Incentives for HPC Users to be More Energy Efficient Through Carbon Credits

Anonymous Author(s)

Abstract

Realizing a shared responsibility between providers and consumers is critical to manage the sustainability of HPC. However, while cost may motivate efficiency improvements by infrastructure operators, broader progress is impeded by a lack of user incentives. We conduct a survey of HPC users that reveals fewer than 30% are aware of their energy consumption, and that energy efficiency is among users' lowest priority concerns. One explanation is that existing pricing models may encourage users to prioritize performance over energy efficiency. We propose two transparent multi-resource pricing schemes, Energy- and Carbon-Based Accounting, that seek to change this paradigm by incentivizing more efficient user behavior. These two schemes charge for computations based on their energy consumption or carbon footprint, respectively, rewarding users who leverage efficient hardware and software. We evaluate these two pricing schemes via simulation, in a prototype, and a user study. We demonstrate there exist trade-offs where existing pricing models would charge 45% more to run the same task on a more efficient machine, but Energy-Based Accounting charges 47% less. When presented to real users, Energy-Based Accounting incentivized people run tasks more efficiently, resulting in 41% energy savings.

1 Introduction

Sustainability of HPC is a shared responsibility between infrastructure *providers* (who buy servers, source electricity, build and cool data centers, etc.) and *consumers* (who write software and decide what, where, and when to compute [1]). These roles should be complementary; however, they currently operate disjointly. Providers tout efficient facilities and investments in renewable energy [2–4] but are constrained by the amount, time, and location of consumer demand. Consumers are told that they can reduce environmental impact by using specific hardware or improving utilization but lack information to accurately account for energy or carbon use. Although many tools and techniques have been developed to improve energy efficiency, they require user adoption [5–8] or at least cooperation between user and provider [9–12]. With computing demands increasing [13] and efficiency improvements slowing, we need users to become conscious of their efficiency and environmental impact.

Existing incentives feed this division of roles. Resource providers often assume all responsibility for energy costs, and thus are motivated to pursue higher efficiency. Users, however, are typically incentivized to choose resources based on opaque pricing or allocation mechanisms. This situation leads to users prioritizing higher performance and cost reduction over potentially more energy-efficient hardware and software. *Innovation in hardware and software efficiency will have little impact without user incentives to adopt more efficient practices.* In this work, we study mechanisms to make sustainability a first-class concern among HPC users.

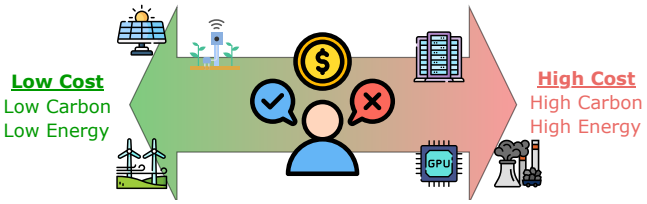


Figure 1. As computing demands increase [13] and efficiency improvements slow, we need users to become conscious of their efficiency and environmental impact to enable sustainable computing.

We first survey more than 300 HPC users to assess knowledge, attitudes, and behaviors in regards to energy consumption. We *find a prevailing lack of awareness of energy consumption, lack of action to reduce energy, disregard for efficiency or sustainability rankings, and a low priority on energy efficiency.*

We then propose to address these issues by *adopting an application's environmental impact as the measure of its resource usage.* We develop two such impact-based accounting approaches: **Energy-Based Accounting (EBA)** which charges jobs based on energy used, and **Carbon-Based Accounting (CBA)**, which charges jobs based on estimated carbon footprint. Thus, a user might be allocated 10 kg carbon emissions ($kgCO_2e$), rather than 100 node hours; be able to estimate the $kgCO_2e$ required for a computation on different machines; and track $kgCO_2e$ rather than node hours. We hypothesize that the adoption of such an accounting scheme across facilities will incentivize users both to run on resources that are energy efficient for their application and to write energy-efficient software, as both choices will allow them to perform more computation for the same cost. We envisage using EBA and CBA for individual HPC system allocations as well as for fungible multi-resource HPC allocations.

To study the effect of impact-based accounting on energy consumption and system utilization, we simulate EBA and CBA with different user behaviors. We then build a CBA prototype that leverages a Function-as-a-Service (FaaS) interface to heterogeneous resources to realize impact-based accounting efficiently. This platform makes user energy consumption

transparent, seamlessly guides users to better resources, and incentivizes sustainable use of computing resources. Finally, we conduct a study of how impact-based accounting changed the behavior of real users in a simulated environment.

