Aalto University
School of Science
Master's Programme in Security and Cloud Computing

Murali A. Abinaov

Simulation framework for carbon-aware workload shifting in the cloud

Master's Thesis
Espoo, October 24, 2024

Supervisors: Assistant Professor Gopika Premsankar, Aalto University
Advisor: Professor György Dán PhD, Kungliga Tekniska högskolan (KTH)

Aalto University
School of Science
Master's Programme in Security and Cloud Computing

ABSTRACT OF MASTER'S THESIS

Author:	Murali A. Abinaov		
Title:	Simulation framework for carbon-aware workload shifting in the cloud		
Date:	October 24, 2024	**Pages:**	65
Major:	Security and Cloud Computing	**Code:**	SCI3042
Supervisors:	Assistant Professor Gopika Premsankar		
Advisor:	Professor György Dán PhD		

Carbon-aware workload shifting algorithms are a method to utilize the global scale and flexibility of computing resources offered by cloud computing concepts to minimize the greenhouse gases emitted from the computation tasks. Specifically, the same computational work can produce more or less emissions depending on the geographic location or time when the work is done. Some workload shifting algorithms have already been implemented by major cloud providers like Google, but they are continually iterated upon and improved. This results in a myriad of workload shifting algorithms with different performances and specialties evaluated in custom simulators with different network layouts.

In our thesis, we design and develop a simulation framework which can test and compare carbon-aware workload shifting algorithms to aid their development and to determine what scenarios a given algorithm may perform well in. The performance of the algorithms are measured via the amount of carbon dioxide equivalent greenhouse gases produced during a simulation. Some tools already exist to compare algorithms, but they often are highly specialized or put a significant emphasis on the specific resources available in simulated data centers.

A core aspect of the framework is the interchangeable carbon intensity data and workloads. The interchangeability of these components save time during the development of workload shifting algorithms by enabling experiments using carbon intensity data from different sources and different workloads. There are common formats for the carbon intensity data and the workloads, a data frame column structure for the former, and a custom API data frame extension for the latter. The common formats ease the compatibility between a given algorithm and the workloads or carbon intensity data not custom designed specifically for the given algorithm. Additionally, this enables two or more algorithms to utilize the same workload without the workload needing to be custom implemented for each algorithm, which in turn supports comparing the algorithms during development.

Keywords:	Carbon-aware workload shifting algorithms, Cloud computing, Green computing, Data centers, Simulation framework
Language:	English

Acknowledgements

My thesis work would not have been possible without the guidance, direction, and feedback provided by my Supervisor Gopika Premsankar. I am also thankful for my Advisor György Dán and Examiner Viktoria Fodor for giving additional feedback or answering questions regarding the thesis administrative process. Many thanks to Professor Tuomas Aura and the Aalto University Computer Science department for their generous support in funding my research. Additionally, my studies would not have been possible without the aid of the many groups and individuals involved in the Erasmus Mundus Programme SECCLO.

I would also like to extend my sincere thanks to Rajini Wijayawardana and the rest of the authors of Reducing the Carbon Impact of Generative AI Inference (today and in 2035) for responding to my questions on their paper. Many thanks should also go to Electricity Maps and WattTime for providing some of their marginal carbon intensity data for use in my research.

Lastly, special thanks to my family, friends, and fellow SECCLO students for keeping me motivated and on track throughout my time working on this thesis.

Espoo, October 24, 2024

With the support of the
Erasmus+ Programme
of the European Union

Murali A. Abinaov

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Data centers, the backbone of cloud computing infrastructure, consume an enormous amount of electricity, over 1% of the entire world's electricity usage in 2021 [21, 38]. This is a growing amount as 460 TWh, close to 2% of the world's electricity demand was consumed in 2022 [18]. This energy consumption contributes towards the climate crisis that has been afflicting our planet throughout the Anthropocene epoch [38, 42]. Data centers alone are expected to produce 8% of global carbon emissions by the end of the decade [23]. Furthermore, the rise of AI has exacerbated this as both Google and Microsoft say that their emissions have risen by 48% and 29%, respectively due to data centers supporting AI workloads [30]. To reduce energy usage efforts have been made to optimize how data centers are able utilize power for their operations, especially in regards to minimizing overhead (cooling, lights, etc.). Data centers are rapidly approaching a physical limit to the gains that can be achieved purely through optimization, as modern data centers use little power beyond that necessary to power their IT equipment [21, 28].
Another method to combat the climatic effects that data centers have, is to not focus on minimizing the electricity consumption of each data center, but directly focus on reducing the environmental costs from the cloud computing system as a whole. Electricity itself can vary in how much carbon was emitted from its production, depending on the time and location of said production [38, 41]. A utilization of this concept is carbon-aware workload shifting i.e., the movement of cloud computing tasks between data centers in accordance to the amount of carbon and other greenhouse gases emitted by the energy consumed by the data center to do the tasks [24, 38, 46]. As a number of implementations for workload shifting algorithms exist that are evaluated through custom simulations, this thesis provides a simulation framework to compare the algorithms. The value of this comes from providing a common framework on which aspects can be interchanged and tested,

such as cloud computing workloads or carbon data. The framework can be further expanded upon by adding new data sources, more workloads, and workload shifting algorithms developed in the future.

1.1 Problem, Purpose, & Goals

While in our research we have found some tools that already exist in similar areas that can compare workload shifting algorithms, they are usually specialized or focused in other aspects. This means that they are not suitable to emulate a large variety of algorithms to provide side-by-side results of the algorithms. We endeavored to develop a more lenient and wide reaching way to answer some questions that existing tools may not respond to as easily.

1. How well do existing workload shifting algorithms minimize carbon dioxide emissions?
2. How do different network layouts affect the ability of the algorithms to reduce emissions?
3. How do the algorithms behave under different metrics for measuring the carbon intensity of the electricity powering data centers?

The objective of this thesis is to build the tool to answer these questions, by developing a simulation framework. Within this simulator, we shall implement a few workload shifting algorithms and their workloads, while leaving it open for others to add additional algorithms or workloads. We aim to keep the implemented algorithms as close to the originals as possible, while simultaneously opening them up to run experiments with different types of carbon intensities, time steps, or workloads than might have originally been intended.

1.2 Ethics and Sustainability

The aims of this thesis are closely intertwined with the sustainability of cloud computing, as the ultimate goal is to minimize the emissions of greenhouse gases. The simulation framework provides a way for companies and organizations to compare algorithms against each other in different situations. This will aid future development of workload shifting algorithms. Better workload shifting algorithms will help organizations from small commercial businesses to countries achieve their environmental goals.

An ethical consideration is that this may also provide a path for businesses to further commercialize their algorithms by aiding them in proving their algorithms are better than their competitors. In this regard, the framework may be abused by unfairly making an algorithm appear ineffective due to it running in an experiment with unsuitable conditions, whether intentionally or otherwise.

1.3 Research Methodology

Initially a literature review was conducted on carbon-aware data centers, workload shifting algorithms, and carbon metrics. Through this literature review, algorithms were gathered for implementation and types of carbon intensities were defined for use in the simulator. Commonalities found between algorithms were the among the driving forces for the use of a discrete event implementation methodology, to ease the assimilation of many existing carbon-aware workload shifting algorithms into the simulator. Further details on the methodology can be obtained from Section 3.1.

1.4 Delimitations

Outside of the scope of this thesis is the development of new workload shifting algorithms. Nor will we be attempting to define new carbon emission metrics. We will not be discussing the energy optimization of cloud workloads from the management of individual data centers. The regulatory role which governments or commercial coordination may play in limiting carbon costs across the industry are also an aspect outside of the thesis focus. Topics such as energy trading on a national or international scale are accounted for in some data sources [14], but are otherwise excluded from the thesis. While it is not entirely discounted, there is a lesser degree of emphasis on embodied emissions from such things as construction and manufacturing, than there is on operational emissions. Embodied emissions are included in some of the metrics and algorithms, but we do not discuss the operational and embodied emission split during evaluation.

1.5 Structure of the Thesis

Chapter 2 provides relevant background information regarding carbon-aware workload shifting algorithms. Chapter 3 presents the methodology, design, and implementation of the simulation framework. Chapter 4 shows the experiments run with the algorithms implemented in the simulator. Finally, Chapter 5 describes our final thoughts and avenues for further work in the future.

Chapter 2

Background

This chapter provides the reader with a background understanding on sustainable cloud computing and gives context for the simulation framework developed in Chapter 3 and analyzed in Chapter 4. It starts with a discussion of the types of carbon intensities. This is followed by denoting the sources for the carbon intensity data we used and the types of emissions that exist. We also examine how data centers limit their carbon impact. Finally we give an overview of some related projects in the space of carbon-aware cloud computing simulation.

2.1 Carbon Intensity

Carbon intensity is the measure of grams of carbon dioxide equivalent generated per kilowatt hour of electricity generated, it is largely synonymous with emission intensity or emission rate [7]. Carbon intensity is generally a better measurement for how green a grid is at a given time than the actual amount of available renewable energy, as it can be difficult to choose to use the energy of a specific source. Renewable energy is usually mixed into the electricity of the grid and its subtraction from the general grid power often just results in other consumers on the grid utilizing a higher proportion of brown energy [35]. This results in a given grid supplied with a large amount of renewable energy potentially producing higher emissions than one with less renewable energy, if it is a small fraction of the total energy in the grid. Furthermore, carbon emissions do not necessarily refer to CO_2 alone, as electricity production often emits other greenhouse gases, such as methane, which also contribute to climate change to a greater of lesser degree per unit of gas compared to CO_2 [8]. As the effect of the greenhouse gases on the climate are the important aspect, rather than the specific gases, the gases are all converted to the equivalent amount of carbon dioxide that would be required to have the same global warming impact, per unit of power produced, or the carbon intensity [8].

The metric usually consists of the carbon intensity of an electricity grid or region, but there are a multitude of variants of carbon intensity that can be utilized [27, 35]. The carbon intensity can vary depending on what assumptions are made and what aspects are considered in its calculations. In this thesis we focus on discussing and comparing the impact of location-based average and location-based marginal carbon intensity, due to the relatively high availability of data for these two metrics [10, 11]. Other metrics, such as market-based average and long-run marginal carbon intensity are also noted to demonstrate some short-falls of the prior metrics.

2.1.1 Location-based Versus Market-based

One aspect that can differ is whether the carbon intensity is location-based or market-based, both of which are accepted by the Greenhouse Gas Protocol [35]. Location-based carbon intensity makes the assumption that all consumers on the electricity grid at a given location consume the same blend of electricity proportional to what the various providers supplied the grid [35]. Renewable Energy Credits or Power Purchase Agreements, the direct subsidization of green energy providers by consumers by purchasing the agreement to produce renewable power into a given electricity grid [43], are not taken into consideration by Location-based carbon intensity, while Market based does. Market-based carbon intensity attempts to take the energy produced under Renewable Energy Credits and Power Purchase Agreements out of account, by subtracting it from the electricity produced in a location before calculating the carbon intensity of that location. This avoids the power from Power Purchase Agreements being claimed multiple times due to discrepancies from their buyers reporting market-based carbon intensity to show their reduced impact, while other organizations utilize location-based intensity [35]. As of this writing, location-based carbon intensity is usually used in workload shifting algorithms [24, 46], as the availability of market-based data is quite limited and is often still being produced and compiled by the companies tracking such data [10, 11].

2.1.2 Average Versus Marginal

In addition to whether the carbon intensity is location or market based, it could also be either the average or the marginal carbon intensity.

$$CI = \frac{\sum_i (c_i \cdot w_i)}{\sum_i w_i} \tag{2.1}$$

The average carbon intensity, CI, as defined by [40], is calculated via the emission factors per generator, c_i, weighted by how much the generator contributes to supplying the given electrical grid, w_i (2.1). To summarize average carbon intensity is determined by dividing the total emissions by the total energy produced within the grid, while marginal is the carbon intensity of the specific electricity source used to handle additional loads on the grid [27]. The reason average and marginal carbon intensity can differ significantly owes to the fact that generators have a limited capacity for generation, so the same generators used to supply prior loads, may not be able to do the same for new loads, which has major consequences for determining the environmental impact of additional strains on a electrical grid. This is due to the marginal generator that provides for new loads primarily being a fossil fuel generator as they are more reactive. Their electricity production can be more easily scaled up or down to meet demand compared to many renewable generators, such as wind and solar, which are dependent on weather. Additionally, some sources are better suited to provide for base-load or continuous operation, as frequently starting and stopping coal or nuclear power plants is expensive and can damage them [32]. Natural gas and hydroelectric plants are usually used to provide marginal power, as their production can be more easily manipulated to increase or decrease generator in accordance with the requirements of the grid [32]. Three countries which illustrate a real world example of the difference between average and marginal carbon intensity are Brazil, France, and Ireland. Ireland has a high capacity for wind power, but when the wind of a given day is insufficient, the remaining electricity is likely provided by the next highest source of power in the country, natural gas [13], resulting in a high marginal and sometimes average carbon intensity when the wind at a given time is low. France, on the other hand primarily uses nuclear power to supply their base-load needs, but as nuclear power cannot be increased very easily, they probably rely on the next two largest sources, natural gas and hydro [13] to provide the marginal power. This means that France usually has a very low average and high marginal carbon intensity. Meanwhile, Brazil uses a combination of wind and large amounts of hydroelectricity, so if there is not enough wind at some time, low carbon hydro generators can be easily scaled up to meet the difference [13].

2.1.3 Short-run Versus Long-run

The downside to marginal carbon intensity is that it is usually focused on the short-term (short-run marginal carbon intensity) and assumes that the grid structure is static [25, 27]. It is unable to capture long-term structural benefits from working at certain times, such as when low carbon energy is produced, when the marginal carbon intensity is the same at both times either due to already being fully utilized or not being produced at such times [25]. An example of a long-term benefit is that shifting most workloads to the day, will promote the construction of additional solar generators, decreasing the carbon intensity over time [25, 27]. To work around this oversight, long-run marginal carbon intensity can be used which model the future to account for changes in the grid [27].

2.2 Carbon Footprint

Operational emissions in cloud data centers refer to the emissions that occur from directly doing computational work, while embodied emissions are the more hidden emissions that occur from fuel, infrastructure, and end-of-life operations [28].

2.2.1 Operational Emissions

The most apparent source of emissions from data centers is the electricity needed for their operation. Data centers use a considerable amount of energy, in the United States alone, they accounted for 1.5% and 1.8% of the country's total electricity consumption, in 2007 and 2017 respectively [23, 42]. Elsewhere, such as Ireland in 2022, data centers represented 14% of national electricity usage [33]. This large consumption is even despite advancements in the efficiency of data centers that bring their power usage effectiveness to a nearly optimal PUE of 1.1 [38] or in other words, only utilizing the power required by the IT equipment itself with very little additional power needed by overhead or cooling [28]. The carbon emissions this electricity consumption represents can be significantly reduced depending on the time and location where the computation work occurs [38]. This is due to numerous sources of electricity with different carbon intensities are often used to supply a given grid, and their capacity for electrical production can be highly variable dependent on time [38, 41]. For example, California would have a much lower carbon intensity on a sunny day versus at night, due to its high solar power infrastructure capacity and need to use less natural gas power, while Denmark would likewise produce less carbon emissions during windy weather, as they have a large amount of installed wind power infrastructure [13]. These temporal and spatial differences in the carbon intensity enable computation work to be moved in order to minimize carbon emissions. Electricity storage is often utilized to help balance out the demand and supply within a given grid, but it is often insufficient [8]. Due to the relative lack of energy storage infrastructure, real-time energy production often has to match the real-time operational demand [32].

