation Computer Systems* 71 (June 2017), pp. 157–170. ISSN: 0167739X. DOI: 10.1016/j.future.2017.02.005.

[48] Maël Madon and Jean-Marc Pierson. "Integrating Pre-Cooling of Data Center Operated with Renewable Energies". In: *2020 IEEE Green Computing and Communications (GreenCom)*. Rhodos, 2020, p. 10. URL: https://hal.archives-ouvertes.fr/hal-02970488.

[49] Gary Cook, Jude Lee, Tamina Tsai, Ada Kong, John Deans, Brian Johnson, and Elizabeth Jardim. *Clicking Clean: Who Is Winning the Race to Build a Green Internet?* Tech. rep. Greenpeace Inc, Jan. 2017, p. 102. URL: https://www.greenpeace.de/sites/default/files/publications/20170110_greenpeace_clicking_clean.pdf.

[50] Jean-Marc Pierson et al. "DATAZERO: DATAcenter with Zero Emission and RObust Management Using Renewable Energy". In: *IEEE Access* 7 (July 2019). DOI: 10.1109/ACCESS.2019.2930368.

[51] Igor Fontana de Nardin. "Online Scheduling for IT Tasks and Power Source Commitment in Data Centers Only Operated with Renewable Energy". PhD thesis. Toulouse, France: Institut National Polytechnique de Toulouse, Dec. 2023. URL: https://theses.hal.science/tel-04361471/.

[52] Xiaoying Wang, Zhihui Du, Yinong Chen, and Mengqin Yang. "A Green-Aware Virtual Machine Migration Strategy for Sustainable Datacenter Powered by Renewable Energy". In: *Simulation Modelling Practice and Theory* 58 (Nov. 2015), pp. 3–14. ISSN: 1569-190X. DOI: 10.1016/j.simpat.2015.01.005.

[53] Laurent Hussenet and Chérifa Boucetta. "A Green-Aware Optimization Strategy for Virtual Machine Migration in Cloud Data Centers". In: *2022 International Wireless Communications and Mobile Computing (IWCMC)*. May 2022, pp. 1082–1087. DOI: 10.1109/IWCMC55113.2022.9825284.

[54] Miguel Vasconcelos, Daniel Cordeiro, Georges da Costa, Fanny Dufossé, Jean-Marc Nicod, and Veronika Rehn-Sonigo. "Optimal Sizing of a Globally Distributed Low Carbon Cloud Federation". In: *2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*. May 2023, pp. 203–215. DOI: 10.1109/CCGrid57682.2023.00028.

[55] G. Rostirolla et al. "A Survey of Challenges and Solutions for the Integration of Renewable Energy in Datacenters". In: *Renewable and Sustainable Energy Reviews* 155 (Mar. 2022), p. 111787. ISSN: 1364-0321. DOI: 10.1016/j.rser.2021.111787.

[56] Manal Benaissa, Georges Da Costa, and Jean-Marc Nicod. "Standalone Data-Center Sizing Combating the Over-Provisioning of the IT and Electrical Parts". In: *2022 International Symposium on Computer Architecture and High Performance Computing Workshops (SBAC-PADW)*. Nov. 2022, pp. 57–62. DOI: 10.1109/SBAC-PADW56527.2022.00019.

[57] Longjun Liu, Chao Li, Hongbin Sun, Yang Hu, Juncheng Gu, and Tao Li. "BAAT: Towards Dynamically Managing Battery Aging in Green Datacenters". In: *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. June 2015, pp. 307–318. DOI: 10.1109/DSN.2015.43.

[58] Arman Shehabi, Sarah Smith, Dale Sartor, Richard Brown, Magnus Herrlin, Jonathan Koomey, Eric Masanet, Nathaniel Horner, Inês Azevedo, and William Lintner. *United States Data Center Energy Usage Report*. Tech. rep. LBNL–1005775, 1372902. June 2016, LBNL–1005775, 1372902. DOI: 10.2172/1372902.

[59] IEA. *Tracking Data Centres and Data Transmission Networks*. Tech. rep. Paris, France: International Energy Agency (IEA), July 2023. URL: https://www.iea.org/energy-system/buildings/data-centres-and-data-transmission-networks (visited on 02/26/2024).

[60] Jessica Jungell-Michelsson and Pasi Heikkurinen. "Sufficiency: A Systematic Literature Review". In: *Ecological Economics* 195 (May 2022), p. 107380. ISSN: 0921-8009. DOI: 10.1016/j.ecolecon.2022.107380.

[61] Thomas Princen. "Principles for Sustainability: From Cooperation and Efficiency to Sufficiency". In: *Global Environmental Politics* 3.1 (Feb. 2003), pp. 33–50. ISSN: 1526-3800. DOI: 10.1162/152638003763336374.

[62] Thierry Salomon, Marc Jedliczka, Yves Marignac, Stephane Hessel, and Amory Lovins. *Negawatt manifesto - Making energy transition a success; Manifeste Negawatt - Reussir la transition energetique*. Jan. 2012. ISBN: 978-2-330-00018-9. URL: https://www.osti.gov/etdeweb/biblio/21539640 (visited on 02/21/2024).