The contributions of this paper are:

- A survey of 300+ HPC users to assess awareness of energy consumption and other sustainability metrics.
- Energy- and Carbon-Based Accounting approaches for pricing HPC use, based on energy consumption and carbon footprint, respectively.
- A prototype FaaS platform that employs Carbon-Based Accounting that can guide users to more sustainable computation.
- A user study to evaluate behavior within an Energy-Based Accounting scheme.

2 Motivation

We first examine the prevailing attitudes of HPC users towards sustainable computing practices. Specifically, we developed a survey to understand how users consider energy relative to other concerns in computing and examine their awareness of existing sustainability metrics and techniques. We describe the survey design and summarize key findings.

2.1 Survey Design

We employed Qualtrics, a platform for assembling, distributing, and managing online surveys. We defined 33 questions that examine a participant's familiarity with their energy consumption and other sustainability-related questions. Our target participant is anyone who submits jobs or develops code for HPC machines. As such, we distributed the survey to HPC user groups, facilities, science collaborations known for use of HPC, and user groups of popular HPC tools. Participants were not required to respond to all questions.

2.2 Results

We received 316 responses, of which 192 completed 90% of the survey or more. 166 respondents were located in Europe, 104 in North America, 4 in Oceania, and 4 in China. 73 respondents were graduate students, 97 were early career researchers/engineers, and 99 were senior researchers/engineers. In lieu of screening candidates by the frequency they use HPC resources or their career stage, we relied on participants to self-identify as HPC users and asked them to estimate the amount of node-hours they used HPC resources. We did not observe a significant difference in responses based on career stage or amount of use.

We first examine HPC user awareness of their resource use. 73% of participants were aware of how many node hours a job/workflow consumes, and 70% indicated that they had taken steps to reduce the number of node hours used. This aligns with the more than 80% of users who were very or

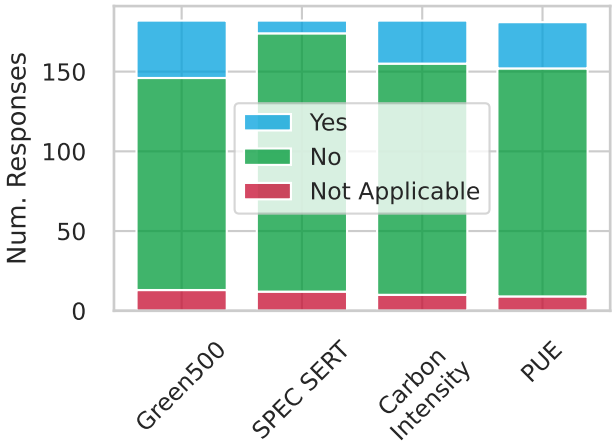


Figure 2. Responses to the question "Are you aware of how the HPC resources you use perform on the following sustainability metrics?" Users do not use existing metrics/rankings when deciding which machine to use to run a job.

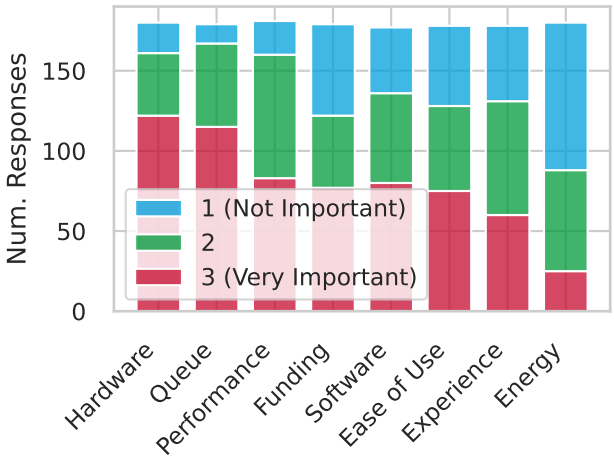


Figure 3. Energy efficiency is among the least important factors for users when choosing where to run a job. Meanwhile, funding is often a very important factor for users.

mildly concerned with finishing their jobs within their allocation, typically measured in node hours. 77% of those concerned about completing their jobs had taken steps to reduce the node hours their jobs consume. **Overall, users are aware of their node-hour usage, concerned about completing their jobs within their allocation, and take steps to reduce their node hours as a result.**

In contrast, only 27% of respondents were aware of the amount of energy their workload consumes and 30% had taken steps to reduce energy use. Counterintuitively, there is not strong overlap between these groups. 39% of people who have taken steps to reduce their energy use were not aware of how much energy their jobs consume. **Few users**

are aware of their energy consumption or have taken steps to reduce it.