2.2.2 Embodied Emissions

Beyond the cost incurred from simply operating a data center, there exists a significant environmental cost from constructing, manufacturing, and supplying the necessary resources for a data center. These embodied emissions constitute around 50% of total emissions generated by a given data center [44]. Embodied emissions have been trending upward across computing industries as demand for hardware rises and modern equipment require a growing number of transistors [28]. The embodied cost has been growing at a faster rate than even the operational cost. An example of this is that the energy consumption of data centers rose by 6% between 2010 and 2018, while the number of compute instances increased six times over in the same period [28].

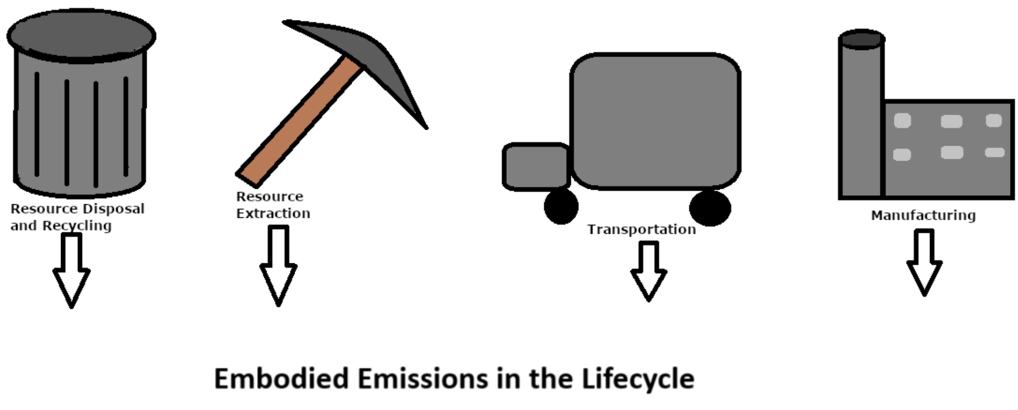


Figure 2.1: Operations that occur during the lifespan of a data center which cause embodied emissions.

Supply-Chain As efforts go towards optimizing energy usage an ever growing percentage of the total emissions of data centers originate from the supply-chain. For 2019, both Google and Meta reported that the majority of the impact left by their data centers of can be derived from the supply-chain, such as upstream manufacturing and transportation [28]. Much of the manufacturing of semiconductors and other hardware also occurs within countries that tend to be more reliant on brown energy sources [31] and from where the equipment must be transported to the data center, tacking on additional carbon costs. These supply-chain costs are easily obscured so, significant efforts must be made to collect and monitor data, so supply-chains can be coordinated and optimized to reduce their environmental impact [49]. This can be solved by life cycle assessments which are key for properly accounting for all emissions upstream and downstream, then localizing them to their consumers [28, 29]. Life cycle assessments are useful not just for accounting for the true environmental costs of data centers, but also for the origins of the power used by the centers themselves. The aspects that are covered by life cycle assessments are the extracting of raw materials, the construction of the generators and other related infrastructure, and the disposal or recycling of material [14].

2.2.3 Data Sources for Carbon Intensity

The exact carbon intensity can be difficult to determine due to the numerous emission factors that may or may not be included in the calculation, especially in near real-time, as they are often reliant on forecasting models [14, 15]. This is even more the case in regards to many of its sub-classifications, such as marginal carbon intensity [15, 27]. These differences often result in the exact same type of carbon intensity for identical location and time, often diverging between sources.

WattTime WattTime provided access to some of their marginal carbon intensity data to us for research use. The marginal carbon intensity data is output from an analytical model which takes into account time series data of: power plant emissions and generation data; data by fuel type of demand, interchange, and generation; energy prices and curtailment volumes; global power plant locations and fuel types; and weather data [15]. Due to the large amount of input data used to determine the carbon intensity, and the fact that some regions are inconsistent in providing data, missing data can be estimated with regression models or the averages from neighboring regions [15].

Electricity Maps Electricity Maps gives freely available access to their average carbon intensity data, which they additionally supplemented by providing us access to some of their marginal carbon intensity data for research use. The carbon intensity is computed by a number of factors, including power plant emission and generation data, which vary by region, similar to WattTime [14, 15]. Unlike WattTime however, specific effort is placed upon tracking and attributing the embodied emissions that occur from energy generation via a life cycle analysis and flow-tracing the origin of electricity in their average and marginal carbon intensities [14].

2.3 Carbon Awareness in Cloud Computing

Cloud computing is, as defined by the National Institute of Standards and Technology, “a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction” [36]. The shared pool of computing resources are usually in a different physical location than the applications that request their usage. Depending on the situation, this could result in a reduction in the total emissions due to reasons such as optimization from economies of scale [28], or it might cause a rise in net emissions due to the sudden increased demand placed upon a grid [39].

2.3.1 Carbon Offsets

A method through which cloud providers often attempt to ’offset’ the emissions they produce is via the purchase of renewable energy credits or carbon credits [23, 33]. Several of these carbon offset strategies have been criticized for being ineffective or even exacerbating the issue and causing a net increase in emission production [22]. Though not specific to cloud data centers, it has been estimated that over half of the projects built by the Clean Development Mechanism, the largest carbon offset program, would have been constructed anyway. This means the program may have instead increased emissions by 6.1 billion tons, as regulated polluters produced more emissions than they were allowed under the assumption that some would be successfully offset [22]. This does not mean carbon credits are entirely useless or harmful, they can still be utilized beneficially and will continue playing an important role in reducing global emissions if they are better regulated [22]. Their use in subsidizing renewable energy production can be regulated to make them more effective, and data centers that produce some of their own renewable electricity can sell excess production back to the grid to offset their carbon [22, 23]. The latter method would truly transfer renewable power, rather than be an accounting trick [33] or a type of voluntary fine businesses pay to appear more 'green' without changing their future behaviour.

2.3.2 Workload Shifting

Workload shifting refers to both moving computational work between data centers in different locations, spatial shifting, and changing the time computational work is done, temporal shifting. Carbon-aware workload shifting can include both types, but emphasizes reducing the carbon emissions produced by the computational work.

Carbon-aware workload shifting is heavily based on the energy source composition of the grid sustaining a given data center at a given time. This means that some data centers would be supplied more plentifully by green energy than others at different times, as renewable sources, like wind and solar, are often highly variable [37, 38]. In addition to using primarily low carbon energy, the minimization of peak demands in data centers, by spreading the jobs out over time, would result in less of an environmental impact, as fewer resources would be needed by each data center to maintain support for as large of a potential max capacity, resulting in less embodied costs for many data centers [38]. By shifting the when and where workloads are done on based on a metric such as carbon intensity, and limiting peaks, the total emissions produced can be significantly reduced [37–39].

An example of carbon-aware cloud workload shifting in practice is Google's Carbon-Intelligent Compute Management system which shifts non-urgent, temporally flexible tasks to when renewable energy is more plentiful [16, 38]. While services that need to be processed immediately, like Google Maps, Search, and YouTube, are unaffected by the system, it does affect more time-independent tasks like machine learning, simulations, and video processing [38]. The framework detailed in this paper aims to primarily simulate more generalized algorithms than those in use by major companies at the moment.

2.3.2.1 CarbonMin

The first algorithm implemented in the simulation framework was CarbonMin, a greedy request redirection algorithm created to model the affects of carbon-aware spatial shifting on Generative AI applications like ChatGPT over time [24]. CarbonMin operates by moving the received requests in each

time interval to the regions with the lowest carbon intensity during that interval, filling the estimated capacity of each region in the order of lowest carbon intensity first until every request has been fulfilled [24]. According to one of the authors of the paper[1], the latency due to the distance between the origin of a request and the location of a region the request is sent to, is not taken into account due to the relatively large delay expected in Generative AI request processing compared to the potential latency. Each request is assumed to require an estimated processing of about 2.07 GPU seconds, derived from several average statistics regarding ChatGPT requests [24].
The carbon emission model from the paper was combined with the request redirection model in order to give a more clear understanding of what the algorithm changes. Embodied and operational emissions per time interval are calculated in the emission model from the number of requests each data center facilitates, the carbon intensity each data center is operating under, and estimates from originating from the technical specifications of Azure and Google data centers [24].
In addition to CarbonMin, the original paper also detailed a workload model used to emulate the requests an application like ChatGPT might experience over twenty-four hours, which was also implemented within the framework. The model was originally designed to be able to create a realistic Generative AI workload for both the current day and a scaled up workload to simulate the future of 2035, so that the CarbonMin might compare its impact on the two. This Generative AI workload model distributes an estimated total number of global jobs over a day between selected relevant cities, according to their ChatGPT usage, population, and active hours [24].

2.3.2.2 Let's Wait Awhile

Let's Wait Awhile is the second workload shifting algorithm implemented within the simulation framework. Unlike CarbonMin, which focuses on spatially shifting workloads between data centers located in different regions but does not significantly change the timing of when the workload is done, Let's Wait Awhile focuses on temporally shifting workloads by altering the schedule in which the jobs in the workload are done for a single data center [24, 46]. It does this with the support of two additional simulation packages, Vessim and LEAF which are described in subsections 2.4.2 and 2.4.1. The former simulator aids in the scheduling of jobs, while the latter models the energy consumption [46].
LEAF models the infrastructure of a data center with a single node, from

[1] through email correspondence

which it is able to derive the Watts used at a given time from the number of active jobs scheduled at that time [46]. The emissions for each time interval are simply calculated by multiplying the active jobs during the interval with the carbon intensity of the data center at that time. Vessim is used to make time series forecasts based on the carbon intensity data fed into the algorithm, which is used to schedule the jobs.

In Let's Wait Awhile, there are two scheduling strategies. The first called periodic, is based on supporting periodic, scheduled batch jobs set to occur in the future, where the batch jobs could be scheduled to be done prior or later to when they are set to occur. The second strategy named ad-hoc, is designed to simulate ad-hoc machine learning training jobs, where the jobs can not be planned for ahead of time, and so each job can only be scheduled to be done as and when it arrives [46]. The ad-hoc strategy also enables choices for different forecast methods and whether the jobs are interruptible or not.

2.3.2.3 Workload Shifting with Green Energy Algorithm (WSG)

WSG is a spatial shifting algorithm which attempts to optimize for minimizing both carbon emissions and the response time of requests [48]. It is a candidate for future implementation that was excluded due to time constraints. The algorithm operates by sending each request to the data center with the least response time, provided that data center has sufficient clean energy. If the data center lacks the renewable electricity to fulfill the request, then the request is sent to the next closest data center with enough green energy. If no data center is available that both does not increase the response time too much and has sufficient renewable power, then the closest data center will work on the request with brown energy [48].

The focus on reducing the response time of requests in the workload contrasts with CarbonMin, which does not emphasize that aspect due to the relatively long response times expected from Generative AI requests [24]. Additionally, WSG attempts to work directly with the amount of renewable energy available on the grids, rather than just the carbon intensity each data center is operating under, as the workload shifting algorithms implemented in the simulation framework do [48]. It is for this latter reason that WSG was not chosen to be implemented yet, as the inclusion of an additional dataset increases the complexity and makes it more difficult to compare the results with other algorithms. Furthermore, it is difficult to only use the energy from a specific source, as all power supplying a given grid is mixed together and given in part to all demands placed on the grid or its specific reduction causes other consumers to use a larger proportion of dirty power [35].

2.3.2.4 Global Server Load Balancer (GSLB)

GSLB is an extremely light-weight algorithm, as it is intended to scale easily to keep up with a large number of requests without overburdening the system it is attached to with overhead [34]. It is another algorithm that could be implemented in the future but was not already due to limited time. Upon receiving a request, the algorithm scores all potential data centers based on the carbon intensity of electricity they have access to and their distance from the origin of the request. The score of these two aspects are normalized to match in scale, multiplied by a coefficient or its inverse to represent how important the aspects are, and summed together. The request is then sent to the data center with the highest overall score [34].

2.4 Related Works

There are several existing projects, packages, and tools used to simulate and reduce the carbon emissions via workload shifting. Two existing packages which are incorporated into some of the workload shifting algorithms that are implemented in the simulation framework, namely the Let's Wait Awhile algorithm, are LEAF and Vessim. Another tool that is used to limit the emissions produced by individual data centers is the Energy Efficiency and Low Carbon Enabler Green IT Framework [42]. In this Section, we explain what these tools are, and how our simulation framework diverges from them.

2.4.1 LEAF

LEAF, Large Energy-Aware Fog computing environments, is a tool which enables the analysis of power usage within the area of fog computing by simulating IT infrastructure [45]. While it was designed explicitly for Fog computing, it is flexible enough to be capable of representing cloud architectures as well, hence its inclusion in the cloud workload shifting algorithm Let's Wait Awhile [45, 46]. Within the algorithm, a LEAF infrastructure node is used to represent an individual data center on which jobs placed, and which itself has a power model node that determines the energy utilization of the infrastructure dependent on how many jobs are running on the infrastructure node at a given time [46]. Due to the heavy focus on infrastructure, LEAF is inefficient if used to determine the optimal shifting of a given workload, it is better when used as a carbon emissions model, as it is in Let's Wait Awhile [46]. Our framework does not attempt to usurp this functionality with its own version, but rather replicates however a given algorithm determines its electricity consumption.

2.4.2 Vessim

Vessim is a simulation environment developed to show the interactions between computer and energy systems [47]. Vessim mirrors the simulation framework discussed in this paper in some ways, notably as an environment in which carbon-aware systems can be tested and compared [47]. A clear commonality, beyond the purpose of comparing, is the ability to directly feed carbon intensity data from WattTime and Electricity Maps into Vessim as well as our simulation framework [47]. Though both simulators could enable the comparison of carbon-aware algorithms, Vessim has a much larger scope which comes with increased complexity, and does not provide the same support to compare or mix-and-match workloads between algorithms.

2.4.3 Energy Efficiency and Low Carbon Enabler Green IT Framework

This framework is a more theoretical tool to implement green technologies in a data center to minimize the environmental costs of the individual data center [42]. The tool lacks a concrete implementation, as it is more akin to a procedural series of steps to make an established data center more green. It consists of five phases; planning, identification & categorization, recycling & low carbon enabler policy, implementation, and analysis which are undertaken by the managers of data centers [42]. While the framework diverges significantly from the more specific area of carbon-aware workload shifting algorithms that our simulation framework focuses on, it is still in a related area that is outside the limitations we had established in Section 1.4.

Chapter 3

Simulation Framework

This chapter describes how our simulation framework was implemented. In Section 3.1 we discuss the methodology used in the framework. The last three Sections 3.2, 3.3, and 3.4 each detail a configuration category.
The objective of our design is to develop a simulation framework implementation that supports the development of carbon-aware workload shifting algorithms by having modular configuration categories, where a component from one category can be used in an experiment with almost any component from the other categories. Specifically, a given algorithm may be run with different workloads and carbon intensity datasets to see how they affect performance, or a given workload can be run on multiple algorithms to see which combination produces the fewest greenhouse gases. The simulation framework we developed is split into three primary configuration categories: Carbon Intensity Data, Workloads, and Workload Shifting Algorithms.

