Carbon-Aware Microservice Deployment for Optimal User Experience on a Budget

Kevin Kreutz*
kreutz@campus.tu-berlin.de
Technical University of Berlin
Berlin, Germany

Philipp Wiesner*
wiesner@tu-berlin.de
Technical University of Berlin
Berlin, Germany

Monica Vitali*
monica.vitali@polimi.it
Politecnico di Milano
Milan, Italy

Abstract

The carbon footprint of data centres has recently become a critical concern. So far, most carbon-aware strategies have focused on leveraging the flexibility of scheduling decisions for batch processing by shifting the time and location of workload executions. However, such approaches cannot be applied to service-oriented cloud applications, since they have to be reachable at every point in time and often at low latencies.

We propose a carbon-aware approach for operating microservices under hourly carbon budgets. By choosing the most appropriate version and horizontal scaleout for each microservice, our strategy maximizes user experience and revenue while staying within budget constraints. Experiments across various application configurations and carbon budgets demonstrate that the approach adapts properly to changing workloads and carbon intensities.

Keywords

Green Applications, Service Computing, Energy Efficiency

1 Introduction

Governing bodies like the European Union (EU) are activating several actions aiming at reaching carbon neutrality by 2050 [6]. The ICT sector plays an important role in this challenging task, contributing to 3.9% of the overall global emissions with an increasing trend over the years [3]. To this aim, the European Green Digital Coalition (EGDC) has involved several companies in the joint effort to measure and reduce the environmental impact of ICT, with a target for carbon neutrality by 2040 [4]. Similarly, the EU Emissions Trading System (EU ETS) [5] promotes carbon neutrality targets by capping the total amount of emissions allowed for specific sectors.

Cloud providers have announced carbon neutrality goals [1, 7, 11]. Although energy efficiency has improved at the infrastructure level, cloud emissions continue to rise due to the increasing demand for computational power from hosted applications. It is therefore important to reduce the environmental impact of these applications. To achieve this, enhancing applications with adaptive features [10, 15] can help reduce their emissions when needed.

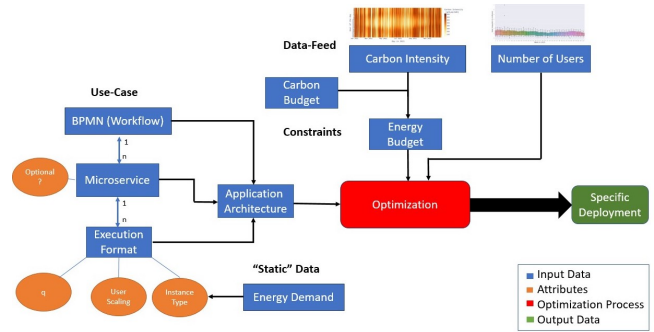*All authors contributed equally to this research.

Figure 1: Overview of the Approach

In the proposed approach, we apply the concept of hourly budgets of carbon emissions or time-based renewable energy certificates [9, 14] to microservice architectures. We propose an algorithm able to continuously reconfigure the microservice deployment according to the current budget, the energy mix of the electricity grid, and the workload. We show that our approach can satisfy the budget constraints while maximizing the user experience and the revenue for the application owner.

2 Approach

The proposed approach includes carbon-awareness in the deployment process of long-running cloud applications, leveraging flexibility [2, 8, 12, 13, 17, 18] to adjust application deployment and scaling based on the current carbon intensity of the electricity grid.

An overview of the approach is shown in Figure 1. At its core is an optimisation algorithm, which selects the best configuration of an application to avoid exceeding the carbon budget while optimising other non-functional requirements, such as the Quality of the Experience (QoE) of the users and the revenue of the application owner. In this section, we provide additional details of the components of the approach and we model the objective function used for the optimisation.

Application Architecture. The type of applications considered in this work are long-running cloud-native applications composed of multiple microservices. The application workflow can be modelled and enriched using BPMN as discussed in [15]. Each microservice can be provided in multiple versions, each one with different characteristics in terms of energy demand and quality of experience provided to the users. For instance, a low-power version of a microservice might remove some computationally demanding functionalities affecting the overall satisfaction of the user. A microservice can also be annotated as optional. Optional microservices
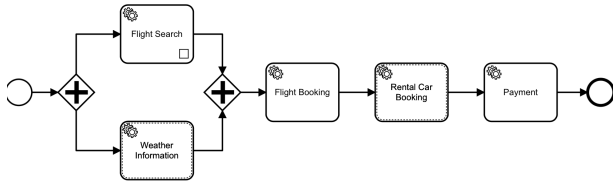
Figure 2: Flight Booking application in BPMN