[63] ADEME. *Transition(s) 2050. Choisir Maintenant. Agir Pour Le Climat*. Tech. rep. French Agency for Ecological Transition (ADEME), Nov. 2021. URL: https://librairie.ademe.fr/recherche-et-innovation/5072-prospective-transitions-2050-rapport.html (visited on 07/15/2022).

[64] French Government. *Plan de sobriété énergétique - Une mobilisation générale*. Tech. rep. French Government, Oct. 2022, p. 50. URL: https://www.ecologie.gouv.fr/sites/default/files/dp-plan-sobriete.pdf (visited on 02/21/2024).

[65] Timothée Parrique. "The Political Economy of Degrowth". PhD thesis. Université Clermont Auvergne, 2019. URL: https://hal.science/tel-02499463v1.

[66] Matthias Schmelzer, Andrea Vetter, and Aaron Vansintjan. *The Future Is Degrowth: A Guide to a World Beyond Capitalism*. Verso Books, June 2022. ISBN: 978-1-83976-584-1.

[67] Ivan Illich, Luce Giard, and Vincent Bardet. *La Convivialité*. Editions du Seuil Paris, 1973. ISBN: 2-02-002201-X.

[68] Philippe Bihouix. *L'Age Des Low Tech. Vers Une Civilisation Techniquement Soutenable*. Editions du Seuil, 2014. ISBN: 978-2-02-116074-1. URL: https://books.google.fr/books?id=W5-jAwAAQBAJ.

[69] Frédéric Bordage. *Sobriété numérique: les clés pour agir*. Buchet Chastel, 2019. ISBN: 978-2-283-03215-2. URL: https://www.greenit.fr/2019/09/10/sobriete-numerique-les-cles-pour-agir/.

[70] Fabrice Flipo. *L'impératif de la sobriété numérique : l'enjeu des modes de vie*. Essais. Éditions Matériologiques, Nov. 2020. ISBN: 978-2-37361-258-5. URL: https://hal.science/hal-03340158 (visited on 03/04/2024).

[71] Hugues Ferreboeuf, Maxime Efoui-Hess, and Zeynep Kahraman. *Lean ICT: Towards Digital Sobriety*. Tech. rep. The Shift Project, Mar. 2019, p. 90. URL: https://theshiftproject.org/wp-content/uploads/2019/03/Lean-ICT-Report_The-Shift-Project_2019.pdf (visited on 12/05/2023).

[72] Nadège Soubiale and Delphine Dupré. "Durabilité Environnementale et Sobriété Numérique à l'ère de La Gouvernance Territoriale". In: *Revue COSSI : communication, organisation, société du savoir et information* 12 (Nov. 2023). URL: https://hal.science/hal-04380641.

[73] Sarah Descamps, Gaëtan Temperman, and Bruno De Lièvre. "Vers une éducation à la sobriété numérique". In: *Humanités numériques* 5 (June 2022). ISSN: 2736-2337. DOI: 10.4000/revuehn.2858.

[74] ADEME. *En route vers la sobriété numérique*. Tech. rep. French Agency for Ecological Transition (ADEME), May 2022, p. 9. URL: https://librairie.ademe.fr/consommer-autrement/5086-en-route-vers-la-sobriete-numerique-9791029718755.html.

[75] AMUE. *Urgence sur les sobriétés numériques !* Tech. rep. AMUE, Oct. 2023. URL: https://www.amue.fr/fileadmin/amue/systeme-information/documents-publications/la-collection-numerique/amue-collection-numerique-n29-avec_compression.pdf (visited on 10/31/2023).

[76] Lorenz M. Hilty. *Information Technology and Sustainability: Essays on the Relationship Between ICT and Sustainable Development*. Books on Demand, 2008. ISBN: 978-3-8370-1970-4.

[77] Steffen Lange and Tilman Santarius. *Smart Green World?: Making Digitalization Work for Sustainability*. Routledge. May 2020. ISBN: 978-0-367-46757-9. URL: https://www.routledge.com/Smart-Green-World-Making-Digitalization-Work-for-Sustainability/Lange-Santarius/p/book/9780367467579 (visited on 08/31/2021).

[78] Joel Millward-Hopkins, Julia K. Steinberger, Narasimha D. Rao, and Yannick Oswald. "Providing Decent Living with Minimum Energy: A Global Scenario". In: *Global Environmental Change* 65 (Nov. 2020), p. 102168. ISSN: 0959-3780. DOI: 10.1016/j.gloenvcha.2020.102168.

[79] Narasimha D. Rao and Jihoon Min. "Decent Living Standards: Material Prerequisites for Human Wellbeing". In: *Social Indicators Research* 138.1 (July 2018), pp. 225–244. ISSN: 1573-0921. DOI: 10.1007/s11205-017-1650-0.