- Carbon Intensity Data: The ability to utilize different carbon intensity datasets
- Workloads: The ability to simulate different sets of jobs
- Workload Shifting Algorithms: The ability to accommodate alternative algorithms

These categories consist of components which users of the framework can add to, alter, or exchange. Components are designed to be interchangeable with other components within the same category, in order to be run with a component from each of the other categories, as long as it is feasible. Checks are put in-place to verify compatibility between components in different categories, but it is left up to the user to modify the output of components used together when they fail compatibility verification.
The interchangeable components enable and simplify comparisons between

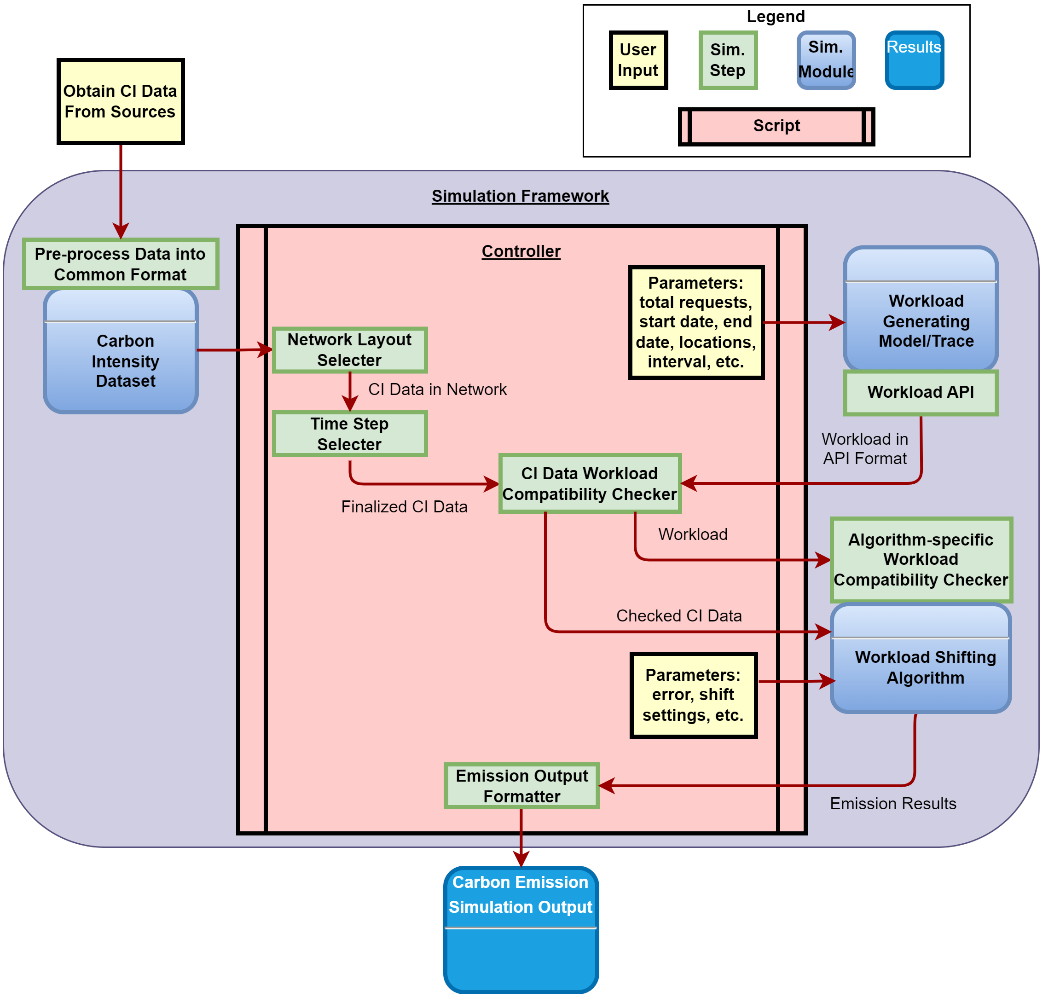


Figure 3.1: Simulation Framework Architecture

workload shifting algorithms, eases utilizing new workloads and illustrating how well a given algorithm handles them, in addition to showing the effects of alternative carbon metrics. While it may not work well for every algorithm, carbon intensity dataset, and workload that could be implemented within the simulation framework, certain abstractions need to be made to simplify the conjoining of different versions of the three components together. The carbon intensity data must follow a specified format, where each carbon intensity data point has a corresponding time and location. The workload module abstraction is also a common format, an extension of the Pandas data frame class which mandates columns for the time, duration, location, and number of duplicates for every job in the workload. The algorithm component only needs to be able to receive and use the carbon intensity data and workload in their formats and validate that the workload is compatible with the given algorithm. The algorithm can have virtually any output, but is presumed to return the carbon emissions produced at each time step it runs.

3.1 Methodology

The intended functionality of the simulation framework is a discrete event simulator where the 'events' or jobs within the workload are broken up between the discrete time steps provided by the carbon intensity data. This list of jobs at specific times with a corresponding carbon intensity at that time step is utilized by the workload shifting algorithm to determine the carbon emissions at each time step. In the case of temporal workload shifting algorithms, they may shift the jobs to another time step before using a corresponding carbon intensity to determine the emissions. This is a fairly simple outline that was common in several workload shifting algorithms, ensuring the simulator is highly compatible with existing algorithms in the space [24, 34, 48]. While discrete event simulation is the intended usage, the simulator is designed to be flexible to handle. Hard restrictions in the workload and algorithm modules are limited to enable the feasibility of algorithms of a different nature to operate within the simulator. The primary hard restriction exists in the carbon intensity module, as continuous carbon intensity data is not currently available from what we have observed, so potential algorithms are expected to rely on discrete carbon intensity data. An implemented example of this flexibility is the CarbonMin algorithm [24] which utilizes discrete time primarily, to determine the energy usage at a given time.

3.2 Carbon Metric Data

In order for the simulation framework to work with carbon metric data from a multitude of sources, a single format had to be chosen which any additional data added at a later point will need to be converted into. The format chosen was adopted from the Decarbonization-Potential paper and Github repository [41] on the potential and limitations of workload shifting algorithms for reducing carbon dioxide emissions. The format from the aforementioned paper has the advantages of being fairly simple, easy to compare the carbon intensity of multiple regions for the same time period, and bares close resemblance to the format used by Electricity Maps [11] which is a common data source for many algorithms in the space [24, 35, 41, 48]. To be used by the simulation framework, the carbon metric data has to be pre-processed into a single file per metric, starting with a sequential datetime column in the format: YYYY-MM-DD HH:MM:SS. The time step interval between each row has to be consistent per file, but can be as short as one minute, to as long as an hour. The columns that follow the datetime column are the given carbon metric at the corresponding datetime of the row for each given region. Code adapted and modified from the Decarbonization-Potential project [41] is provided in the process_data sub-directory to show how to pre-process carbon intensity data from Electricity Maps and WattTime.

Datetime	Region 1 Carbon Intensity	Region 2 Carbon Intensity	...

3.2.1 Cloud Provider Network Selection

Specific Cloud Provider networks can be chosen by the user by only using the carbon intensity of regions known to have a data center in the given network. The Network Selector feature functions by receiving carbon intensity data from several regions and the preferred Cloud Provider(s), then returning the carbon intensity data from regions that correlate to the chosen providers. We found that there are currently not any consistent, up-to-date sources for the locations of the data centers of multiple Cloud Providers, so we assembled a file with the information from several sources [1–6]. A csv file is used to determine which carbon intensity zones have a data centers in networks in a manner quite similar to what was done in the Decarbonization-Potential project [41]. The csv contains columns for the WattTime zone, the ElectricityMaps zone, the city/region, the country, several columns for major Cloud Providers which will have a one to indicate the presence of a data center for the Cloud Provider, and a final column for the continent. Users are expected to keep the csv file updated, if they intend to use the Network Selector feature.

WT Zone	EM Zone	Region	Country	AWS	Azure	GCP	Sum	Continent

3.2.2 Time Step Selection

Potentially the time step between each row in a given carbon intensity file are more precise than a given algorithm accepts, or than a user desires. In cases such as this, the Time Step Selector feature can be used to choose a new, larger time step which is a multiple of the existing time step. Currently, support exists for time intervals between one minute to one hour, though future expansions to the feature would expand this range.

3.3 Workloads

Workloads also need to have a standardized format so that they may be used by algorithms. Several different formats were tested, but the most efficient format found was to use an extension of the Pandas data frame class which ensures at least four common columns exist. Namely these common columns are: start_timestamp, duration, location, num_of_requests. They are defined respectively as: the starting timestamp of a given job, how many seconds are necessary to compute the job, the location of where the job originated, and how many jobs within the workload exactly match all other attributes. The last common attribute exists for the optimization of very large workloads, so it is expected, but not enforced that any compatible workload generator increments the num_of_requests attribute, rather than adding numerous duplicate rows in the workload extended data frame that is output. This is to support potential exceedingly large workloads which take too many resources otherwise. These required attributes also must be of type; start_timestamp:datetime64[ns], duration:timedelta64[ns], num_of_requests:int64 or float64. Additional columns can be included in the workload, but these would not be regulated nor would any potential workload shifting algorithm be expected to be able to handle these columns. During validation on whether the data provided meets the requirements of the extended data frame class, the rows are sorted from smallest start_timestamp to largest. Two methods are also included in the class extension to convert the workload object into a normal Pandas data frame or into a list of the standard columns in a consistent order.
Prior to the finalized decision of using an extension of the Pandas data frame class, other formats were implemented and tested for use as a standard workload format, such as API objects. The main issue found with utilizing an API object to represent jobs is the increased memory usage and decreased performance associated with manipulating a list of objects, rather than a data frame. When an API object implementation was used for large workloads such as the public 2019 Azure Functions Trace [17], the memory usage grew explosively to the point that the process was killed off prior to completion due to taking too much RAM.
The following subsections detail workloads which are currently implemented in the simulation framework.

3.3.1 Let's Wait Awhile Workloads

The Let's Wait Awhile paper [46] detailed two temporal workloads, one to represent periodic nightly batch jobs with a consistent duration and scheduled arrival time, the other for ad-hoc machine learning jobs with a variable duration and time. Both of these workload generation models have been recreated from code provided for the original algorithm and expanded upon in the simulator. Additionally, both of these workloads are designed for temporal algorithms, so each job will be from a single location if provided or N/A otherwise.

The periodic workload creates a job with a given duration at a given time each day for a given number of days. The duration, start time, and number of days and therefore jobs can all be provided to the model to create a suitable workload. The ad-hoc workload is a bit more substantial, as it can choose a total number of jobs, a start date and end date for the workload, a location, whether to only create jobs on workdays, a start time and end time for each workday, and a duration range for the jobs. The number of jobs on each day, the start times, and the duration are all randomized.

3.3.2 CarbonMin Workload

The workload described in the CarbonMin paper [24] is a spatial workload created to replicate what a generative AI application such as ChatGPT may experience. As no code for the workload is provided, we developed a model for our best approximation of it. The model receives the total number of jobs each day should have and the total hours it should generate for, then distributes them between the hours of the day and several location origin countries based on a metric of the country such as population and the active population during the UTC localized hour of the country.

A script is provided to create a country distribution data frame which takes in a set of countries and some metric to weight them, such as the number of social media users which is the metric we used to create the default country distribution data frame. The data frame consists of four columns, the name of the country, the ISO2 code of the country, the UTC time offset of the capital of the country, and a percentage for the weighting of the country based on the used metric.

The workload generating model uses the country distribution data frame by cycling through the hours it should make jobs for and localizing the hour to each country using the UTC offset, then it uses the localized hour to find the percentage of the population active at the given hour from a table based on the population distribution findings of a paper [20]. The percentage of active population is multiplied to the weight of the country and the total daily jobs to find how many jobs said country creates at the given hour. This process is repeated for all the countries in the country distribution data frame and for all hours. All jobs are given a duration of about 2.07 seconds based on the generative AI request findings from the original CarbonMin paper [24].

3.4 Workload Shifting Algorithms

Carbon-aware workload shifting algorithms aim to direct the processing of jobs within the workload to a time or place in order to minimize the carbon emissions that the workload might otherwise produce when processed. Therefore, within the framework, they include two components, a way to shift the workload and a carbon emission model or way to determine the consequences of the shift by providing as output the emissions the workload shifting algorithm generates given the specific workload and carbon metric dataset. This means that any algorithm developed in the framework would receive a workload and carbon intensity data frame as input, then output the carbon emissions produced. Algorithms are free to use workload data frames in formats that can be utilized more efficiently as well as to not employ columns that are unnecessary for the algorithm. Algorithms can potentially also use additional columns, though they are encouraged not to, despite the lack of hard restrictions. The use non-standard columns decreases the interoperability with other workloads, as only workloads created explicitly for the given algorithm would be possible to use with it, negating many of the comparative benefits that the simulation framework provides unless an outside consensus is reached regarding a column. Algorithms in the simulation framework are expected to verify that incoming workloads meets their requirements and to reformat the workloads for their own use, but aspects where relevant information is lost must be left to the sole discretion of the user. These aspects include the dropping of unnecessary columns and the conversion of column data types to significantly less precise types, i.e. datetime to timedelta. The purpose of this burden placed upon users is to ensure the user is aware of what information is actually being compared when different workloads and algorithms are used together. Should the workload received by the algorithm fail its verification, the algorithm is expected to raise an attribute error specifying the issue in the workload.

Workload shifting algorithms output the carbon emissions per time step, given their in-built carbon emission model at a minimum, though other statistics can also be output. CarbonMin, for example, also writes the regions requests were shifted to with the number of requests sent to the region in the order of most utilized to a file. The output should be in a csv file or files in the directory 'simulation_framework/output/<name of workload shifting algorithm>/<name of workload>/<name of carbon metric>/'. The two algorithms chosen for implementation in the framework and described below, are CarbonMin and Let's Wait Awhile, as both are relatively simple and do not require supporting sets of data. The two algorithms also cover a distinct concept each, spatial shifting and temporal shifting, respectively.

3.4.1 Let's Wait Awhile

The Let's Wait Awhile workload shifting algorithm focuses on temporally shifting jobs within a single region in order to minimize carbon emissions. The algorithm was adapted for the framework from the paper and Github repository [46]. The algorithm has two modes, ad-hoc and periodic which each place slightly different requirements upon input workloads. The ad-hoc mode denotes the scheduling strategy as single directional temporal shifting, as jobs can only be postponed to be done in the future due to a lack of information on what jobs require until they are actually received, such as for the training of machine learning models. The periodic mode indicates that the workload is to undergo bidirectional temporal shifting, where the jobs are repeated and expected like batch jobs, so they can be scheduled to be done both before or after they are strictly received. For the simulation framework, this results in the single direction shifting mode requiring the duration column in the workload data frame that is input, while the bidirectional shifting mode does not, as it assumes the duration is both known and is equal to the carbon intensity measurement intervals. The periodic duration being equal to the measurement interval is part of the original scheduling strategy, rather than something of our own design.