Microservice	Versions	Optional	Instance-Type	ED	q	uc	QoE	rev
	Low Power		t3.micro	13.0	0.5	20000 or 5000	0.5 or 0.1	0
Flight Search	Normal	no	t3.xlarge	39.9	0.7	20000 or 5000	0.75 or 0.3	0
	High Performance		g2.2xlarge	305.4	0.9	20000 or 5000	1	0
Weather Information	Off	yes		0.0	0.9		0	0
	Normal		t3.micro	13.0	1.0	20000 or 5000	1	0.2
Flight Booking	Low Power	no	t3.micro	13.0	0.5	20000 or 5000	0.5 or 0.1	0
	Normal		t3.xlarge	39.9	0.95	20000 or 5000	1	0
Rental Car Booking	Off	yes		0.0	0.75		0	0
	Normal		t3.xlarge	39.9	0.9	20000 or 5000	0.75 or 0.3	2
	High Performance		g2.2xlarge	305.4	1.0	20000 or 5000	1	2
Payment	Normal	no	t3.xlarge	39.9	0.99	20000 or 5000	1	0

Table 1: Flight Booking microsevices' annotations

are not necessary to reach the overall goal of the application. Still, they can increase the QoE of the user and/or generate rewards for the application owner (e.g., advertisement).

Energy Budget. It expresses the carbon-related information including the carbon budget assigned to the application and the carbon intensity of the energy grid. We assume the carbon budget is assigned yearly but it is dynamically split hourly according to the expected workload. The carbon intensity is the current mix of green and brown energy in the electricity grid. This value can be used to compute how much energy can be used in the time slot without exceeding the carbon budget.

Number of Users. The application can be associated with an estimated number of requests (workload) during the year according to the information collected in the previous years.

The objective function can be expressed as in Equation 1

$$\max \left(\alpha \sum_{ms \in CA} \frac{QoE_{ms}}{\#ms} + \beta \sum_{ms \in CA} \frac{rev_{ms}}{revMax} \right), \quad (1)$$

where QoE_{ms} is the QoE associated with a microservice version, rev_{ms} is the revenue for the application owner associated with a microservice, α and β are two weights associated to the relevance of these to aspects for the application owner. The complete implementation of the optimization algorithm is available to the public[1].

3 Experiments and Results

We validated the approach with a simple application, shown in Figure 2 using an enriched BPMN [16]. The application consists of five microservices. Two of them (*Weather Information* and *Rental Car Booking*) are optional. The *Flight Search* microservice is instead provided with multiple versions. The specific annotations of the five microservices are summarised in Table 1, including versions and the optional label. For each version, the required instance type for deployment is specified along with the estimated energy demand (*ED*). Each version is linked to a revenue (*rev*, applicable

---

[1]https://github.com/POLIMIGreenISE/carbonbudget.git
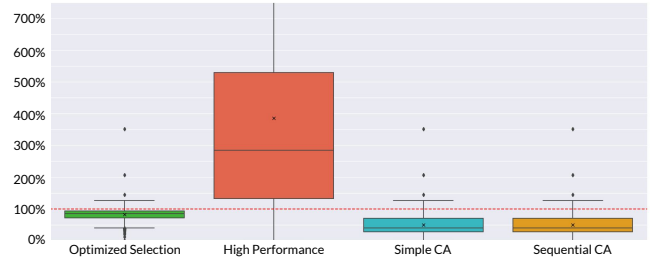

Figure 3: Carbon budget utilisation

only to optional functionalities) and a *QoE* value. The parameter *uc* represents the number of user requests handled by a single instance, while *q* denotes the percentage of users who move on to the next microservice. In the experiments, most parameters were kept fixed, while varying the values of *uc* and *QoE*.

The proposed algorithm (*Optimal Selection (OS)*), is designed to choose the best configuration for each microservice, ensuring budget constraints are met while optimizing the objective function (Equation 1). It is compared against three baselines: (i) *High Performance (HP)* - all microservices are deployed in high-performance versions and optional microservices are deployed by default; (ii) *Sequential Carbon-Aware (SCA)* - the application adjusts based on the available budget using three fixed sequential configurations for all microservices; and (iii) *Simple Carbon-Aware (CA)* - similar to *SCA*, but the three configurations do not have to be sequential.

Experiments were conducted using different carbon budgets. For instance, Figure 3 shows the results obtained with a carbon budget set between the emissions needed to operate the application in high-performance mode and those required for low-power mode over the course of a year. As can be observed, the *HP* baseline significantly violates the carbon budget, while all the other solutions can fulfil the budget constraints. The proposed approach effectively maximizes budget utilization, resulting in higher revenue and QoE. Compared to the carbon-aware baselines, the OS algorithm delivers an average QoE improvement of 5.44% and a revenue increase of 32.38%.