We also examine familiarity with rankings assessing efficiency. Rankings such as the Green500 [14] and Graph Green500 [15], and benchmarks like the SPEC Server Efficiency Rating Tool (SERT) [16], reduce machine efficiency to a single number to help providers or users select resources. Carbon intensity captures carbon emissions from producing electricity, which varies based on facility location [17]. Power Usage Efficiency (PUE) captures the fraction of facility electricity that is used for computing compared to cooling or other needs [2]. Participants reported some familiarity with these metrics, with 94 (51%) and 55 (30%) respondents knowing of Green500 and Carbon intensity, respectively. However, this knowledge does not affect user behavior. As shown in Figure 2, few respondents were aware of how the resources they use perform on these metrics. For instance, of the 94 people familiar with the Green500 list, only 36 (20% of all respondents) knew how the machine they were using performed on that ranking. **While users are familiar with metrics designed to improve sustainability, most do not know how they apply to their own machines.**

Given these findings, it is not surprising that energy efficiency is among the least important metrics people use when selecting a machine. Figure 3 shows participant responses when asked how important various parameters were in selecting which machine to use. While 83 respondents said machine performance was very important when selecting which machine to use, only 25 said energy efficiency was very important. This is problematic for providers looking to invest in more efficient options for users: without a shift in incentives, users will prioritize performance over efficiency. **In summary, users do not select machines based on energy efficiency, instead they prioritize hardware availability, queue times, performance and funding.**

3 Methods

We learned that, for the most part, users of HPC resources do not consider energy (or carbon) when deciding what to run, where to run it, and when to run it. However, the study also revealed that users are motivated by limited resource allocations and funding directives. We propose new resource pricing models in which the allocation cost of using a computational resource is based on the energy or carbon consumed by the computation, rather than the execution time or the execution time scaled by machine-specific factors. Thus, users seeking to maximize their allocations will be incentivized towards more sustainable behavior. As we discuss in the following, our approach combines solutions to two problems: (1) How to incorporate both energy and usage into an accounting method; and (2) How to account for carbon.

3.1 Background: Fungible Allocations

We design accounting methods for *fungible* allocations: allocations that can be used for computing on multiple resources. Such allocations are employed, for example, by the US National Science Foundation (NSF) Advanced Cyberinfrastructure Coordination Ecosystem: Services and Support ACCESS [18], Chameleon Cloud [19], or even internally within Google [20]. Fungible allocations provide flexibility by allowing users to select between different machines, which may potentially have different efficiencies. Fungible allocations may be based only on time (e.g., Chameleon Cloud allocates users a constant number of node hours that can be redeemed on any node) or on a mix of time and performance (e.g., ACCESS grants users *service units* that can be exchanged for allocations on more than 30 machines based on machine-specific exchange rates; Google tracks *Google Compute Units* which standardizes core-time to the same amount of computational power on any machine in the fleet).

3.2 Energy-Based Accounting Model

Our **Energy-Based Accounting (EBA)** model *charges users for energy used rather than time spent computing.*

Intel and AMD CPUs support accurate CPU energy readings [21] that can be collected and accounted for by a cluster management tool [22]. Baseboard Management Controllers (BMCs) could also be used, or the energy could be multiplied by the power usage efficiency (PUE) to give a more holistic accounting of the energy a node uses.

One challenge with charging for energy-consumed is the trade-off between potential and actual usage. When accounting for time, i.e., node hours, providers charge based on the potential usage of a resource—a user is charged for a resource regardless of how they use it because it cannot be allocated to another user. Replacing time with energy in the accounting model implies that the end price depends on user activity. This is a double-edged sword: users who write more efficient software are rewarded with reduced costs, but so are users who do not fully use allocated hardware, even though the provider cannot charge others for those resources. In the limit, an incompetent user would be charged a small amount for a large allocation if they put those resources to sleep.

To address this issue, we incorporate the potential use of a node into our calculation. We use a processor's Thermal Design Power (TDP)—the maximum sustained power that a processor is designed to dissipate—as a surrogate for a node's possible utilization. We define a resource's TDP as the sum of the TDPs for all devices on that resource. We then charge users for the average of the actual energy consumed and the amount of energy that potentially could have been consumed had they used the resource fully. For a job j on resource R that uses energy e_j and takes duration d_j, the cost based on the **energy charge** is:

$$\hat{e}_j = (e_j + \beta \cdot d_j \cdot \mathrm{TDP}_R)/2, \quad (1)$$

The parameter β can be used to balance the weight placed on the TDP for scenarios where the TDP of a device is much greater than the typical power use and skews the cost. For this paper, we set $\beta = 1$, so that Equation 1 becomes an unweighted average of the power and TDP.