[80] Hélène Gorge, Maud Herbert, Nil Özçağlar-Toulouse, and Isabelle Robert. "What Do We Really Need? Questioning Consumption Through Sufficiency". In: *Journal of Macromarketing* 35.1 (Mar. 2015), pp. 11–22. ISSN: 0276-1467, 1552-6534. DOI: 10.1177/0276146714553935.

[81] Kelly Victoria Widdicks, Tina Ringenson, Daniel Pargman, Vishnupriya Kuppusamy, and Patricia Lago. "Undesigning the Internet:An Exploratory Study of Reducing Everyday Internet Connectivity". In: *ICT4S 2018: International Conference on Information and Communication Technology for Sustainability*. Ed. by Birgit Penzenstadler, Steve Easterbrook, Colin Venters, and Syed Istiaque Ahmed. May 2018, pp. 384–397. DOI: 10.29007/s221.

[82] Kelly Widdicks, Christian Remy, Oliver Bates, Adrian Friday, and Mike Hazas. "Escaping Unsustainable Digital Interactions: Toward "More Meaningful" and "Moderate" Online Experiences". In: *International Journal of Human-Computer Studies* 165 (Sept. 2022), p. 102853. ISSN: 10715819. DOI: 10.1016/j.ijhcs.2022.102853.

[83] Leila Elgaaied-Gambier, Laurent Bertrandias, and Yohan Bernard. "Cutting the Internet's Environmental Footprint: An Analysis of Consumers' Self-Attribution of Responsibility". In: *Journal of Interactive Marketing* 50.1 (May 2020), pp. 120–135. ISSN: 1094-9968. DOI: 10.1016/j.intmar.2020.02.001.

[84] Anja Kollmuss and Julian Agyeman. "Mind the Gap: Why Do People Act Environmentally and What Are the Barriers to Pro-Environmental Behavior?" In: *Environmental Education Research* 8 (Aug. 2002), pp. 239–260. DOI: 10.1080/13504620220145401.

[85] Jon Froehlich, Leah Findlater, and James Landay. "The Design of Eco-Feedback Technology". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Atlanta Georgia USA: ACM, Apr. 2010, pp. 1999–2008. ISBN: 978-1-60558-929-9. DOI: 10.1145/1753326.1753629.

[86] Angela Sanguinetti, Kelsea Dombrovski, and Suhaila Sikand. "Information, Timing, and Display: A Design-Behavior Framework for Improving the Effectiveness of Eco-Feedback". In: *Energy Research & Social Science* 39 (May 2018), pp. 55–68. ISSN: 2214-6296. DOI: 10.1016/j.erss.2017.10.001.

[87] Adel Noureddine, Martin Dieguez Lodeiro, Noelle Bru, and Richard Chbeir. "The Impact of Green Feedback on Users' Software Usage". In: *IEEE Transactions on Sustainable Computing* (2022), pp. 1–14. ISSN: 2377-3782. DOI: 10.1109/TSUSC.2022.3222631.

[88] Saurabh Kumar Garg, Chee Shin Yeo, and Rajkumar Buyya. "Green Cloud Framework for Improving Carbon Efficiency of Clouds". In: *Euro-Par 2011 Parallel Processing*. Ed. by Emmanuel Jeannot, Raymond Namyst, and Jean Roman. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2011, pp. 491–502. ISBN: 978-3-642-23400-2. DOI: 10.1007/978-3-642-23400-2_45.

[89] Gregor von Laszewski and Lizhe Wang. "GreenIT Service Level Agreements". In: *Grids and Service-Oriented Architectures for Service Level Agreements*. Ed. by Philipp Wieder, Ramin Yahyapour, and Wolfgang Ziegler. Boston, MA: Springer US, 2010, pp. 77–88. ISBN: 978-1-4419-7319-1 978-1-4419-7320-7. DOI: 10.1007/978-1-4419-7320-7_8.

[90] Christian Bunse, Sonja Klingert, and Thomas Schulze. "GreenSLAs: Supporting Energy-Efficiency through Contracts". In: *Energy Efficient Data Centers*. Ed. by Jyrki Huusko, Hermann de Meer, Sonja Klingert, and Andrey Somov. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2012, pp. 54–68. ISBN: 978-3-642-33645-4. DOI: 10.1007/978-3-642-33645-4_6.

[91] Md. E. Haque, Kien Le, Íñigo Goiri, Ricardo Bianchini, and Thu D. Nguyen. "Providing Green SLAs in High Performance Computing Clouds". In: *2013 International Green Computing Conference Proceedings*. June 2013, pp. 1–11. DOI: 10.1109/IGCC.2013.6604503.

[92] Ahmed Amokrane, Rami Langar, Mohamed Faten Zhani, Raouf Boutaba, and Guy Pujolle. "Greenslater: On Satisfying Green SLAs in Distributed Clouds". In: *IEEE Transactions on Network and Service Management* 12.3 (Sept. 2015), pp. 363–376. ISSN: 1932-4537. DOI: 10.1109/TNSM.2015.2440423.