4 Conclusion and Future Work

We proposed a carbon-aware approach to adapt the operation of cloud-native applications based on the availability of renewable energy, a carbon budget, and the current workload, while maximizing both QoE and revenue. Our approach was tested under dynamic workloads and fluctuating energy mixes, demonstrating its capability to meet the carbon budget while maximizing its utilization, leading to higher QoE and revenue. In future work, we plan to explore additional configurations and experiment with different weights for revenue and QoE in the objective function.

References

[1] Amazon. 2023. Amazon Sustainability Report. https://sustainability.aboutamazon.com/2023-amazon-sustainability-report.pdf

[2] Tayebeh Bahreini, Asser Tantawi, and Alaa Youssef. 2022. An Approximation Algorithm for Minimizing the Cloud Carbon Footprint through Workload Scheduling. In *2022 IEEE 15th International Conference on Cloud Computing (CLOUD)*. 522–531. https://doi.org/10.1109/CLOUD55607.2022.00075

[3] European Commission. 2024. Rolling Plan for ICT standardisation. https://joinup.ec.europa.eu/collection/rolling-plan-ict-standardisation/ict-environmental-impact-rp2024

[4] Ministerial Declaration. 2021. A Green and Digital Transformation of the EU.

[5] Ministerial Declaration. 2023. European Trading System Directive. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:02003L0087-20230605

[6] Constanze Fetting. 2020. The European green deal. *ESDN Report, December* 2, 9 (2020).

[7] Google. 2024. Environmental Report. https://www.gstatic.com/gumdrop/sustainability/google-2024-environmental-report.pdf

[8] Conor Kelly, Eleni Mangina, and Antonio Ruzelli. 2011. Putting a CO2 figure on a piece of computation. In *11th International Conference on Electrical Power Quality and Utilisation*. 1–7. https://doi.org/10.1109/EPQU.2011.6128960

[9] Michael Kelly and David Gonzalez. 2023. *The Role of Hourly EAC Markets in Facilitating the Clean Energy Transition.* Technical Report. Google.

[10] Justin J Meza, Thote Gowda, Ahmed Eid, Tomiwa Ijaware, Dmitry Chernyshev, Yi Yu, Md Nazim Uddin, Rohan Das, Chad Nachiappan, Sari Tran, et al. 2023. Defcon: Preventing Overload with Graceful Feature Degradation. In *17th USENIX Symposium on Operating Systems Design and Implementation (OSDI 23)*. 607–622.

[11] Microsoft. 2024. How can we advance sustainability? 2024 Environmental Sustainability Report. https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RW1lMjE

[12] Tobias Piontek, Kawsar Haghshenas, and Marco Aiello. 2023. Carbon emission-aware job scheduling for Kubernetes deployments. *The Journal of Supercomputing* (06 2023), 1–21. https://doi.org/10.1007/s11227-023-05506-7

[13] Thanathorn Sukprasert, Abel Souza, Noman Bashir, David E Irwin, and Prashant J Shenoy. 2023. Quantifying the benefits of carbon-aware temporal and spatial workload shifting in the cloud. *CoRR* (2023).

[14] Maud Texier. 2022. Timely progress towards around-the-clock carbon-free energy. Google. https://cloud.google.com/blog/topics/sustainability/t-eacs-help-drive-around-the-clock-carbon-free-energy

[15] Monica Vitali. 2022. Towards greener applications: enabling sustainable-aware cloud native applications design. In *International Conference on Advanced Information Systems Engineering*. Springer, 93–108.

[16] Monica Vitali, Paul Schmiedmayer, and Valentin Bootz. 2023. Enriching Cloud-native Applications with Sustainability Features. In *2023 IEEE International Conference on Cloud Engineering (IC2E)*. IEEE, 21–31.

[17] Philipp Wiesner, Ilja Behnke, Dominik Scheinert, Kordian Gontarska, and Lauritz Thamsen. 2021. Let's Wait Awhile: How Temporal Workload Shifting Can Reduce Carbon Emissions in the Cloud. In *22nd International Middleware Conference*. ACM. https://doi.org/10.1145/3464298.3493399

[18] Philipp Wiesner, Dennis Grinwald, Philipp Weiß, Patrick Wilhelm, Ramin Khalili, and Odej Kao. 2024. Quality Time: Carbon-Aware Quality Adaptation for Energy-Intensive Services. *arXiv preprint* (2024).