Both modes require the start_timestamp and the num_of_requests columns, though the user must ensure certain criteria are met for the workload to be accepted. First the start_timestamp column must be in minute precision at minimum, to start counting up from the start of the first job, and in intervals that are multiples of the carbon intensity measurement interval. In practice, this means that start_timestamp is of type timedelta64[ns], has dt.components.seconds equal to zero, and the dt.components.minutes modulated by the measurement interval equal to zero. For the single direction shifting, the duration column has identical practical requirements to the start_timestamp column. No other columns than those listed above are allowed in the workload.

When the algorithm is provided a workload and it is verified as acceptable, the workload is converted into a different format more akin to what the algorithm was originally developed to expect. The start_timestamp is converted into the total minutes as an int while the num_of_requests of a given row is used to creates the same number of duplicates of the row rather than existing as a column itself. After conversion, the workload is a 2-D list where each job exists as a list of the start time and the duration, both in minutes. The bidirectional mode workload uses the measurement interval as the duration. Aside from the workload verification and conversion methods, care was taken to avoid making changes to the logic of the algorithm. A non-cosmetic change that had been made to the code of Let's Wait Awhile is the ability to use other measurement intervals other than the hard-coded thirty minute interval the algorithm originally was only able to use. This addition necessitated the replacement of hard-coded numbers with small calculations, such as for determining the number of intervals which occur in an hour or a day. Section 4.1.1 prove that these changes do not alter the results when running the algorithm in any easily discernible way.

3.4.2 CarbonMin

The CarbonMin algorithm spatially shifts jobs between the considered regions in order to minimize emissions. It is designed for generative AI jobs such as ChatGPT to reduce generative AI emissions both now and in the future [24]. Though CarbonMin was designed to compare its affects on current Generative AI requirements versus those in the future, only the modern day aspect was implemented, as the future side was deemed out of the scope of the framework for now 2.3.2.1. Acceptable workloads need to have two columns, a start_timestamp in minute intervals that match the carbon intensity data intervals, as well as the num_of_requests that occur during that interval. The standard duration and location columns are excluded as the algorithm is designed for and assumes the duration of any jobs it receives are equal to the average processing duration for a ChatGPT request, 2.07 GPU seconds, as found by the paper [24]. It does not take into account latency from moving from the location of the job's origin , as the type of job, generative AI requests, as the aforementioned high processing duration time far exceeds the potential latency measured in milliseconds. In order to match the original workload the algorithm was designed for, workloads in the common data frame format are converted into a two dimensional list by the algorithm, prior to use.

When CarbonMin is run, it receives the total number of requests it needs to shift for each time interval, according to the workload. It then shifts the requests to fill up the resource capacity of the regions with the lowest carbon intensity out of the set of potential data center regions. The resource capacity is determined from dividing the total number of requests in the given hour by the number of potential regions and then dividing by 0.3, which represents the assumed 30% exploitable capacity used in the CarbonMin paper [24]. Once the number of requests shifted to each data center is determined, both the operational and embodied emissions of the decision are calculated. The operational emissions of a data center with the given requests is calculated by multiplying the computation duration per request, the thermal design power, the power usage effectiveness, carbon intensity, and the number of requests together. Likewise, the embodied emissions are calculated by multiplying the number of requests, the computation duration per request, and the resource capacity of the data center together, then dividing by the hardware lifespan and multiplying by the emissions per GPU. The exact numbers for all of these variables barring the number of requests, the carbon intensity, and the resource capacity, are static, identical for all data centers, and taken directly from the paper [24].

Chapter 4

Evaluation

This chapter evaluates our simulation framework in terms of functional correctness of algorithms implemented in it and demonstrations of the developed features. The features available in the simulation framework are: choosing time step intervals, using different types of carbon intensity data, utilizing different experiment run durations, experimenting with different workloads, and accounting for the regional availability of cloud provider networks. The chapter concludes with a discussion of the overall benefits that the simulation framework provides.

4.1 Functional Correctness

This section details how closely the algorithms implemented in the simulator represent their original counter-parts by replicating experiments done in the original papers.

4.1.1 Let's Wait Awhile

The article describing Let's Wait Awhile had two experiments [46], one each to demonstrate the two scheduling strategies, periodic and ad-hoc. Recall from Section 2.3.2.2 that periodic is intended for scheduled batch jobs, where the job is known and can be scheduled prior to it being officially received, while ad-hoc is for a randomized workload that cannot be scheduled ahead of when it is received. The emissions model, LEAF [45], was used to represent a single data center on which the temporally-shifted workload is run. A common detail is that all are run starting on January 1st, 2020, but with an end date dependent on the specific experiment. The experiments also utilize the average carbon intensity data from California, France, Germany, and the

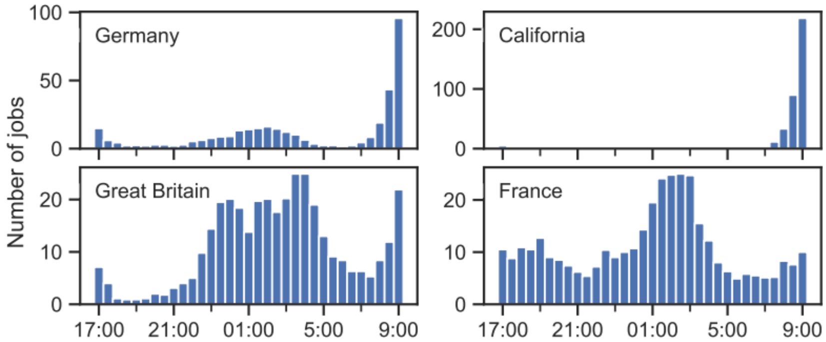
United Kingdom that is provided in the repository of Let's Wait Awhile, measured in 30 minute intervals [46].

Periodic Experiment

Setup The first replicated experiment is Scenario 1: Nightly Jobs, for a periodic workload and shifting strategy [46]. The workload consists of jobs which start at 1 a.m. and each occupy 30 minutes of computation time on the data center, one job each day between 1/1/2020 - 31/12/2020 for a total of 366 jobs. Additionally, a forecast error rate of 0 or 0.05 times the mean carbon intensity is applied to each interval of the carbon intensity data. The mean of ten runs of each experiment is used to generate the graphs.

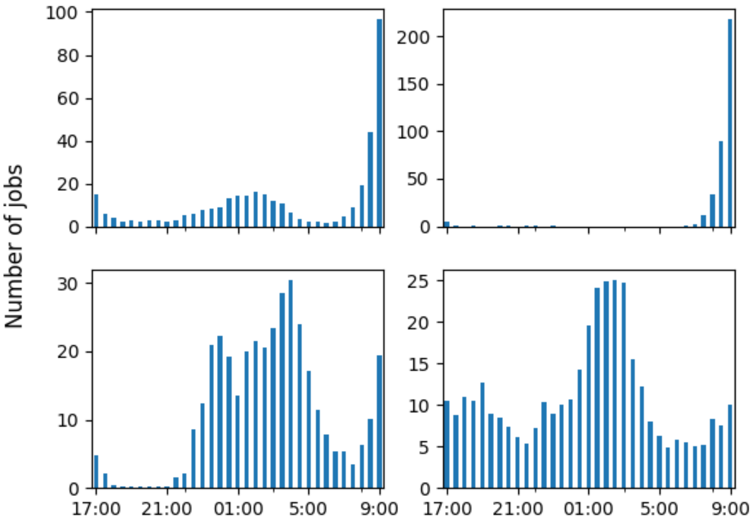
Results As can be seen from Figures 4.1.1 and 4.1.1, the workload is shifted by the algorithm implemented in our simulator in an almost identical manner to what is shown in the plots in the original Let's Wait Awhile paper [46]. The trends show that both the Californian and German experiments shift almost their entire workloads, over 95% for California, to between 7 and 9 a.m. while France and Great Britain are more spread out with a large portion of the workload shifted to around noon. The plots in Figures 4.1.1 and 4.1.1 are quite close, but slight deviations can be noted. The difference between the results shown in the plots from the paper and the plots generated in the simulation framework is most easily observed by comparing the Great Britain plots. The peak around 4:00 in the simulator graph is a bit higher than in its counterpart with the later of the two peak bars being clearly higher than the other, compared to the two having an identical height in the original plot. As the code is available in a repository [46], we are able to run the experiment without making any changes to the code for the simulator, the results of which are shown in Figure 4.1.1. As can be seen, it is indistinguishable from the plots obtained in our simulation framework and in fact slightly differs from the graphs shown in the paper. We believe this difference is due to updates made to the related code in the Let's Wait Awhile repository after the paper was published, including the integration of Vessim in the scheduling strategy nine months ago, as of writing [12, 46, 47]. Additionally, we calculated the mean squared error of active jobs at each hour due to the temporal shifting, by comparing the simulator periodic experiments on each country with the error rate of 0 and 0.05 with the same periodic experiments run outside the simulator with no changes to the code. Every experiment comparison resulted in a mean squared error of 0, except for California with an error rate of 0.05. This sole exception was still only $4.276554177951085e-28$, which is such a small number that it is likely caused from Python floating

Original Let's Wait Awhile periodic experiment workload shifting over a sixteen hour scheduling window and a 5% forecast



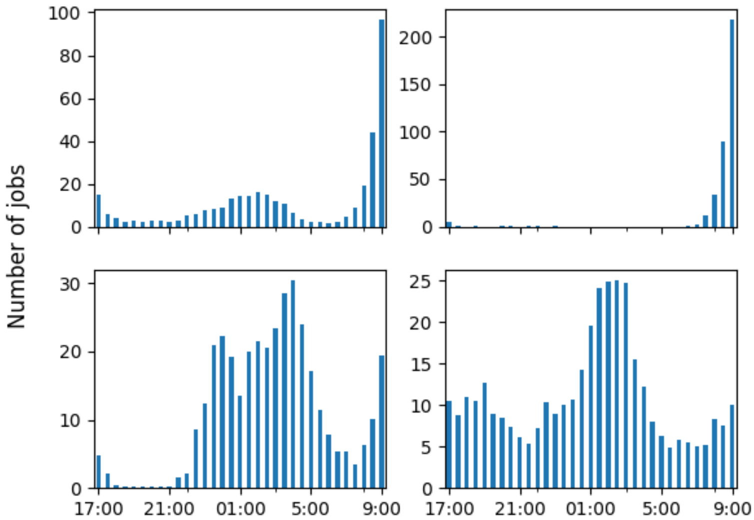
error [46].

Periodic Let's Wait Awhile experiment within our simulation



framework.

Recreation of the results from the Let's Wait Awhile periodic experiment with no changes to code from the latest commit in the git repository of the paper (Nov 13,



2023).

point arithmetic issues rather than an actual difference in emissions.

Ad-hoc Experiment

Setup The second replicated experiment is Scenario 2: Machine Learning Project, for an ad-hoc workload and shifting strategy [46]. The workload has a total of 3387 jobs divided randomly between the 30 minute intervals within the work hours of, 9:00 - 17:00 on Mondays through Fridays. Additionally, the duration of each job is anywhere between four and ninety hours as long as it is divisible by 30 minutes. The experiment is run for each combination of one variable from the following bullet points:

- Country: California, France, Germany, United Kingdom
- Forecast Error Rate: 0, 0.05, 0.1
- Interruptible Workload: True, False
- Forecast Method: Next Workday, Semi-Weekly

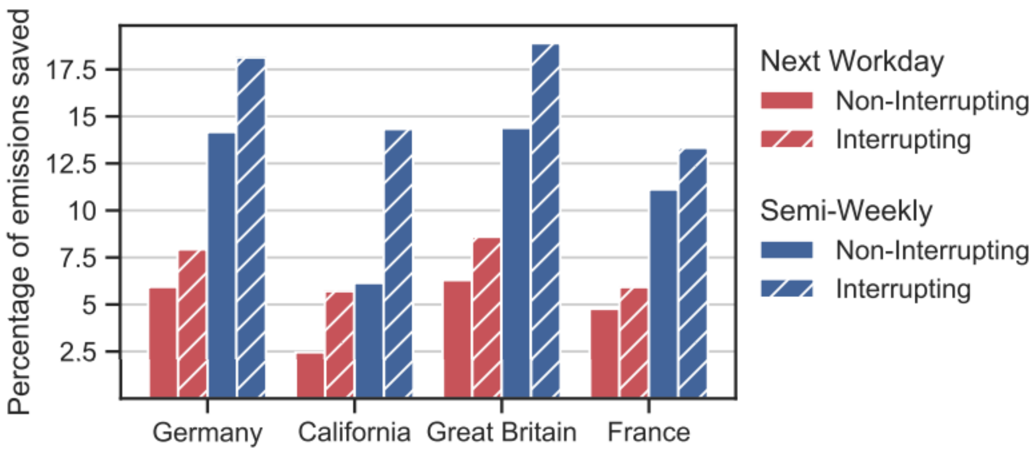


Figure 4.1: Original Let's Wait Awhile ad-hoc experiment demonstrating the emission savings with different countries and constraints [46].

Results Figures 4.1 and 4.2 appear to be virtually identical, which show that the ad-hoc scheduling strategy implemented in the simulator is quite accurate to the original algorithm. The change since the writing of the original paper, to using Vessim in the strategies is also true for the ad-hoc strategy, just as it was for the periodic scheduling strategy discussed above [46].

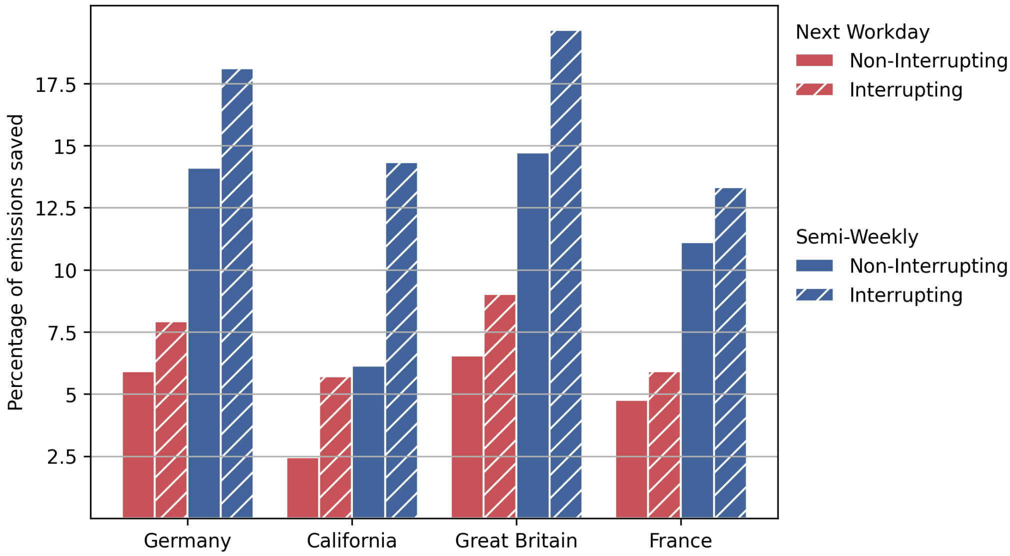


Figure 4.2: Recreation of the Let's Wait Awhile ad-hoc experiment within the simulator.