[93] Sudhir Goyal, Seema Bawa, and Bhupinder Singh. "Green Service Level Agreement (GSLA) Framework for Cloud Computing". In: *Computing* 98.9 (Sept. 2016), pp. 949–963. ISSN: 1436-5057. DOI: 10.1007/s00607-015-0481-6.

[94] M. S. Hasan, Y. Kouki, T. Ledoux, and J. Pazat. "Exploiting Renewable Sources: When Green SLA Becomes a Possible Reality in Cloud Computing". In: *IEEE Transactions on Cloud Computing* 5.2 (Apr. 2017), pp. 249–262. ISSN: 2168-7161. DOI: 10.1109/TCC.2015.2459710.

[95] Peini Liu, Gusseppe Bravo, and Jordi Guitart. "Energy-Aware Dynamic Pricing Model for Cloud Environments". In: *Economics of Grids, Clouds, Systems, and Services*. Ed. by Karim Djemame, Jörn Altmann, José Ángel Bañares, Orna Agmon Ben-Yehuda, and Maurizio Naldi. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 71–80. ISBN: 978-3-030-36027-6. DOI: 10.1007/978-3-030-36027-6_7.

[96] Mohammad Aldossary, Karim Djemame, Ibrahim Alzamil, Alexandros Kostopoulos, Antonis Dimakis, and Eleni Agiatzidou. "Energy-Aware Cost Prediction and Pricing of Virtual Machines in Cloud Computing Environments". In: *Future Generation Computer Systems* 93 (Apr. 2019), pp. 442–459. ISSN: 0167-739X. DOI: 10.1016/j.future.2018.10.027.

[97] Debdeep Paul, Wen-De Zhong, and Sanjay Kumar Bose. "Energy Aware Pricing in a Three-Tiered Cloud Service Market". In: *Electronics* 5.4 (Dec. 2016), p. 65. DOI: 10.3390/electronics5040065.

[98] Andrea Borghesi, Andrea Bartolini, Michela Milano, and Luca Benini. "Pricing Schemes for Energy-Efficient HPC Systems: Design and Exploration". In: *The International Journal of High Performance Computing Applications* 33.4 (July 2019), pp. 716–734. ISSN: 1094-3420. DOI: 10.1177/1094342018814593.

[99] Sonja Klingert, Andreas Berl, Michael Beck, Radu Serban, Marco di Girolamo, Giovanni Giuliani, Hermann de Meer, and Alfons Salden. "Sustainable Energy Management in Data Centers through Collaboration". In: *Energy Efficient Data Centers*. Ed. by Jyrki Huusko, Hermann de Meer, Sonja Klingert, and Andrey Somov. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2012, pp. 13–24. ISBN: 978-3-642-33645-4. DOI: 10.1007/978-3-642-33645-4_2.

[100] Yiannis Georgiou, David Glesser, Krzysztof Rzadca, and Denis Trystram. "A Scheduler-Level Incentive Mechanism for Energy Efficiency in HPC". In: *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. Shenzhen, China: IEEE, May 2015, pp. 617–626. ISBN: 978-1-4799-8006-2. DOI: 10.1109/CCGrid.2015.101.

[101] A. Orgerie, L. Lefèvre, and J. Gelas. "Save Watts in Your Grid: Green Strategies for Energy-Aware Framework in Large Scale Distributed Systems". In: *2008 14th IEEE International Conference on Parallel and Distributed Systems*. Dec. 2008, pp. 171–178. DOI: 10.1109/ICPADS.2008.97.

[102] Cinzia Cappiello, Paco Melià, Barbara Pernici, Pierluigi Plebani, and Monica Vitali. "Sustainable Choices for Cloud Applications: A Focus on CO2 Emissions". In: *ICT for Sustainability 2014 (ICT4S-14)*. Atlantis Press, Aug. 2014, pp. 352–358. ISBN: 978-94-6252-022-6. DOI: 10.2991/ict4s-14.2014.43.

[103] David Guyon, Anne-Cécile Orgerie, Christine Morin, and Deb Agarwal. "Involving Users in Energy Conservation: A Case Study in Scientific Clouds". In: *International Journal of Grid and Utility Computing* 10.3 (Jan. 2019), pp. 272–282. ISSN: 1741-847X. DOI: 10.1504/IJGUC.2019.099667.

[104] David Guyon, Anne-Cécile Orgerie, and Christine Morin. "Energy - Efficient IaaS-PaaS Co-Design for Flexible Cloud Deployment of Scientific Applications". In: *2018 30th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*. Sept. 2018, pp. 69–76. DOI: 10.1109/CAHPC.2018.8645888.

[105] Robert Basmadjian, Juan Felipe Botero, Giovanni Giuliani, Xavier Hesselbach, Sonja Klingert, and Hermann De Meer. "Making Data Centers Fit for Demand Response: Introducing GreenSDA and GreenSLA Contracts". In: *IEEE Transactions on Smart Grid* 9.4 (July 2018), pp. 3453–3464. ISSN: 1949-3061. DOI: 10.1109/TSG.2016.2632526.

