Carbon-Aware Workload Scheduling in Modern Cloud Platforms: Mechanisms, Practices, and Challenges

Tie Haoran

Department of Computer Science, Your University your.email@example.com

Abstract. Cloud computing's rapid growth has raised concerns about its environmental impact, particularly its carbon footprint. In response, the concept of *carbon-aware scheduling* has emerged—dynamically placing or delaying workloads based on real-time or forecasted carbon intensity of electricity grids. This report explores the technical foundations, industry implementations, and practical challenges of carbon-aware workload scheduling in modern cloud platforms. We focus on two core strategies: time-shifting (temporal deferral) and geo-shifting (spatial relocation), analyzing their mechanisms, enabling technologies, and limitations. The discussion emphasizes real-world feasibility over theoretical novelty, making it accessible to practitioners and students alike. No new algorithms are proposed; instead, we synthesize current practices to illustrate how sustainability is becoming a first-class concern in cloud architecture.

1 Introduction

Cloud computing has become the backbone of modern digital infrastructure, powering everything from enterprise applications to large-scale machine learning workloads. However, this rapid growth comes at an environmental cost. Data centers worldwide consumed approximately 1–2% of global electricity in recent years, and their carbon footprint is expected to grow as demand increases [2]. In response, major cloud providers—such as Microsoft Azure, Google Cloud Platform, and Amazon Web Services—have committed to ambitious sustainability goals, including achieving net-zero carbon emissions by 2030 or earlier.

A key technical strategy to reduce the carbon impact of cloud workloads is **carbon-aware scheduling**: the practice of dynamically placing or delaying computational tasks based on real-time or forecasted carbon intensity of the local electricity grid. This approach falls under the broader umbrella of green and sustainable software engineering, which seeks to minimize the environmental impact of software systems throughout their lifecycle [1]. Unlike traditional scheduling that optimizes for latency, cost, or resource utilization alone, carbon-aware scheduling introduces **environmental sustainability** as a first-class optimization objective.

This report explores the principles, mechanisms, and real-world implementations of carbon-aware workload scheduling in contemporary cloud platforms.

We focus on two core strategies—**time-shifting** and **geo-shifting**—and analyze how they are operationalized using publicly available data sources and open-source tooling. We also discuss practical limitations and open challenges that prevent widespread adoption.

Our contribution is not a new algorithm, but rather a **comprehensive synthesis** of emerging practices, system architectures, and policy considerations that define this niche yet rapidly evolving area of sustainable cloud computing.

2 Background: Carbon Intensity and Grid Dynamics

2.1 What Is Carbon Intensity?

Carbon intensity measures the amount of carbon dioxide equivalent emitted per unit of electricity generated, typically expressed in grams of CO_2e per kilowatt-hour:

$$\mathrm{CI}(t) = \frac{\text{Total } CO_2\text{e emissions from grid at time } t}{\text{Total electricity generated at time } t} \tag{1}$$

This metric varies significantly by region and time due to differences in energy mix—such as coal, natural gas, nuclear, wind, or solar—and fluctuations in electricity demand. For example, carbon intensity in California may drop below 100 gCO_2/kWh during midday when solar generation peaks, but exceed 400 gCO_2/kWh at night when fossil-fuel plants dominate.

2.2 Why Time and Location Matter

Because renewable energy sources like solar and wind are intermittent, the carbon intensity of a power grid is **temporally and geographically dynamic**. This variability creates opportunities for smarter scheduling:

- **Temporal flexibility**: Delay non-urgent workloads—such as batch processing, data analytics, or model training—to periods when the local grid's carbon intensity is lower.
- **Geographic flexibility**: Route workloads to data centers located in regions where the grid is cleaner at that moment.

These two ideas form the foundation of carbon-aware scheduling in cloud environments and have been explored in multi-cloud contexts to jointly optimize cost, performance, and emissions [3].