Presumably, the reason a slight difference can be noted in the periodic experiments but not in the ad-hoc graphs, is due to figures 4.1.1, 4.1.1, and 4.1.1 showing the exact number of jobs that occur in each interval over 24 hours with a total of 366 jobs, while figures 4.1 and 4.2 show the percentage difference is emissions with a much larger set of jobs (3387) so minor deviations are obscured far more easily. We determined the mean squared error of active jobs per hour for every variation of the experiment between the ad-hoc implementation in the simulator and the original from the repository. The error rate found was always quite small, usually 0, with the largest being $3.4066974976243976e-26$ which is small enough to be a Python float arithmetic error.

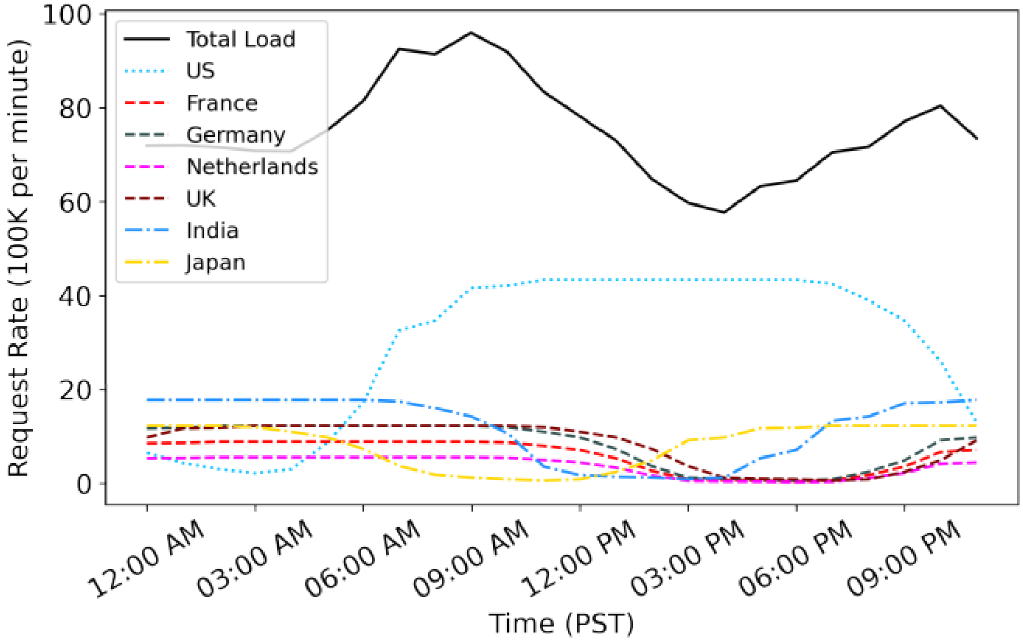
4.1.2 CarbonMin

CarbonMin had one experiment run on it to determine functional correctness, as only the shifting experiment titled Today was implemented from the original paper [24], rather than the shifting of 2035. Unlike Let's Wait Awhile, no publicly available repository is provided for CarbonMin, so far less information on the details regarding both the algorithm and the experiment is available. This combined with details of the workload being locked behind a paywall [9], result in an experiment which is our best approximation of the original. The carbon emissions of the data centers with a given workload distribution is determined in the same way that it was detailed in the paper, so both the operational emissions from working the workload and the embodied emissions from the wear-and-tear on the hardware of the data center are taken into consideration, the details are provided in Section 3.4.2.

Setup This experiment emulates the Results:Today experiment from the paper introducing CarbonMin [24]. As the original paper used data from ElectricityMaps, we use the same source for this experiment. The average carbon intensity data of a year from the ElectricityMaps zones IN-WE, NL, US-TEX-ERCO, FR, US-CAL-CISO, JP-TK, DE, and GB are used in hourly intervals to cover the respective test set of regions; India, the Netherlands, Texas, France, California, Japan, Germany, and the United Kingdom [14]. 80825 requests per day, from the CarbonMin paper [24], are then divided between these regions according to their respective localized active hours, population sizes, and ChatGPT usages. The workload could not be matched exactly to the original, as the source for determining the distribution of request origin location, similarweb [9], charges for historical data. Therefore, the distribution had to be estimated from what was stated in the paper, "the load is dominated by USA (39%) and European Countries (35%), reflecting their higher ChatGPT usage", and what can be gleamed from the present load Figure 4.1.2 (obtained from the paper). Additionally, a slightly different active hour distribution graph [20] was utilized in the workload generation model, than was used in the CarbonMin paper. This all results in the workload distribution of Figure 4.1.2 which is used in our experiment. Exact details used in the experiment can be obtained from the project repository. While most countries appear to somewhat match in both distributions, the two largest, the U.S. and India diverge, as India represents a larger proportion of the total workload and the U.S. awakes earlier as the UTC offset of the capitol of the countries are used. This means the total workload has a larger peak in the PST morning with a smaller peak in the night. While maintaining consistent rules for each country, this is the closest match for the workload distribution we were able to create.

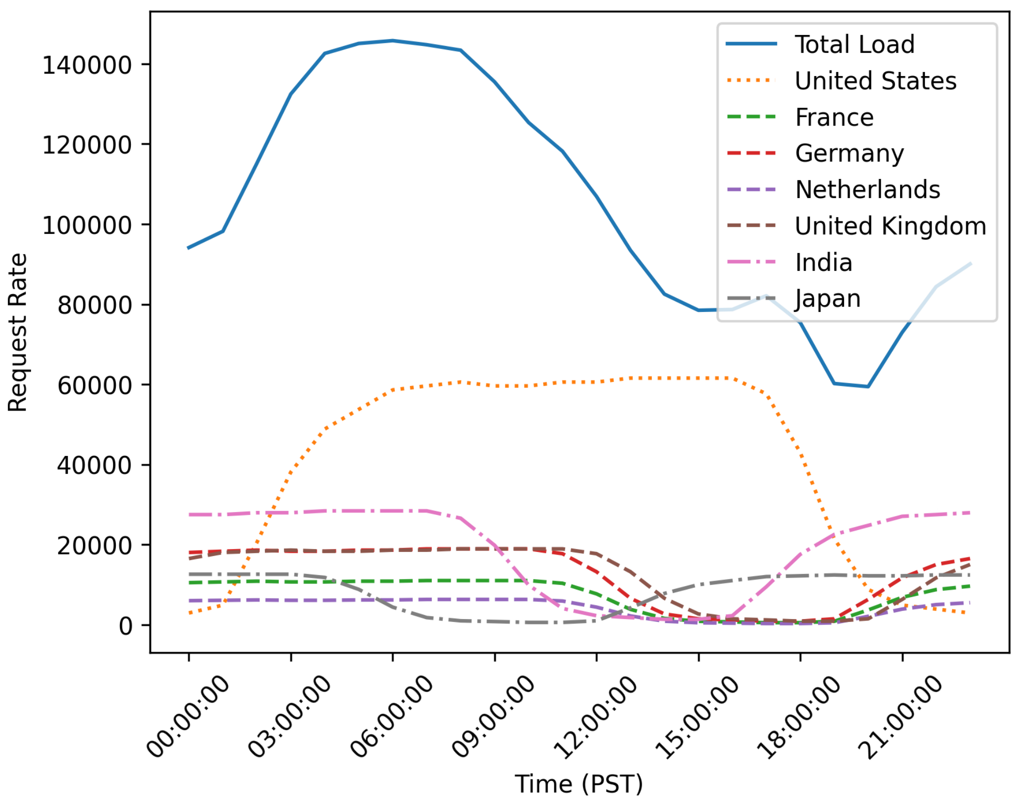
Results Figures 4.3 and 4.1.2 show the emissions produced by CarbonMin and Local in the paper and from the implementation in the simulation framework. Recall that Local refers to the shifting strategy where jobs are not shifted, as they are worked on in the same region they originate from. They appear to be different due to differences in source data, but the actual average reduction in emissions between the Local and CarbonMin shifting

The original present day request distribution plot over time, taking into account the ChatGPT usage, population, and active hours of each country [24]. It is licensed under a Creative Commons Attribution International 4.0



License.

Recreation of the request distribution plot over time, taking into account the population and active hours of each country, with ChatGPT usage estimated from Fig-



ure 4.1.2.

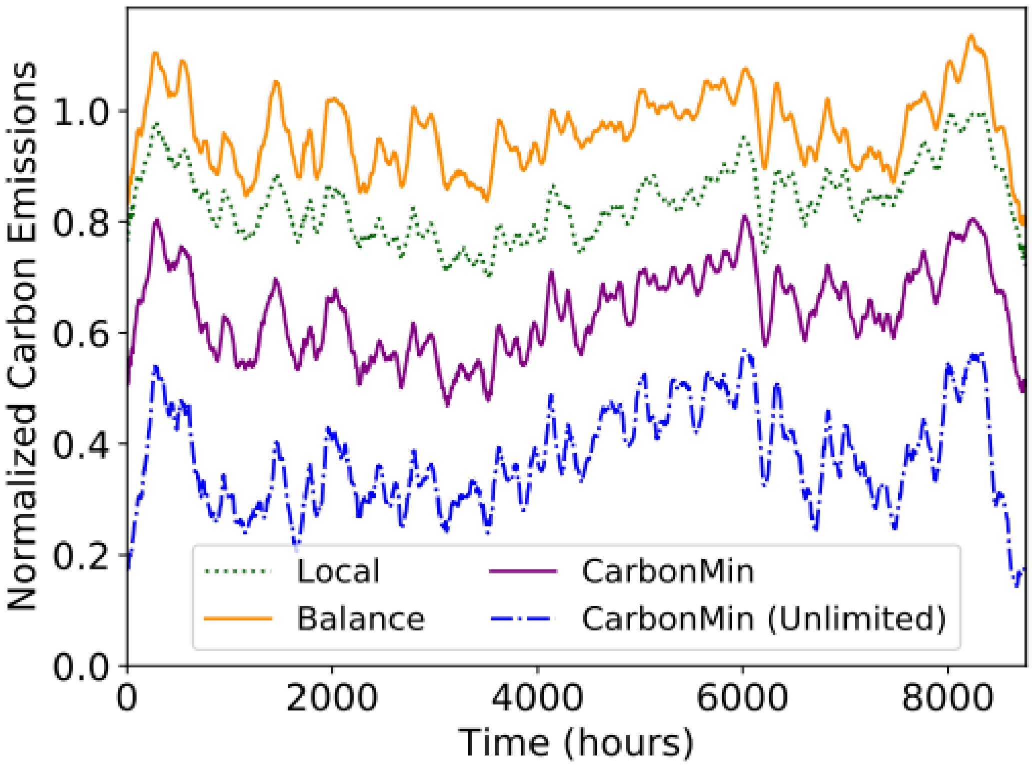
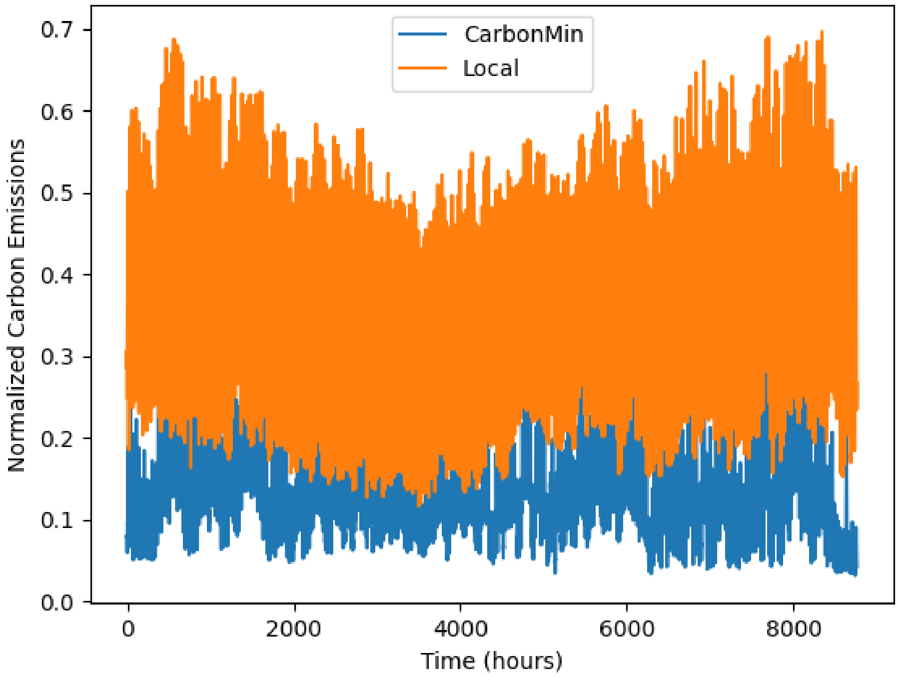


Figure 4.3: Original CarbonMin experiment demonstrating the emissions saved due to different spatial shifting strategies. Balance means the requests are shifted between data centers so that they are all equally utilized, ignoring the potential emissions completely. Local essentially means no shifting occurs, as all requests are handled in the same region they originate from. CarbonMin is the carbon-aware spatial shifting algorithm and CarbonMin (Unlimited) is identical aside it not taking the capacity of the data centers into account [24]. Local was the baseline so only it and CarbonMin were replicated for the experiment in the simulator. Licensed under a Creative Commons Attribution International 4.0 License.

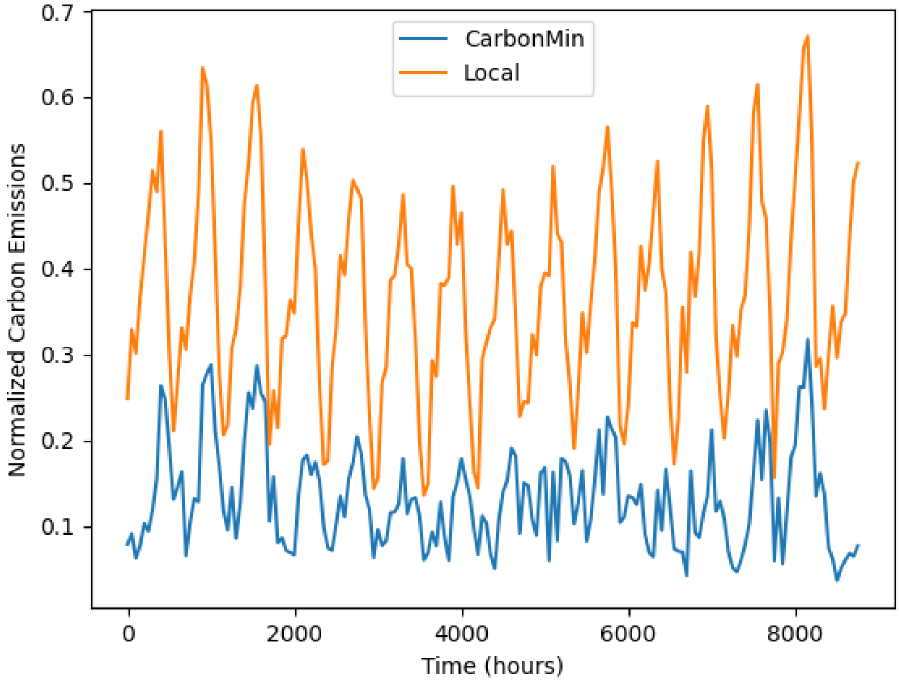
strategies are quite similar to that reported in the paper. In our implementation, Carbon produces 37% fewer carbon emissions compared to the Local strategy baseline, which is similar to the 35% reduction reported in the paper [24]. From evaluating the emission results, the CarbonMin implemented in the simulator always produces fewer emissions than the Local strategy at every hour in the experiment. This can be witnessed better from Figure 4.1.2 which only plots one from every fifty data points, but is otherwise identical

Recreation of the CarbonMin experiment within the simulator with all data points



plotted.