[106] Franz C. Heinrich, Alexandra Carpen-Amarie, Augustin Degomme, Sascha Hunold, Arnaud Legrand, Anne-Cécile Orgerie, and Martin Quinson. *Predicting the Performance and the Power Consumption of MPI Applications With SimGrid*. Jan. 2017. URL: https://inria.hal.science/hal-01446134 (visited on 01/31/2024).

[107] 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 (June 2014), p. 2899. DOI: 10.1016/j.jpdc.2014.06.008.

[108] Pierre-François Dutot, Michael Mercier, Millian Poquet, and Olivier Richard. "Batsim: A Realistic Language-Independent Resources and Jobs Management Systems Simulator". In: *20th Workshop on Job Scheduling Strategies for Parallel Processing*. Chicago, United States, May 2016. DOI: 10.1007/978-3-319-61756-5_10.

[109] Loic Desquilbet et al. *Vers Une Recherche Reproductible*. Ed. by Unité régionale de formation à l'information scientifique et technique de Bordeaux. Unité régionale de formation à l'information scientifique et technique de Bordeaux, May 2019. URL: https://hal.science/hal-02144142.

[110] ACM. *Artifact Review and Badging Version 1.1*. Aug. 2020. URL: https://www.acm.org/publications/policies/artifact-review-and-badging-current (visited on 11/21/2023).

[111] Jay Zarnikau and Dan Thal. "The Response of Large Industrial Energy Consumers to Four Coincident Peak (4CP) Transmission Charges in the Texas (ERCOT) Market". In: *Utilities Policy* 26 (Sept. 2013), pp. 1–6. ISSN: 0957-1787. DOI: 10.1016/j.jup.2013.04.004.

[112] Zhenhua Liu, Adam Wierman, Yuan Chen, Benjamin Razon, and Niangjun Chen. "Data Center Demand Response: Avoiding the Coincident Peak via Workload Shifting and Local Generation". In: *Performance Evaluation*. Proceedings of IFIP Performance 2013 Conference 70.10 (Oct. 2013), pp. 770–791. ISSN: 0166-5316. DOI: 10.1016/j.peva.2013.08.014.

[113] Adam Wierman, Zhenhua Liu, Iris Liu, and Hamed Mohsenian-Rad. "Opportunities and Challenges for Data Center Demand Response". In: *International Green Computing Conference*. Nov. 2014, pp. 1–10. DOI: 10.1109/IGCC.2014.7039172.

[114] Dalibor Klusáček, Šimon Tóth, and Gabriela Podolníková. "Real-Life Experience with Major Reconfiguration of Job Scheduling System". In: *Job Scheduling Strategies for Parallel Processing*. Ed. by Narayan Desai and Walfredo Cirne. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2017, pp. 83–101. ISBN: 978-3-319-61756-5. DOI: 10.1007/978-3-319-61756-5_5.

[115] Loïc Lannelongue, Jason Grealey, and Michael Inouye. "Green Algorithms: Quantifying the Carbon Footprint of Computation". In: *Advanced Science* 8.12 (2021), p. 2100707. ISSN: 2198-3844. DOI: 10.1002/advs.202100707.

[116] Roel Roscam Abbing. "'This Is a Solar-Powered Website, Which Means It Sometimes Goes Offline': A Design Inquiry into Degrowth and ICT". In: *LIMITS Workshop on Computing within Limits*. June 2021. DOI: 10.21428/bf6fb269.e78d19f6.

[117] Google. *Data Centers: Belgian Site Adds Solar - Google Sustainability*. Sept. 2018. URL: https://sustainability.google/operating-sustainably/stories/belgium-solar/ (visited on 12/25/2023).

[118] Iain Staffell and Stefan Pfenninger. "Using Bias-Corrected Reanalysis to Simulate Current and Future Wind Power Output". In: *Energy* 114 (Nov. 2016), pp. 1224–1239. ISSN: 0360-5442. DOI: 10.1016/j.energy.2016.08.068.

[119] Stefan Pfenninger and Iain Staffell. "Long-Term Patterns of European PV Output Using 30 Years of Validated Hourly Reanalysis and Satellite Data". In: *Energy* 114 (Nov. 2016), pp. 1251–1265. ISSN: 0360-5442. DOI: 10.1016/j.energy.2016.08.060.

[120] Philipp Wiesner, Dominik Scheinert, Thorsten Wittkopp, Lauritz Thamsen, and Odej Kao. "Cucumber: Renewable-Aware Admission Control for Delay-Tolerant Cloud and Edge Workloads". In: *Euro-Par 2022: Parallel Processing*. Ed. by José Cano and Phil Trinder. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2022, pp. 218–232. ISBN: 978-3-031-12597-3. DOI: 10.1007/978-3-031-12597-3_14.