One consequence of factoring potential use into our charging equation is that energy consumption must still be balanced with job duration. Energy efficiency improvements that result in too much slowdown will increase total cost. Using the average between energy used and TDP keeps accounting simple, understandable, and transparent.

3.3 Carbon-Based Accounting Model

Energy consumption is not the only factor that contributes to the sustainability of a computing facility [23]. Estimating the carbon footprint of a computation gives us a mechanism to account for more holistic impacts. To that end, our **carbon-based accounting (CBA)** model charges users based on the gCO_2e, i.e., grams of carbon dioxide equivalent, used. This footprint encompasses the operational carbon of the electricity used to run a job and a portion of the machine's embodied carbon that we attribute to a job.

We use the **carbon intensity** (I_f) of the electricity grid at facility f to estimate the operational carbon of a job. Carbon intensity estimates the carbon emissions used to generate electricity in gCO_2e per kWh based on the generation source, which varies by location and time. These estimates can be obtained from grid operators or public APIs [17]. Incorporating carbon intensity in the accounting model incentivizes users to choose to run jobs in locations with renewable energy and at times when renewable energy is widely available.

To attribute embodied carbon to a job, we differ from the standard practice of allocating the embodied carbon of a device linearly based on time [24]. Instead we treat the embodied carbon more like a capital expense invested in the machine, that is then depreciated over time as jobs are run on the machine. Our rationale for this approach is that the embodied carbon allocated to a job should be proportional to the utility derived from using the machine. The embodied carbon of a machine is emitted largely before being deployed, through mining materials, fabricating chips, and transporting parts. Users demanding the latest technologies drive providers to acquire new machines. Alternatively, users who leverage older technology allow hardware providers to extend refresh cycles [25], reducing emissions in the long term as fewer machines are manufactured. Thus, users who use a machine earlier in it's lifespan are charged a higher rate.

Specifically, we employ a form of accelerated depreciation called double declining balance [26]. In line with standard guidance and typical refresh periods [25], we assume an HPC machine has a depreciation period of five years which corresponds to a 40% depreciation rate. For a machine with C_f total embodied carbon, the unaccounted-for carbon in year t is:

$$R_f(t) = C_f \cdot (1 - 0.4)^t,$$

the embodied carbon allocated to year t is:

$$D_f(t) = 0.4 \cdot R_f(t),$$

and the carbon-rate per hour of resource utilization is

$$D_f(t)/(24 * 365).$$

Thus the total **carbon charge** for a job j run at facility f with carbon intensity I_f that uses e_j kWh is:

$$c_j = e_j \cdot I_f + d_j \cdot D_f(t)/(24 * 365). \quad (2)$$

4 Simulation Studies

To analyze EBA and CBA at scale, we conduct simulations of real workloads and machines. We modify an existing batch simulator to use the proposed accounting policies on multiple machines [27]. Our goals are to (1) understand how different choices affect the cost of running a workload under EBA and CBA, (2) examine how different accounting models translate to resources used, and (3) explore how low-carbon scenarios may affect the accounting models.

4.1 Machines

We characterize in Table 1 the machines used in our simulation. For each, we indicate: Cores per Node; CPU Thermal Design Power (the maximum heat in watts that a CPU can dissipate); Idle Power (total power consumed by the CPUs when running only the monitoring code); Carbon Rate (see Section 3.3); and Carbon Intensity, based on the yearly average emissions for generating electricity on the grid where the cluster is located, retrieved from Electricity Maps [17].

Three of the machines are in HPC centers: FASTER, Institutional Cluster (IC), and Theta. The fourth machine is a personal computer referred to here as Desktop. Each system is unique, and (as we show below) selecting one over another can yield significantly different results for a user.

4.2 Workload

We use a published dataset of per-job energy use from two HPC clusters [28] over a five-month period. We discard jobs run by the same user with the same walltime and nodes as we consider them to be repetitions of the same app and have the same cross-platform characteristics. Finally, we discard jobs that lack an associated energy value. This reduces the dataset from ~84k jobs to 71,190 unique executions. We repeat each execution twice to generate a workload of 142,380 jobs. Any job can run anywhere, except that 17% of jobs require more cores than are available on the one-node Desktop.