Results from Figure 4.1.2, smoothed by only plotting every 50^{th} data



point.

to Figure 4.1.2. It shows that the Local shifting strategy consistently produces higher emissions than CarbonMin. This supports the conclusion that the CarbonMin implementation is quite faithful to the original.

4.2 Demonstrations of Simulator Features

This section demonstrates the benefits of the simulation framework by running similar experiments after altering one or more aspects of the experiment. These changes include the use of a different time interval step, different type of carbon intensity, a different experiment duration, a different workload, or a different set of regions.

4.2.1 Different Time Step Intervals

Setup The first demonstration of the simulator is the ability to choose different time steps, provided that the data and algorithm support the new time step by having a shorter interval than is preferred. We show this feature with the periodic temporal shifting strategy. In the functional correctness testing in Section 4.1.1, the algorithm was tested with a time step interval of thirty minutes, we compare those baseline results to another run with a sixty minute interval, but otherwise identical setup. The setup consists of the average carbon intensity data for the regions of California, France, Germany, and Great Britain and a workload consisting of one job at 1:00 a.m. per day. The data comes from the Let's Wait Awhile repository [46]. We do not use a forecasting error in this demonstration.

Results Figure 4.4 demonstrates how the simple change of time step can affect the emissions, even with otherwise identical carbon intensity data. We believe one major contributor to the increased emissions is the longer sixty minute intervals mean there are fewer opportunities to shift workloads. This gives the algorithm fewer options to shift to and so, less flexibility for the algorithm when attempting to minimize the emissions. This is evidenced by the relative lack of variation in the experiments for France and the United Kingdom, who do not have a highly variable carbon intensity over time, and so prefer to spread their workload more widely over the time span. California and to an extant, Germany do have significant swings in their carbon intensity data over time, so they do not spread out their workloads as much, and are harmed by the reduction in shifting options. Figures 4.1.1 and 4.1.1 corroborate the variability of the workload distribution of each country, despite

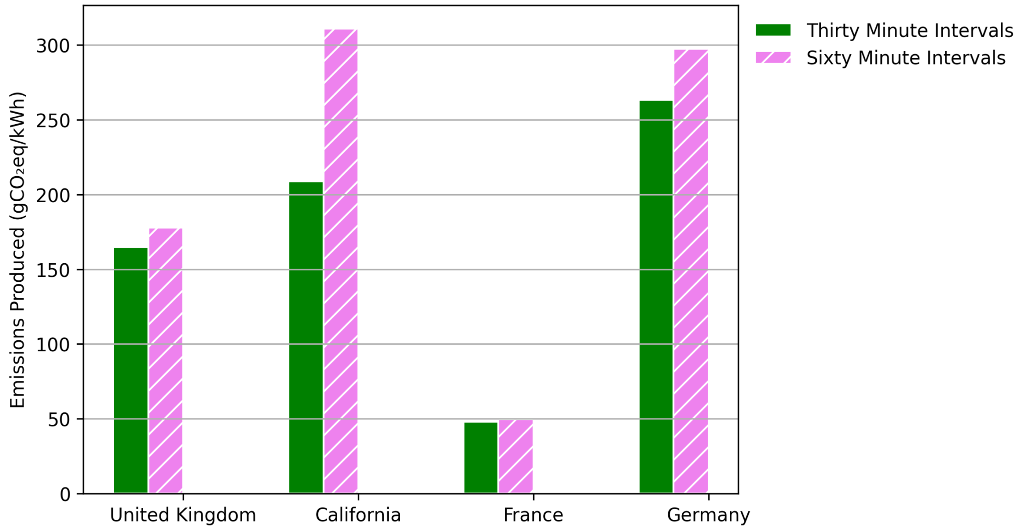


Figure 4.4: The periodic experiment with time steps of half an hour or an hour.

showing the shorter thirty minute intervals. They show that the Californian and German experiments schedule most of their jobs around 9:00, while the other two experiments have jobs more evenly distributed throughout the day.

4.2.2 Different Carbon Intensity Metrics

Setup Another feature of the simulator is the possibility to use different carbon intensity data. For the Let's Wait Awhile functional correctness experiments in the prior section 4.1.1, the same average carbon intensity data that was provided with the original was used. Now we show the carbon emissions produced when the periodic algorithm with a workload identical to the validity experiment is run with average carbon intensity or marginal carbon intensity, both sourced from Electricity Maps [11] for the year of 2023. The Electricity Maps data is used in place of average carbon intensity data provided in the Let's Wait Awhile repository [46], to make scheduling decisions and to determine the amount of emissions generated by the scheduled workload. For comparison purposes, we also show the emissions with the original average carbon intensity data from the Let's Wait Awhile repository [46] for the latest year available, 2020. As the smallest time step interval which Electricity Maps provides is hourly, that is what we will use. The workload is still one job per day, with a start time of 01:00. The experiment was run for the same regions as the original periodic experiment, namely, California, France, Germany, and the United Kingdom.

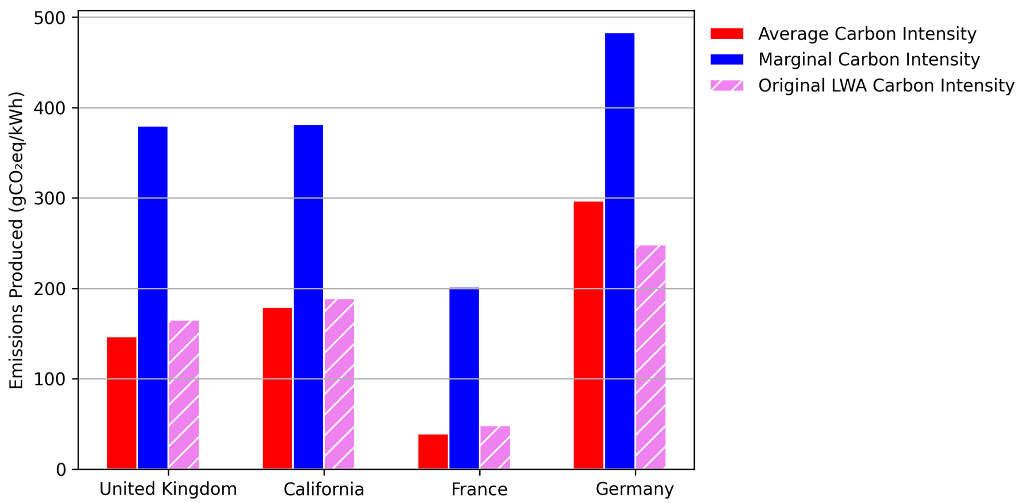


Figure 4.5: The impact of Electricity Maps' average vs marginal carbon intensity on the periodic strategy in each country. The emissions that result with the Let's Wait Awhile average carbon intensity data are included to compare.

Results Figure 4.5 shows the notable discrepancy between the emissions with either average or marginal carbon intensity, as in there are higher emissions produced with marginal carbon intensity data. This is to be expected as Section 2.1.2 stated that the marginal primarily consists of more carbon intensive energy sources due to their generation being relatively easy to scale up and down. The difference seen between average and marginal is less pronounced in regions with high average emissions, as fossil fuels provide much of the power for both the base load and the marginal requirements. Meanwhile in regions like France, where very little of the base load comes from carbon intensive generators [13], the marginal emissions dwarf the average several times over. The two average carbon intensities, one from Electricity Maps, the other from the Let's Wait Awhile repository, have much closer results to each other than to run with the marginal carbon intensity data. They vary due to different years used for each and different methodologies used in collating the carbon intensity data.

4.2.3 Different Experiment Durations

Setup This experiment uses the Let's Wait Awhile Ad-hoc algorithm to test running an algorithm for different durations. The workload is consistent with what the ad-hoc functional correctness experiment used, only varying in duration. The workload either has jobs that start between 2020-01-01 00:00:00 to 2020-6-30 23:30:00 or 2020-01-01 00:00:00 to 2020-11-30 23:30:00. The algorithm is simulated for thirty days longer than the end of each workload in order to provide time to shift jobs at the end of the workload. During this extra time no new jobs are created, so it is akin to the few weeks the original experiment would extend past its official end date. The commonalities of the workload are that there are 3387 total jobs, jobs only start between 09:00 and 17:00 on workdays, and they have a duration between four and ninety hours, as in the original experiment [46]. The carbon intensity data is the same which is provided in the Let's Wait Awhile repository and no error or forecast method were used.

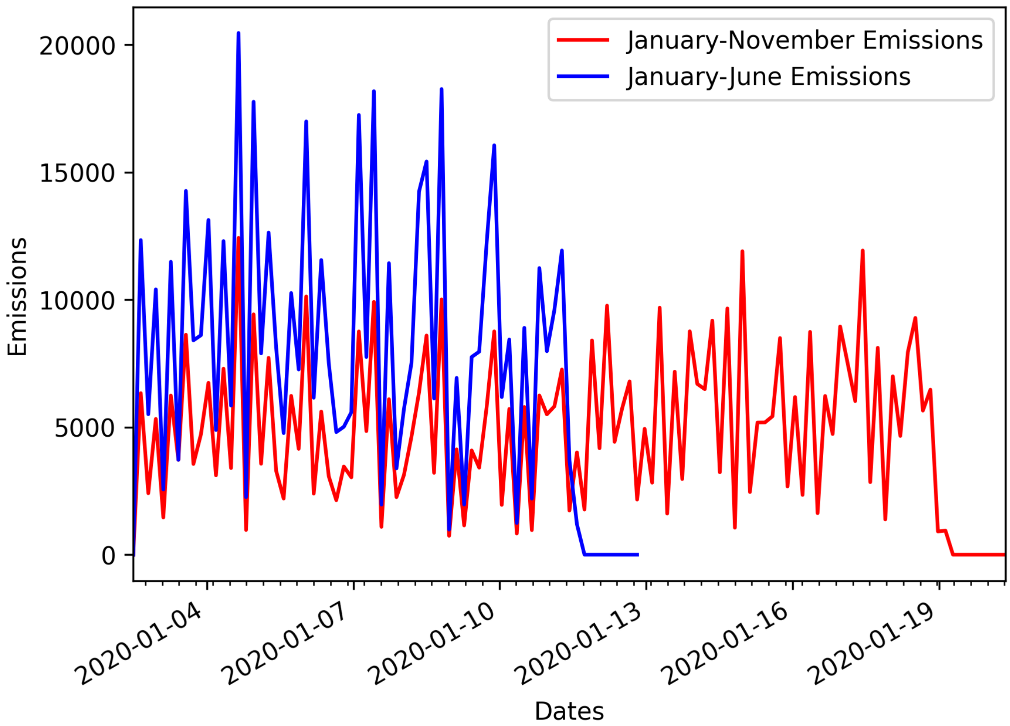


Figure 4.6: Every 150^{th} hour of the Ad-hoc experiment with a workload and simulated run-time of California for either a year or half a year.

Results Figure 4.6 shows how the ad-hoc strategy shifts jobs in California with workloads of different lengths. As both workloads have the same number of jobs, the half year workload had roughly twice as many active jobs at a given interval during its runtime to cover the same number of jobs. One would think this means that the half year run would have higher emissions than the full year run, as there are the same number of jobs but less time available, so less time slot options to try to shift to minimize the emissions, but this is would be incorrect. The emissions generated were less in the half year than the full year. For the United Kingdom, the emissions were $6.3990358e7$ gCO_2eq in the half year and $6.6655812e7$ gCO_2eq in the full year, while for California $7.901625e7$ and $8.5956501e7$ in the half year and full year, respectively. We believe the increased emissions are a result of lower average carbon intensity in the first half of the year than the latter, in all regions in the experiment. Table 4.1 provides evidence for this hypothesis, as the second half of the year in each region tested had a higher average carbon intensity than the first half. It can also be seen that there exists a greater difference between the emissions in workload runs for California and Germany, regions with a comparatively large increase in the mean of the average carbon intensity between the first half of the year and the second, than France and Great Britain, which have a lower difference.

Table 4.1: The mean of the average carbon intensity in each half of a year.

Countries	**January-June CI**	**July-December CI**
California	253.6	305.5
United Kingdom	201.2	224.5
France	47.9	64.6
Germany	267.0	359.2

4.2.4 CarbonMin with a Different Workload

Setup The simulator enables the evaluation of an algorithm with other types of workloads, provided there is some compatibility between the algorithm and the workload. To demonstrate this, we chose to run the CarbonMin algorithm with an ad-hoc temporal workload from the workload generating model of Let's Wait Awhile. The reason we decided on this combination rather than the inverse, is due to the Let's Wait Awhile algorithm requiring very long durations (in the range of minutes to hours) for the jobs in workloads given to it, whereas the generative AI workload model from CarbonMin produces jobs with a short duration of about 2.07 seconds. CarbonMin on the other hand, disregards the duration of the jobs. The carbon intensity data used is the marginal data from Electricity Maps from the start of June 2022 to the end of the month with an hourly time step interval. The generative AI workload used in prior CarbonMin experiments consisted of $80,825$ jobs per day or for the 30 days of June, a total of approximately $2,424,750$ jobs. The latter number is what was provided to the ad-hoc temporal workload model for the number of jobs it should generate. Even though the duration is ignored by CarbonMin, we also had the model give every job a duration of an hour, to increase the speed of the generation process. The static duration streamlines the workload generation process by shrinking the size of the common workload format data frame the ad-hoc temporal workload model creates by condensing all jobs starting at a given time to one row, rather than rows for each duration at each start time.

Results In Figure 4.7, the experiment with the temporal workload is shown to have sharp drops to zero emissions. This is due to the workload adhering to workdays and work hours, so no jobs are sent outside of the hours between 9:00 a.m. to 5:00 p.m., Monday through Friday. As there are a scant few hours in the month in which the $2,424,750$ jobs can be sent, a very large number of requests are sent whenever it is possible. The generative AI workload by comparison, sends requests every hour in a twenty-hour hour pattern. Despite the seemingly large discrepancy between the two workloads, the total emissions output are quite similar as the CarbonMin with the temporal workload created $448,871,576$ gCO_2 equivalent while the generative AI workload generated $431,714,433$ due to the same carbon emission model from CarbonMin is used in both to determine the emissions produced. While the total jobs are about the same, barring rounding errors in the workload generating model calculations, the difference in emissions comes from higher carbon intensities during the work hours when the temporal workload requests are sent, as work hours tend to have higher carbon intensities [26].