[121] Igor Fontana de Nardin, Patricia Stolf, and Stephane Caux. "Analyzing Power Decisions in Data Center Powered by Renewable Sources". In: *2022 IEEE 34th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*. Nov. 2022, pp. 305–314. DOI: 10.1109/SBAC-PAD55451.2022.00041.

[122] Netanel Zakay and Dror G. Feitelson. "Preserving User Behavior Characteristics in Trace-Based Simulation of Parallel Job Scheduling". In: *Proceedings of the 8th ACM International Systems and Storage Conference*. Haifa Israel: ACM, May 2015, pp. 1–1. ISBN: 978-1-4503-3607-9. DOI: 10.1145/2757667.2778191.

[123] Stephan Schlagkamp. "Influence of Dynamic Think Times on Parallel Job Scheduler Performances in Generative Simulations". In: *Job Scheduling Strategies for Parallel Processing*. Ed. by Narayan Desai and Walfredo Cirne. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2017, pp. 123–137. ISBN: 978-3-319-61756-5. DOI: 10.1007/978-3-319-61756-5_7.

[124] Bianca Schroeder, Adam Wierman, and Mor Harchol-Balter. "Open versus Closed: A Cautionary Tale". In: *Symposium on Networked Systems Design and Implementation*. San Jose, CA, USA: Carnegie Mellon University, May 2006. URL: https://dl.acm.org/doi/abs/10.5555/1267680.1267698.

[125] Netanel Zakay and Dror G. Feitelson. "On Identifying User Session Boundaries in Parallel Workload Logs". In: *Job Scheduling Strategies for Parallel Processing*. Ed. by Walfredo Cirne, Narayan Desai, Eitan Frachtenberg, and Uwe Schwiegelshohn. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2013, pp. 216–234. ISBN: 978-3-642-35867-8. DOI: 10.1007/978-3-642-35867-8_12.

[126] Dalibor Klusáček, Šimon Tóth, and Gabriela Podolníková. "Complex Job Scheduling Simulations with Alea 4". In: *SIMUTOOLS'16: International Conference on Simulation Tools and Techniques*. Brussels, Belgium, Aug. 2016, pp. 124–129. ISBN: 978-1-63190-120-1. DOI: 10.5555/3021426.3021446.

[127] David A. Lifka. "The ANL/IBM SP Scheduling System". In: *Job Scheduling Strategies for Parallel Processing*. Ed. by Gerhard Goos, Juris Hartmanis, Jan Leeuwen, Dror G. Feitelson, and Larry Rudolph. Vol. 949. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 295–303. ISBN: 978-3-540-60153-1 978-3-540-49459-1. DOI: 10.1007/3-540-60153-8_35.

[128] Dror G. Feitelson. "Resampling with Feedback — A New Paradigm of Using Workload Data for Performance Evaluation". In: *Euro-Par 2016: Parallel Processing*. Ed. by Pierre-François Dutot and Denis Trystram. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 3–21. ISBN: 978-3-319-43659-3. DOI: 10.1007/978-3-319-43659-3_1.

[129] Nicole Wolter, Michael O McCracken, Allan Snavely, Lorin Hochstein, Taiga Nakamura, and Victor Basili. "What's Working in HPC: Investigating HPC User Behavior and Productivity". In: (2006), p. 14. URL: http://www.cs.umd.edu/~basili/publications/journals/J105.pdf.

[130] Peter Mell and Timothy Grance. *The NIST Definition of Cloud Computing*. Tech. rep. Special Publication 800-145. National Institute of Standards and Technology, Sept. 2011, p. 7. URL: http://faculty.winthrop.edu/domanm/csci411/Handouts/NIST.pdf.

[131] gov.uk. *Flexible Working: Definition by the UK Government*. URL: https://www.gov.uk/flexible-working (visited on 09/05/2022).

[132] Jonathan Stiles. "Working at Home and Elsewhere in the City: Mobile Cloud Computing, Telework, and Urban Travel". PhD thesis. Rutgers University - School of Graduate Studies, 2019. DOI: 10.7282/t3-nw1e-yr83.

[133] Anja Feldmann et al. "The Lockdown Effect: Implications of the COVID-19 Pandemic on Internet Traffic". In: *Proceedings of the ACM Internet Measurement Conference*. IMC '20. New York, NY, USA: Association for Computing Machinery, Oct. 2020, pp. 1–18. ISBN: 978-1-4503-8138-3. DOI: 10.1145/3419394.3423658.

[134] William O'Brien and Fereshteh Yazdani Aliabadi. "Does Telecommuting Save Energy? A Critical Review of Quantitative Studies and Their Research Methods". In: *Energy and Buildings* 225 (Oct. 2020), p. 110298. ISSN: 0378-7788. DOI: 10.1016/j.enbuild.2020.110298.

[135] Ericsson. *The Dematerialized Office*. Tech. rep. Ericsson, Oct. 2020. URL: www.ericsson.com/industrylab.