The dataset that we use to construct this workload provides, for each job, energy consumption only on one machine. We use a best-effort approach to extrapolate these data to our machines. To begin, we assume that the reported runtime and energy consumption correspond to running on IC, the

Table 1. Machines used for simulation. TDP=Thermal Design Power, in Watts. Idle power is for all sockets on the node.

Machine	Year Deployed	CPU Model	# of Cores	CPU TDP (W)	Idle Power (W)	Carbon Rate (gCO_2e/h)	Carbon Intensity (gCO_2e/kWh)
TAMU FASTER	2023	2 × Intel Xeon 8352Y	64	205	205	105.2	389
Desktop	2022	Intel Core i7-10700	16	65	6.51	12.2	502
Institutional Cluster	2021	2 × Intel Xeon 6248R	48	205	136	16.7	502
ALCF Theta	2017	Intel KNL 7320	64	215	110	2.0	502

system most similar to the source dataset. We then adapt a method to predict energy use and runtime had jobs been run on the other machines [29]. First, we generate realistic values for hardware performance counters (i.e., LLC Misses/sec., Instructions/sec) for each job using a Gaussian Mixture Model trained on data collected on IC. We then use a KNN [29] trained on a set of benchmark applications [30] to estimate runtime and power consumption on the other machines.

4.3 Policies

To simulate user choice, we define eight machine selection *policies* that select which machine to submit a job to based on the system state (queue times) and job profile (energy use, runtime, or cost on each machine). Each simulated user is limited to one running job on a cluster at a time. The eight policies are:

- **Greedy**: Select machine that minimizes cost, according to the accounting method (EBA or CBA) used.
- **Energy**: Select machine that minimizes energy consumption.
- **Mixed**: Balance runtime and cost. Select machine with the least allocation cost (in terms of EBA or CBA) *unless* another machine can complete the job in half the time, in which case select that machine.
- **EFT** (earliest finish time): Select machine that minimizes completion time, i.e., queue time + runtime.
- **Runtime**: Select machine with shortest runtime.
- **Theta**, **IC**, and **FASTER**: Always select that machine.

4.4 Results with Energy-Based Accounting

Figure 4 depicts the amount of work (in core-hours) completed by a user when applying each policy n a fixed allocation (cost). Specifically, to calculate "work", for each job, we take its average runtime across all the machines multiplied by the number of cores requested. This places higher weight on larger and longer jobs, without favoring any one machine.

In general, the single-machine policies (*Theta*, *IC*, and *FASTER*) and policies that do not consider energy (*EFT* and *Runtime*) perform less work for the same cost. A user using *Theta* is severely disadvantaged by EBA because they submit all jobs to an older machine that is inefficient for most tasks, resulting in high energy consumption. Using *Greedy*, a user

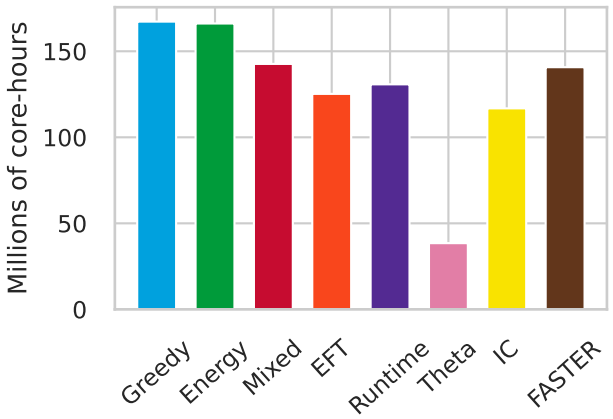


Figure 4. Work completed with fixed allocation using EBA.

is able to complete the most work, because, by definition, they use the cheapest machine for every task. This results in completing 28% more work than when using the performance focused *EFT* policy. A user employing *Energy* can complete 99% of the work done using *Greedy* because the most efficient machine is often the cheapest machine. **Under EBA, by optimizing for energy, a user is indirectly optimizing for cost; in other words, the cost incentivizes users to be more energy efficient.**

Next, we examine how the different policies relate to completion time. Using a single machine is detrimental in terms of completion time because of long queue times. This is visible in Figure 5, which shows the number of jobs completed by different policies over time. From Figure 5, we also see that using *Mixed* a user is able to compute 100,000 jobs as fast as using *EFT* while using less allocation. **By balancing energy efficiency and performance, users can reduce cost under EBA without impacting completion time.**