4.2.5 Different Cloud Provider Network

Setup This final experiment demonstrates the ability to choose between different cloud networks, provided information on the infrastructure of the network has been supplied to the simulator. We use the CarbonMin algorithm, its generative AI workload model with its default regional request origin distribution, and the Watt Time marginal carbon intensity data for the month of January 2022. As the workload model follows a twenty-four hour pattern in request distribution, the same number of requests, $80,825$, are sent each day for a total of $2,505,575$ requests. The carbon intensity

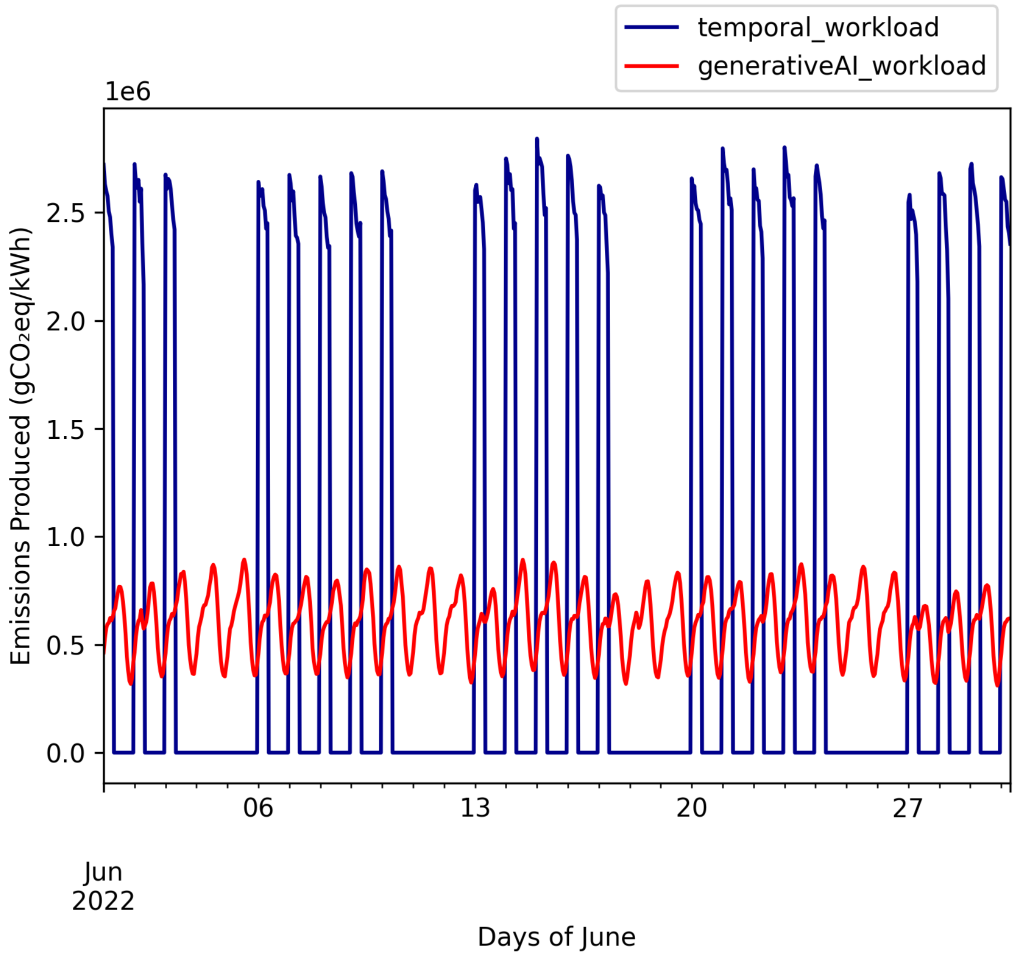


Figure 4.7: The emissions produced when the CarbonMin algorithm is run with a workload from its generative AI workload model vs the temporal workload model of Let's Wait Awhile.

data used an hourly time step. Regions in the baseline carbon intensity set, which includes all regions used in the experiment, are added to network sets according to the network_zone dataset we have compiled. Each region in a network set is treated as a data center that requests can be spatially shifted to. The networks we have considered are Amazon Web Services, Microsoft Azure, Google Cloud Platform. It should be noted that we obtained the carbon intensity of regions from Watt Time, based on how many networks include the given region, so there is a large degree of overlap in the network sets. The entire list of ISO2/Electricity Maps region codes chosen for

Table 4.2: Regions in Each Network

Networks	Regions
AWS	AU-NSW, AU-VIC, BR-CS, DE, ES, FR, GB, IE, IN-WE, IT-NO, JP-KN, JP-TK, KR, SE, SG, US-CAL-CISO, US-MIDA-PJM, US-NW-BPAT
GCP	AU-NSW, AU-VIC, BR-CS, CA-ON, DE, ES, FR, GB, IE, IN-WE, IT-NO, JP-KN, JP-TK, KR, NL, PL, SG, US-CAL-CISO, US-MIDA-PJM, US-MIDW-MISO, US-NW-BPAT, US-TEX-ERCO
Azure	AU-NSW, BR-CS, CA-ON, DE, FR, GB, IE, IN-WE, IT-NO, JP-TK, KR, NL, PL, SE, SG, US-CAL-CISO, US-MIDA-PJM, US-MIDW-MISO, US-NW-BPAT, US-TEX-ERCO

the experiment, the base case, are: AU-NSW, AU-VIC, BR-CS, CA-ON, DE, ES, FR, GB, IE, IN-WE, IT-NO, JP-KN, JP-TK, KR, NL, PL, SE, SG, US-CAL-CISO, US-MIDA-PJM, US-MIDW-MISO, US-NW-BPAT, and US-TEX-ERCO. Table 4.2 shows the regions included in each network set, seventeen regions in AWS, twenty-two in GCP, and twenty in Azure.

Results Figure 4.8 provides an illustration of how requests are distributed between the different regions, the Google Cloud Platform network specifically in this case. The bar plot, figure 4.9, offers a view of how each network compares to the other and to the base case of every region considered in the experiment. The plot shows that out of the set of regions used in the experiment, the AWS network produced the highest number of emissions, about 928,000,000 gCO_2eq, which is slightly more than the 924,000,000 gCO_2eq generated with the Azure network. The base case of the entire set of regions follows as the second lowest emissions produced, 902,000,000, and the GCP network with the smallest number of emissions, about 885,000,000 gCO_2eq. Figure 4.10 illustrates that the AWS run of the experiment had the highest emissions for many intervals, while GCP was often among the lowest. The AWS network had the smallest subset in the set of regions, while GCP had the largest subset, as seen in Table 4.2. This explains some of the discrepancy, as more regions mean more options to find the lowest to shift to at a given time step, but the base case of the entire set of regions had higher emissions than GCP in figure 4.9. This was due to CarbonMin determining the resource capacity of regional data centers in part by dividing the total

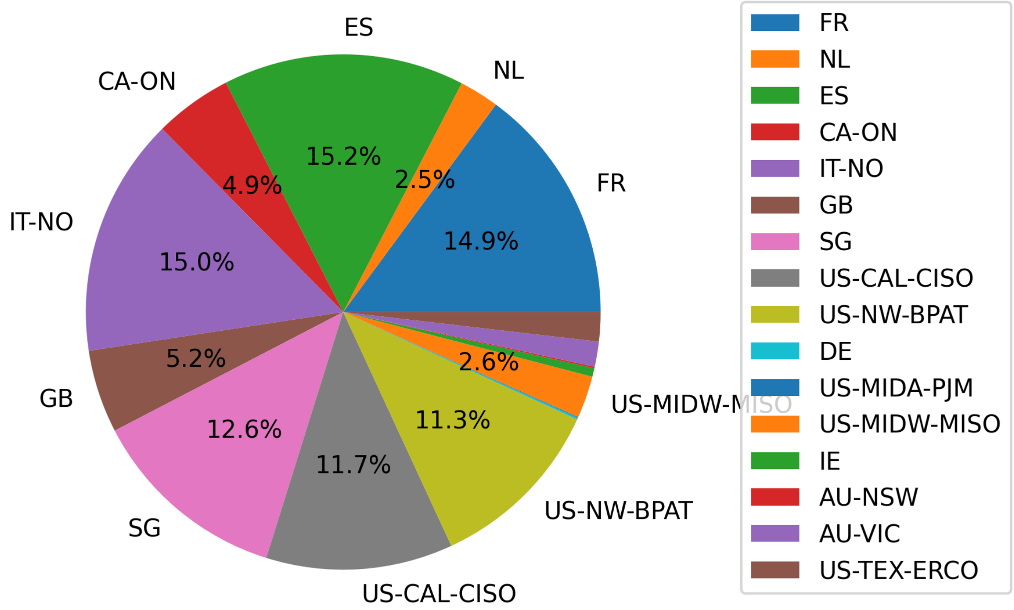


Figure 4.8: Pie chart of the percentage of requests redirected to each region in a subset of the Google Cloud Platforms cloud network by CarbonMin.

requests at the time step by the number of regions, so more potential regions to shift to result in less capacity in the regions. This means that of the set of regions, the GCP network had more options than the other networks to find the lowest carbon intensities. The base case on the other hand was bloated with many highly carbon intensive regions which lower the individual resource capacity of each data center.

4.3 Discussion

In this chapter, Section 4.1 illustrated that carbon-aware workload shifting algorithms can be faithfully implemented and run within the simulation framework developed within this thesis. Additionally, Section 4.2 exhibited the features provided by the simulation framework to further aid the development of carbon-aware workload shifting algorithms.
A feature like the manipulation of time steps in the carbon intensity data enables easier comparisons by showing how different time steps in the car-

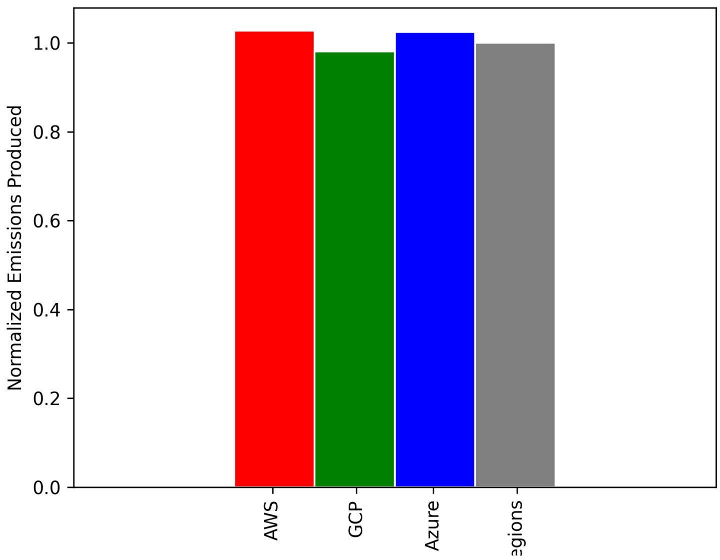


Figure 4.9: Total emissions created from CarbonMin run with each cloud network, as well as the base case of the entire set of regions used in the experiment. Normalization was done by dividing the total emissions of a given network by the total emissions of the base case.

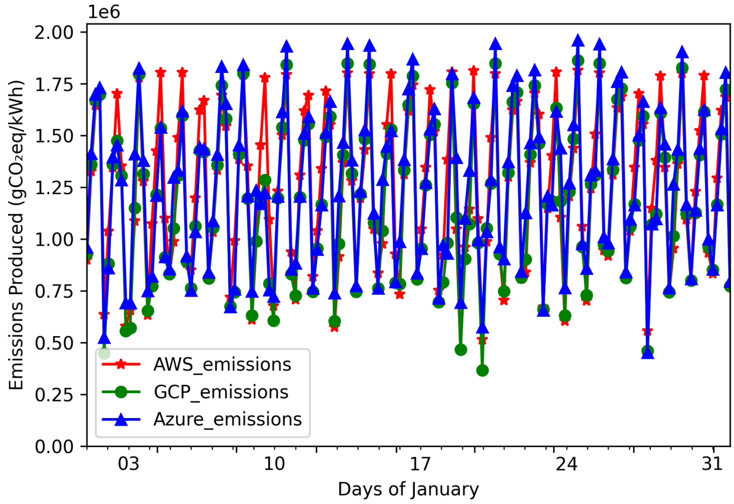


Figure 4.10: Line plot of the emissions CarbonMin produces with each cloud network every fifth hour.

bon intensity data may affect the performance of a given algorithm. As Section 4.2.1 shows, different time steps can alter the outcome from the workload shifting algorithm substantially. For example in the experiment, the Californian run with hourly intervals produced approximately 150% of the carbon emissions of the thirty minute interval run for the same region, despite the original carbon intensity data used in both runs matching. Potentially, some future algorithms may perform better or worse at different time steps, so this feature aids additional development within the space.
Other features like the ability to choose different carbon intensity datasets varying in type or source, the ability to alter the experiment duration, the ability to run algorithms on additional workloads beyond those custom designed for the specific algorithm, and the ability to choose the cloud provider network infrastructure used in experiments also enable algorithms implemented within the simulation framework to be tested in new scenarios without designing an experiment to replicate the scenario from scratch. For instance, rather than creating workloads, users of the framework might co-opt the workload of another algorithm to test an algorithm or one workload can be used in several algorithms to compare them to each other, all with limited additional work for the user.

The combination of features, in addition to the ability to accurately implement workload shifting algorithms within the simulation framework enable the research questions from Section 1.1 to be answered, though the answer is subjective to the parameters of the experiment performed in the framework. Namely, the first question, *How well do existing workload shifting algorithms minimize carbon dioxide emissions?*, is answered by the framework all together with an answer that varies by experiment. The network infrastructure feature answers the second question, *How do different network layouts affect the ability of the algorithms to reduce emissions?*, by allowing carbon-aware workload shifting algorithms to be run with different network infrastructures. In our experiment in Section 4.2.5, we saw that the CarbonMin algorithm performed best in the Google Cloud Platform network subset and performed worst in the Amazon Web Services network subset, which produced slightly more emissions than the Azure network subset. This result may differ depending on the algorithm and the set of regions used in an experiment. The final research question, *How do the algorithms behave under different metrics for measuring the carbon intensity of the electricity powering data centers?*, is answered by the experiment in Section 4.2.2 which shows the sizeable difference in emissions between the same Let's Wait Awhile periodic algorithm with average and with marginal carbon intensity data.

Chapter 5

Conclusions and Future work

This chapter initiates with Section 5.1, describing how well the final state of the thesis fulfilled its original objectives. We follow this with a discussion of the short-comings that limited the project in Section ??. Section 5.2 details directions that the framework may be taken in the future.

5.1 Conclusions

We believe that we have achieved the original goals that we had set upon in Chapter 1. Namely, we have developed a simulation framework that is able to run carbon-aware workload shifting algorithms within it that are identical or close to their counterparts outside the simulator, as shown in Section 4.1. In addition, Section 4.2 shows that we are able run the algorithms in situations other than those they were explicitly designed for, to see how they perform in the new scenarios. We have found through our work and results, that the topic of carbon-aware workload shifting is quite large and ever evolving. There is a great deal of minutiae involved which can cause significant differences in the performance results of the algorithms. For this reason, we believe that workload shifting algorithms should be tested in a variety of different scenarios to fully ascertain their effectiveness.