3 Core Strategies in Carbon-Aware Scheduling

3.1 Time-Shifting (Temporal Shifting)

Time-shifting involves **deferring flexible workloads** to future time windows when the local grid's carbon intensity is expected to be lower. This approach

requires two conditions: first, the workload must tolerate some execution delay (i.e., it has slack time); second, the system must have access to reliable short-term forecasts of carbon intensity, usually 24 to 48 hours ahead.

The objective is to minimize total carbon emissions over the possible execution window. If a job consumes a constant amount of power P (in kW) and runs for duration D, its total emissions depend on when it executes:

$$E = \sum_{t=t_0}^{t_0+D} P \cdot \Delta t \cdot \mathrm{CI}(t) \cdot x(t) \tag{2}$$

where:

- Δt is the time step (e.g., one hour),
- $x(t) \in \{0, 1\}$ indicates whether the job runs at time t,
- and the sum of $x(t)$ over the window equals the required number of time slots.

In practice, many systems use simple heuristics—such as selecting the next available low-carbon window—rather than solving complex optimization problems, due to uncertainty in forecasts. Recent work demonstrates that even basic time-shifting can yield significant emission reductions in batch-oriented cloud workloads [3].

3.2 Geo-Shifting (Spatial Shifting)

Geo-shifting routes a workload to a **different geographic region** where the current or predicted carbon intensity is lower. This strategy can yield larger savings than time-shifting but introduces additional complexity.

Key requirements include:

- The ability to deploy workloads across multiple regions.
- Tolerance for increased network latency or data transfer costs.
- Compliance with data sovereignty regulations, which may restrict where user data can be processed.

The potential carbon saving is proportional to the difference in carbon intensity between regions. For a workload consuming E_{comp} kilowatt-hours, running it in a region with CI = 200 $\mathrm{gCO_2/kWh}$ instead of one with CI = 500 $\mathrm{gCO_2/kWh}$ avoids $300 \cdot E_{\mathrm{comp}}$ grams of $\mathrm{CO_2e}$.

However, geo-shifting is only suitable for **stateless or loosely coupled** applications. Interactive services—such as web APIs or real-time dashboards—typically cannot be moved far from end users without degrading user experience.

Large cloud providers with global infrastructure have demonstrated successful geo-shifting for specific workloads, such as video transcoding or scientific simulations, by leveraging real-time grid data and private interconnect networks. Multi-cloud orchestration frameworks further enhance this capability by enabling cross-provider workload migration based on carbon signals [4, 3].

4 System Architecture and Enabling Technologies

Modern carbon-aware systems rely on three interconnected layers: data sources, middleware, and orchestration engines.

4.1 Carbon Data Providers

Real-time and forecasted carbon intensity data are provided through public or commercial APIs. Examples include Electricity Maps, which offers historical and live grid data for over 100 zones worldwide, and WattTime, which provides marginal carbon intensity forecasts tailored for automated decision-making. National grid operators—such as CAISO in California or National Grid ESO in the UK—also publish open datasets that researchers and developers can use.

These services typically deliver carbon intensity values at hourly resolution, often accompanied by confidence intervals or uncertainty estimates.

4.2 Middleware and SDKs

To bridge raw carbon data with application logic, new open-source tools have emerged. One notable example is the Carbon Aware SDK, developed by the Green Software Foundation. It abstracts differences between data providers and offers simple functions like "get best run time" or "get greenest region."

Additionally, power monitoring agents such as Kepler and Scaphandre estimate real-time energy consumption of containers and virtual machines in Kubernetes environments. While not directly carbon-aware, they provide the foundational metrics needed to correlate compute activity with emissions.

4.3 Integration with Orchestrators

In cloud-native platforms like Kubernetes, custom schedulers can incorporate carbon intensity into node selection. For example, a scheduler plugin might assign higher scores to nodes in regions with lower current CI, or delay CronJobs until a low-carbon window opens.

Although such features are not yet part of mainstream Kubernetes distributions, experimental implementations and internal tools at major tech companies show that integration is technically feasible and increasingly practical. Recent research proposes a Carbon-Aware Cloud Architecture (CACA) that integrates real-time carbon data into dynamic multi-cloud scheduling decisions via weighted optimization models, demonstrating that sustainability can be embedded as a first-class principle in production systems [4].