[136] Moira Maguire and Brid Delahunt. "Doing a Thematic Analysis: A Practical, Step-by-Step Guide for Learning and Teaching Scholars." In: *All Ireland Journal of Higher Education* 9.3 (Oct. 2017). ISSN: 2009-3160. URL: https://ojs.aishe.org/index.php/aishe-j/article/view/335 (visited on 01/19/2023).

[137] Maël Madon and Patricia Lago. *Replication Package for the Study "Digital Sufficiency in Flexible Work"*. Feb. 2023. DOI: 10.5281/zenodo.7645519.

[138] Donella Meadows. "Leverage Points: Places to Intervene in a System". In: *The Sustainability Institute* (1999), p. 18.

[139] Maliha Nawshin Rahman. "Digital Sufficiency of Cloud Usage in Flexible Work". MA thesis. Lappeenranta-Lahti University of Technology, 2023. URL: https://lutpub.lut.fi/handle/10024/166173 (visited on 12/22/2023).

[140] Arun Vishwanath, Fatemeh Jalali, Kerry Hinton, Tansu Alpcan, Robert W. A. Ayre, and Rodney S. Tucker. "Energy Consumption Comparison of Interactive Cloud-Based and Local Applications". In: *IEEE Journal on Selected Areas in Communications* 33.4 (Apr. 2015), pp. 616–626. ISSN: 0733-8716. DOI: 10.1109/JSAC.2015.2393431.

[141] Leonhard Wattenbach, Basel Aslan, Matteo Maria Fiore, Henley Ding, Roberto Verdecchia, and Ivano Malavolta. "Do You Have the Energy for This Meeting?: An Empirical Study on the Energy Consumption of the Google Meet and Zoom Android Apps". In: *IEEE/ACM International Conference on Mobile Software Engineering and Systems*. Pittsburgh Pennsylvania, May 2022. DOI: 10.1145/3524613.3527812.

[142] Briag Dupont, Nesryne Mejri, and Georges Da Costa. "Energy-Aware Scheduling of Malleable HPC Applications Using a Particle Swarm Optimised Greedy Algorithm". In: *Sustainable Computing: Informatics and Systems* 28 (Dec. 2020), p. 100447. ISSN: 22105379. DOI: 10.1016/j.suscom.2020.100447.

[143] Dror G. Feitelson. "Resampling with Feedback: A New Paradigm of Using Workload Data for Performance Evaluation". In: *Job Scheduling Strategies for Parallel Processing*. Ed. by Dalibor Klusáček, Walfredo Cirne, and Gonzalo P. Rodrigo. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2021, pp. 3–32. ISBN: 978-3-030-88224-2. DOI: 10.1007/978-3-030-88224-2_1.

[144] Elizabeth Shove. "Beyond the ABC: Climate Change Policy and Theories of Social Change". In: *Environment and Planning A: Economy and Space* 42.6 (June 2010), pp. 1273–1285. ISSN: 0308-518X, 1472-3409. DOI: 10.1068/a42282.

[145] Albert Bandura. "Toward a Psychology of Human Agency". In: *Perspectives on psychological science* 1.2 (2006), pp. 164–180. DOI: 10.1111/j.1745-6916.2006.00011.x.

[146] Ian Brooks, Minna Laurell Thorslund, and Aksel Biorn-Hansen. "Tech4Bad in the Oil and Gas Industry: Exploring Choices for ICT Professionals". In: *ICT4S 2023: International Conference on ICT for Sustainability*. Rennes, France: IEEE, 2023, pp. 142–153. ISBN: 9798350311099. DOI: 10.1109/ICT4S58814.2023.00023.

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

Université de Toulouse

Titre : Étude de comportements de sobriété dans les centres de calculs

Mots clés : Centres de calcul, Gestion de l'énergie, Responsabilisation utilisateur, Green IT, Informatique soutenable, Sobriété