5.2 Future work

Future work on the simulation framework could include the implementation of additional workloads, carbon-aware workload shifting algorithms, and carbon intensity datasets. An aspect that could be expanded upon are trace-based workload implementations, as an Azure server trace [17] is implemented

to provide a workload within the simulation framework, but due to the algorithms that have been implemented being designed for very different workloads, it was not entirely tested nor included in the evaluation experiments. This Azure workload implementation could be tested and generalized more extensively. Specifically, the workload currently implemented only takes into consideration the five most frequent function calls, so it may be made more all encompassing. Likewise, workloads from Google Cloud [19] or other data center traces could be added to supplement the available workloads.
Another direction that could be taken is to add onto the features provided by the simulation framework. For instance, a standardized format with which to interact with location information would be highly useful. This would increase the interchangeability of the modules within the framework by ensuring the location data that may be provided in a given carbon intensity dataset or workload can definitively be utilized by a given algorithm, rather than there existing potential incompatibility. A more substantial addition might be a feature that is able to track economic cost that might be occurred from a given algorithm, operating with a given workload, and utilizing a given set of data centers. While this is outside of the scope of the thesis, it would be useful as the users of the algorithms in practice would not operate in a vacuum, but rather they would be affected by economic forces.

Bibliography

[1] AWS Services and Regions, . URL https://www.aws-services.info/regions.html.

[2] Azure Availability Zones | Azure Region Information, . URL https://www.azurespeed.com/Information/AzureAvailabilityZones.

[3] Choose the Right Azure Region for You | Microsoft Azure, . URL https://azure.microsoft.com/en-us/explore/global-infrastructure/geographies.

[4] Discover our data center locations, . URL https://www.google.com/about/datacenters/locations/.

[5] GCP List of 31 Regions and 94 Zones, . URL https://www.economize.cloud/resources/gcp/regions-zones-map/.

[6] Global Locations - Regions & Zones, . URL https://cloud.google.com/about/locations.

[7] Carbon Intensity, . URL https://carbonintensity.org.uk/.

[8] Carbon Intensity calculations Â· electricitymaps/electricitymaps-contrib Wiki Â· GitHub, . URL https://github.com/electricitymaps/electricitymaps-contrib/wiki/Carbon-intensity-calculations.

[9] chat.openai.com Traffic Analytics, Ranking & Audience [August 2024], . URL https://www.similarweb.com/website/chat.openai.com/.

[10] Data Signals Overview, . URL https://watttime.org/data-science/data-signals/.

[11] Electricity Maps | Pricing, . URL https://www.electricitymaps.com/pricing.

[12] implemented vessim.TimeSeriesApi for strategies Â· dos-group/lets-wait-awhile@331e6a6 Â· GitHub, . URL https://github.com/dos-group/lets-wait-awhile/commit/331e6a66a7ed3d6410cecd9526084ad322814945.

[13] Live 24/7 COâ emissions of electricity consumption, . URL http://electricitymap.tmrow.co.

[14] Methodology, . URL https://www.electricitymaps.com/methodology.

[15] Methodology + Validation, . URL https://watttime.org/data-science/methodology-validation/.

[16] Our data centers now work harder when the sun shines and wind blows, . URL https://blog.google/inside-google/infrastructure/data-centers-work-harder-sun-shines-wind-blows/.

[17] Azure/AzurePublicDataset, May 2024. URL https://github.com/Azure/AzurePublicDataset. original-date: 2017-08-18T17:37:45Z.

[18] Electricity 2024 - Analysis and forecast to 2026. 2024. URL https://iea.blob.core.windows.net/assets/18f3ed24-4b26-4c83-a3d2-8a1be51c8cc8/Electricity2024-Analysisandforecastto2026.pdf.

[19] google/cluster-data, May 2024. URL https://github.com/google/cluster-data. original-date: 2015-07-29T17:52:23Z.

[20] Claudia Bergroth, Olle JÃ¤rv, Henrikki Tenkanen, Matti Manninen, and Tuuli Toivonen. A 24-hour population distribution dataset based on mobile phone data from Helsinki Metropolitan Area, Finland. *Scientific Data*, 9(1):39, February 2022. ISSN 2052-4463. doi: 10.1038/s41597-021-01113-4. URL https://www.nature.com/articles/s41597-021-01113-4. Publisher: Nature Publishing Group.

[21] Rajesh Bose, Sandip Roy, Haraprasad Mondal, Dipan Roy Chowdhury, and Srabanti Chakraborty. Energy-efficient approach to lower the carbon emissions of data centers. *Computing*, 103(8):1703–1721, August 2021. ISSN 1436-5057. doi: 10.1007/s00607-020-00889-4. URL https://doi.org/10.1007/s00607-020-00889-4.

[22] Raphael Calel, Jonathan Colmer, Antoine DechezleprÃªtre, and Matthieu Glachant. Do carbon offsets offset carbon? ISSN 1556-5068. doi: 10.2139/ssrn.3950103. URL https://www.ssrn.com/abstract=3950103.

[23] Zhiwei Cao, Xin Zhou, Han Hu, Zhi Wang, and Yonggang Wen. Toward a systematic survey for carbon neutral data centers. 24(2):895–936. ISSN 1553-877X. doi: 10.1109/COMST.2022.3161275. URL https://ieeexplore.ieee.org/abstract/document/9739671. Conference Name: IEEE Communications Surveys & Tutorials.

[24] Andrew A Chien, Liuzixuan Lin, Hai Nguyen, Varsha Rao, Tristan Sharma, and Rajini Wijayawardana. Reducing the Carbon Impact of Generative AI Inference (today and in 2035). In *Proceedings of the 2nd Workshop on Sustainable Computer Systems*, pages 1–7, Boston MA USA, July 2023. ACM. ISBN 9798400702426. doi: 10.1145/3604930.3605705. URL https://dl.acm.org/doi/10.1145/3604930.3605705.

[25] Olivier Corradi. Marginal vs average: which one to use for real-time decisions? URL https://www.electricitymaps.com/blog/marginal-vs-average-real-time-decision-making.

[26] Anders Fremstad, Mark Paul, and Anthony Underwood. Work hours and co 2 emissions: Evidence from u.s. households. *Review of Political Economy*, 31:42–59, 01 2019. doi: 10.1080/09538259.2019.1592950.

[27] Pieter Gagnon and Wesley Cole. Planning for the evolution of the electric grid with a long-run marginal emission rate. *iScience*, 25(3), March 2022. ISSN 2589-0042. doi: 10.1016/j.isci.2022.103915. URL https://www.cell.com/iscience/abstract/S2589-0042(22)00185-7. Publisher: Elsevier.

[28] Udit Gupta, Young Geun Kim, Sylvia Lee, Jordan Tse, Hsien-Hsin S. Lee, Gu-Yeon Wei, David Brooks, and Carole-Jean Wu. Chasing carbon: The elusive environmental footprint of computing. 42(4):37–47. ISSN 1937-4143. doi: 10.1109/MM.2022.3163226. URL https://ieeexplore.ieee.org/document/9744492. Conference Name: IEEE Micro.

[29] RenÃ© Itten, Roland Hischier, Anders S. G. Andrae, Jan C. T. Bieser, Livia Cabernard, Annemarie Falke, Hugues Ferreboeuf, Lorenz M. Hilty, Regula L. Keller, Etienne Lees-Perasso, Chris Preist, and Matthias Stucki. Digital transformationâlife cycle assessment of digital services, multifunctional devices and cloud computing. 25(10):2093–2098. ISSN 1614-7502. doi: 10.1007/s11367-020-01801-0. URL https://doi.org/10.1007/s11367-020-01801-0.

[30] Dara Kerr. AI brings soaring emissions for Google and Microsoft, a major contributor to climate change. *NPR*, July

2024. URL https://www.npr.org/2024/07/12/g-s1-9545/ai-brings-soaring-emissions-for-google-and-microsoft-a-major-contributor-to-clima

[31] LoÃ¯c Lannelongue, Hans-Erik G. Aronson, Alex Bateman, Ewan Birney, Talia Caplan, Martin Juckes, Johanna McEntyre, Andrew D. Morris, Gerry Reilly, and Michael Inouye. GREENER principles for environmentally sustainable computational science. 3(6):514–521. ISSN 2662-8457. doi: 10.1038/s43588-023-00461-y. URL https://www.nature.com/articles/s43588-023-00461-y. Number: 6 Publisher: Nature Publishing Group.

[32] Matthew D. Leonard, Efstathios E. Michaelides, and Dimitrios N. Michaelides. Energy storage needs for the substitution of fossil fuel power plants with renewables. *Renewable Energy*, 145:951–962, January 2020. ISSN 0960-1481. doi: 10.1016/j.renene.2019.06.066. URL https://www.sciencedirect.com/science/article/pii/S0960148119308948.

[33] Liuzixuan Lin and Andrew A Chien. Adapting datacenter capacity for greener datacenters and grid. In *Proceedings of the 14th ACM International Conference on Future Energy Systems*, e-Energy '23, pages 200–213. Association for Computing Machinery. ISBN 9798400700323. doi: 10.1145/3575813.3595197. URL https://dl.acm.org/doi/10.1145/3575813.3595197.

[34] Diptyaroop Maji, Ben Pfaff, Vipin P R, Rajagopal Sreenivasan, Victor Firoiu, Sreeram Iyer, Colleen Josephson, Zhelong Pan, and Ramesh K Sitaraman. Bringing Carbon Awareness to Multi-cloud Application Delivery. In *Proceedings of the 2nd Workshop on Sustainable Computer Systems*, pages 1–6, Boston MA USA, July 2023. ACM. ISBN 9798400702426. doi: 10.1145/3604930.3605711. URL https://dl.acm.org/doi/10.1145/3604930.3605711.

[35] Diptyaroop Maji, Noman Bashir, David Irwin, Prashant Shenoy, and Ramesh K. Sitaraman. The Green Mirage: Impact of Location- and Market-based Carbon Intensity Estimation on Carbon Optimization Efficacy. In *The 15th ACM International Conference on Future and Sustainable Energy Systems*, pages 13–24, June 2024. doi: 10.1145/3632775.3639587. URL http://arxiv.org/abs/2402.03550. arXiv:2402.03550 [cs].

[36] Peter Mell and Tim Grance. The NIST Definition of Cloud Computing. Technical Report NIST Special Publication (SP) 800-145, Na-

tional Institute of Standards and Technology, September 2011. URL https://csrc.nist.gov/pubs/sp/800/145/final.

[37] Pei Pei, Zongjie Huo, Oscar SanjuÃ¡n MartÃnez, and RubÃ©n GonzÃ¡lez Crespo. Minimal green energy consumption and workload management for data centers on smart city platforms. 12(8):3140. ISSN 2071-1050. doi: 10.3390/su12083140. URL https://www.mdpi.com/2071-1050/12/8/3140. Number: 8 Publisher: Multidisciplinary Digital Publishing Institute.

[38] Ana RadovanoviÄ, Ross Koningstein, Ian Schneider, Bokan Chen, Alexandre Duarte, Binz Roy, Diyue Xiao, Maya Haridasan, Patrick Hung, Nick Care, Saurav Talukdar, Eric Mullen, Kendal Smith, MariEllen Cottman, and Walfredo Cirne. Carbon-aware computing for datacenters. 38(2):1270–1280. ISSN 1558-0679. doi: 10.1109/TPWRS.2022.3173250. URL https://ieeexplore.ieee.org/abstract/document/9770383. Conference Name: IEEE Transactions on Power Systems.

[39] Hannah Smith. climateaction-tech/grid-aware-software. URL https://github.com/climateaction-tech/grid-aware-software. original-date: 2023-11-02T17:39:54Z.

[40] Thanathorn Sukprasert, Noman Bashir, Abel Souza, David Irwin, and Prashant Shenoy. On the Implications of Choosing Average versus Marginal Carbon Intensity Signals on Carbon-aware Optimizations. In *The 15th ACM International Conference on Future and Sustainable Energy Systems*, pages 422–427, Singapore Singapore, June 2024. ACM. ISBN 9798400704802. doi: 10.1145/3632775.3661953. URL https://dl.acm.org/doi/10.1145/3632775.3661953.

[41] Thanathorn Sukprasert, Abel Souza, Noman Bashir, David Irwin, and Prashant Shenoy. On the Limitations of Carbon-Aware Temporal and Spatial Workload Shifting in the Cloud, March 2024. URL http://arxiv.org/abs/2306.06502. arXiv:2306.06502 [cs, eess].

[42] Mueen Uddin and Azizah Abdul Rahman. Energy efficiency and low carbon enabler green IT framework for data centers considering green metrics. 16(6):4078–4094. ISSN 1364-0321. doi: 10.1016/j.rser.2012.03.014. URL https://www.sciencedirect.com/science/article/pii/S1364032112001979.

[43] OAR US EPA. Renewable Energy Certificates (RECs), January 2022. URL `https://www.epa.gov/green-power-markets/renewable-energy-certificates-recs`.

[44] Jaylen Wang, Udit Gupta, and Akshitha Sriraman. Peeling back the carbon curtain: Carbon optimization challenges in cloud computing. In *Proceedings of the 2nd Workshop on Sustainable Computer Systems*, HotCarbon '23, pages 1–7. Association for Computing Machinery. ISBN 9798400702426. doi: 10.1145/3604930.3605718. URL `https://dl.acm.org/doi/10.1145/3604930.3605718`.

[45] Philipp Wiesner and Lauritz Thamsen. LEAF: Simulating Large Energy-Aware Fog Computing Environments. In *2021 IEEE 5th International Conference on Fog and Edge Computing (ICFEC)*, pages 29–36, May 2021. doi: 10.1109/ICFEC51620.2021.00012. URL `https://ieeexplore.ieee.org/document/9458907`.

[46] Philipp Wiesner, Ilja Behnke, Dominik Scheinert, Kordian Gontarska, and Lauritz Thamsen. Let's wait awhile: How temporal workload shifting can reduce carbon emissions in the cloud. In *Middleware'21: 22nd International Middleware Conference*. ACM, 2021. doi: 10.1145/3464298.3493399.

[47] Philipp Wiesner, Ilja Behnke, Paul Kilian, Marvin Steinke, and Odej Kao. Vessim: A Testbed for Carbon-Aware Applications and Systems, June 2024. URL `http://arxiv.org/abs/2306.09774`. arXiv:2306.09774 [cs, eess].

[48] Minxian Xu and Rajkumar Buyya. Managing renewable energy and carbon footprint in multi-cloud computing environments. *Journal of Parallel and Distributed Computing*, 135:191–202, January 2020. ISSN 0743-7315. doi: 10.1016/j.jpdc.2019.09.015. URL `https://www.sciencedirect.com/science/article/pii/S0743731519303132`.

[49] Eleni Zampou, ELTRUN Research Center, Athens University of Economics and Business, Greece, Ioannis Mourtos, ELTRUN Research Center, Athens University of Economics and Business, Greece, Katerina Pramatari, ELTRUN Research Center, Athens University of Economics and Business, Greece, Stefan Seidel, and University of Liechtenstein, Liechtenstein. A design theory for energy and carbon management systems in the supply chain the quest for innovation in information systems research: Recognizing, stimulating, and promoting novel and useful knowl-

edge. 23(1):329–371. ISSN 15369323. doi: 10.17705/1jais.00725. URL https://aisel.aisnet.org/jais/vol23/iss1/2/.