5 Industry Practices and Case Studies

Several leading cloud providers have begun integrating carbon awareness into their platforms.

One major provider introduced carbon-aware virtual machine placement for batch workloads, using grid forecasts to select both start times and regions. Internal evaluations showed emission reductions of up to 30% for flexible AI training jobs, with no impact on completion deadlines.

Another company has shifted focus from annual renewable energy matching to **24/7 carbon-free energy**—ensuring that every hour of electricity consumption is matched with clean generation in the same grid. Their internal scheduler moves non-urgent compute tasks across dozens of global data centers to maximize this hourly matching rate, achieving over 90% carbon-free operation in several regions.

A third provider offers customers a dashboard to view estimated emissions by service and region, encouraging manual selection of greener deployment zones. While automated scheduling is not yet available, the transparency helps organizations make informed decisions.

6 Limitations and Challenges

Despite promising progress, several barriers hinder broader adoption of carbon-aware scheduling.

First, **forecast accuracy** remains limited. Carbon intensity predictions beyond 24 hours are often unreliable, and acting on poor forecasts can inadvertently increase emissions.

Second, most real-world applications have **limited flexibility**. Only a fraction of cloud workloads—typically those that are batch-oriented or offline—can be delayed or relocated without violating service-level agreements.

Third, there is **no industry-wide standard** for carbon data formats, APIs, or metrics. This fragmentation makes it difficult to build portable, multi-cloud carbon-aware applications.

Fourth, current models usually ignore **embodied carbon**—the emissions from manufacturing servers and networking equipment—as well as the energy used in data transmission across networks.

Finally, without strong economic incentives—such as carbon taxes or regulatory mandates—many organizations continue to prioritize cost and performance over sustainability, even when greener options exist.

7 Conclusion and Future Directions

Carbon-aware workload scheduling represents a pragmatic and technically feasible path toward more sustainable cloud computing. By exploiting the natural variability of power grids in time and space, cloud platforms can significantly reduce emissions for a meaningful subset of workloads—particularly those in data engineering, scientific computing, and artificial intelligence.

For practitioners, immediate actions include assessing workload flexibility, selecting cloud regions with cleaner grids, and adopting open tools that expose

carbon metrics. For system designers, embedding carbon awareness into schedulers, monitoring stacks, and developer workflows will be essential.

Looking ahead, research opportunities abound: robust scheduling under uncertainty, standardized carbon APIs, integration with financial cost models, and redefinition of SLAs to include environmental criteria. As climate pressures mount and regulations tighten, carbon-aware computing is poised to evolve from an optional optimization to a core pillar of responsible cloud architecture—where efficiency is measured not just in speed or dollars, but in the planet's carrying capacity.

References

1. Mourão, B.C., Karita, L., do Carmo Machado, I.: Green and Sustainable Software Engineering - a Systematic Mapping Study. In: Proceedings of the XVII Brazilian Symposium on Software Quality. pp. 121–130. SBQS '18, Curitiba, Brazil (2018). https://doi.org/10.1145/3275245.3275258
2. Thakur, S., Chaurasia, A.: Towards Green Cloud Computing: Impact of carbon footprint on environment. In: 2016 6th International Conference - Cloud System and Big Data Engineering (Confluence). pp. 209–213 (2016). https://doi.org/10.1109/CONFLUENCE.2016.7508115
3. Xu, M., Buyya, R.: Managing renewable energy and carbon footprint in multi-cloud computing environments. Journal of Parallel and Distributed Computing **135**, 191–202 (2020). https://doi.org/10.1016/j.jpdc.2019.09.015
4. Muddada, S.T.: Carbon-Aware Cloud Architecture: Dynamic Multi-Cloud Scheduling for Sustainable Computing. Journal of Computer Science and Technology Studies **7**(10), 511–520 (2025). https://doi.org/10.32996/jcsts.2025.7.10.50