Résumé : L'industrie des technologies de l'information a une empreinte carbone croissante (2,1 à 3,9 % des émissions mondiales de gaz à effet de serre en 2020), incompatible avec la décarbonation rapide nécessaire pour atténuer le changement climatique. Les centres de données y contribuent significativement en raison de leur consommation d'électricité : 205 TWh, soit 1 % de la consommation mondiale d'électricité en 2018. Pour réduire cette empreinte, les travaux de recherche se sont principalement concentrés sur les mesures d'efficacité énergétique et l'utilisation des énergies renouvelables. Si ces travaux sont nécessaires, ils entraînent également des effets rebond, à savoir une augmentation de la demande en réponse aux gains d'efficacité. Pour cette raison, il apparaît essentiel de les accompagner de mesures de sobriété, c'est-à-dire d'une utilisation raisonnée de ces technologies, afin de diminuer la consommation d'énergie et de ressources en valeur absolue. Dans cette thèse, nous présentons un modèle de centre de données et de ses utilisateurs. Dans une première partie, nous nous concentrons sur les utilisateurs directs, qui interagissent avec l'infrastructure en soumettant des tâches. Nous définissons cinq leviers de sobriété qu'ils peuvent adopter pour réduire leur impact sur l'infrastructure, à savoir Délai, Reconfiguration, Dégradation Spatiale, Dégradation Temporelle et Renoncement. Nous caractérisons ces leviers à l'aide de simulations sur des données réelles. Les résultats nous permettent de les classer en fonction de leur potentiel d'économie d'énergie, de leur impact sur les métriques d'ordonnancement et de l'effort requis de la part des utilisateurs. L'un des inconvénients des leviers de sobriété est leur inertie, que nous expliquons à l'aide de métriques ad hoc. Nous étudions ensuite le potentiel des leviers dans un contexte de gestion des énergies renouvelables. Nous montrons que l'adoption des leviers de sobriété en période de faible production renouvelable conduit à des économies d'énergie non renouvelable. Les économies sont proportionnelles aux efforts fournis par les utilisateurs. Dans une deuxième partie, nous nous appuyons sur notre modèle d'utilisateur et son implémentation pour aborder un problème ouvert dans la simulation de systèmes distribués. La plupart des travaux utilisent des traces réelles pour simuler les soumissions dans l'infrastructure, en rejouant des tâches ayant les mêmes caractéristiques et les mêmes temps de soumission. Toutefois, ce modèle pose problème lorsque les performances simulées diffèrent de celles de l'infrastructure d'origine. Nous modélisons et mettons en œuvre le "rejeu avec feedback", une façon d'utiliser les traces réelles, en préservant le temps de réflexion entre les tâches plutôt que les temps de soumission originaux. Nous fournissons une analyse approfondie de l'impact de notre méthode à l'aide de nouvelles métriques. Dans une dernière partie, nous nous concentrons sur les utilisateurs indirects des centres de données, en étudiant des utilisateurs de cloud professionnel. Nous menons une étude qualitative afin d'examiner ce que signifierait, en pratique, une utilisation sobre du cloud. L'étude comprend trois focus groups analysés par le biais d'une analyse thématique. Les résultats dressent une image préliminaire de la nature de nos besoins professionnels numériques, ainsi qu'une liste de "tactiques vers la sobriété", c'est-à-dire d'actions concrètes pour se concentrer sur l'essentiel tout en limitant son empreinte environnementale. Ce manuscrit offre un aperçu de la sobriété numérique dans les centres de données, impliquant à la fois la simulation et les sciences sociales. Nous espérons que nos codes libres et nos campagnes de simulation reproductibles seront utiles pour de futurs travaux dans cette direction.

Title: Digital Sufficiency in Data Centers: Studying the Impact of User Behaviors

Key words: Data centers, Energy aware, User empowerment, Green IT, Sustainable Computing, Sufficiency

Abstract: The Information Technologies (IT) industry has an increasing carbon footprint (2.1-3.9% of global greenhouse gas emissions in 2020), incompatible with the rapid decarbonization needed to mitigate climate change. Data centers hold a significant share due to their electricity consumption amounting to 1% of the global electricity consumption in 2018. To reduce this footprint, research has mainly focused on energy efficiency measures, and use of renewable energy. While these works are needed, they also convey the risk of rebound effects, i.e., a growth in demand as a result of the efficiency gains. For this reason, it appears essential to accompany them with sufficiency measures, i.e., a conscious use of these technologies, in order to decrease the total energy and resource consumption. In this thesis, we introduce a model for data centers and their users. In the first part, we focus on direct users, interacting with the infrastructure by submitting jobs. We define five sufficiency behaviors they can adopt to reduce their stress on the infrastructure, namely Delay, Reconfig, Space Degrad, Time Degrad and Renounce. We characterize these behaviors through simulation on real-world inputs. The results allow us to classify them according to their energy saving potential, impact on scheduling metrics and effort required from users. One drawback of sufficiency behaviors is their inertia, that we explain with appropriate metrics. We investigate thereafter the behaviors' usefulness in the context of renewable energy management. A three-state energy feedback mechanism informs the users on the status of electricity production. We show that adopting the sufficiency behaviors when renewable energy is scarce leads to brown energy savings. Savings are proportional to the efforts made by users. In a second part, we build upon our user model and implementation to tackle an open issue in distributed system simulation. Most works use recorded traces to simulate workloads, by replaying jobs of the same characteristics and same submission time. However, this model is problematic when the simulated performances differ from the original infrastructure. We model and implement "replay with feedback", a way of using recorded traces, preserving the think time between jobs rather than the original dates of submission. We provide an in-depth analysis of our method's impact with the help of novel metrics. In a last part, we shift our focus to indirect users of data centers by studying professional cloud users. We design and conduct a qualitative study to investigate what a sufficient use of the cloud would mean, in practice. The study involves three focus groups analyzed through thematic analysis. The results provide a preliminary picture of the nature of our digital professional needs, along with a list of "tactics towards sufficiency", concrete actions to focus on the essential while limiting environmental footprint. This manuscript offers an insight into digital sufficiency in data centers, involving both simulation and social sciences. We hope that our open-source code and reproducible simulation campaigns will be useful for future works in that direction.