We now consider only the multi-machine policies and examine how each policy affects total energy consumption. Figure 6 shows the energy consumed using each policy over the full workload (i.e., not considering the initial allocation). Logically, we see that using *Energy* consumes the least energy. A user optimizing cost instead of the energy by using *Greedy* resulted in only 2% more energy use. In contrast,

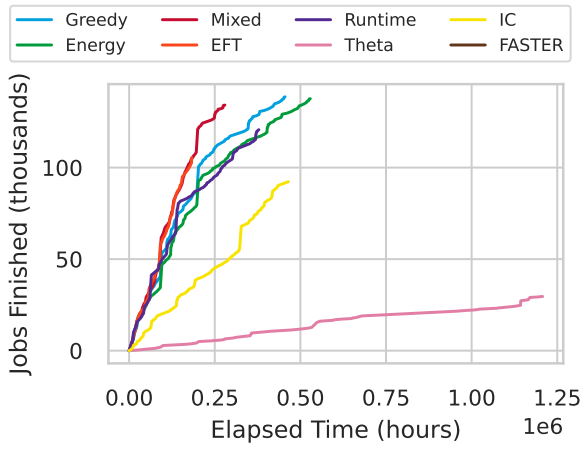


Figure 5. Number of jobs completed over time with a fixed allocation using EBA.

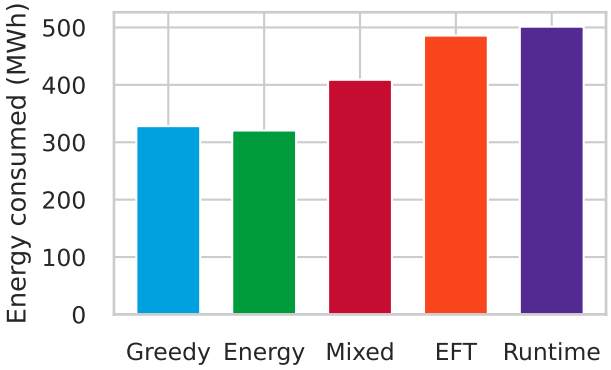


Figure 6. Energy consumed per policy over full workload.

applying *EFT* or *Runtime* results in 51% or 57% more energy respectively. Performance and efficiency are not always aligned. **A user prioritizing the speed of their workload in terms of either makespan or core hours may make inefficient decisions.**

Finally, we investigate how EBA might affect the usage of each machine. Figure 7 shows how jobs are distributed with each policy. *Greedy* and *Energy* policies distribute jobs similarly to FASTER, Desktop, and IC. *Greedy* and *Energy* allocate no tasks to Theta because it is neither the cheapest nor the most energy efficient. In contrast, *Mixed* distributes tasks to all four machines to reduce the completion time. **Overall, EBA may lower utilization of inefficient machines to reduce energy consumption, but users may choose to use less efficient machines at a higher cost to avoid long queues.**

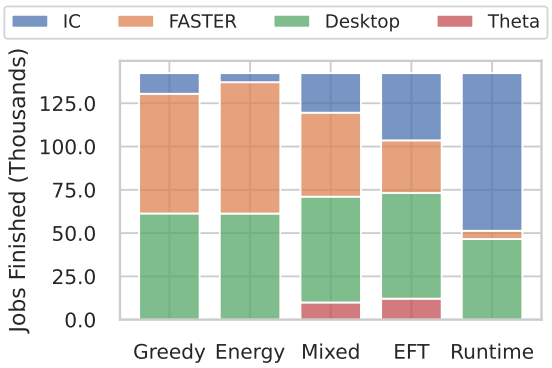


Figure 7. Distribution of jobs over machines by policy.

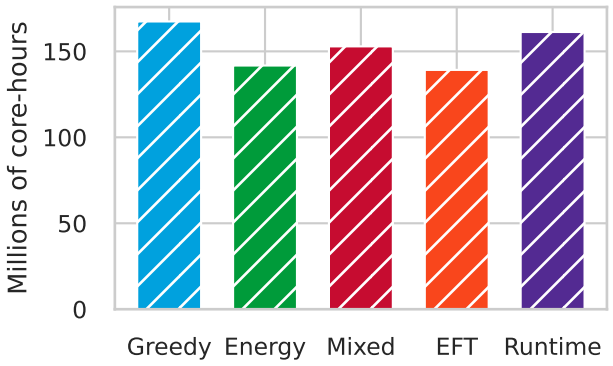


Figure 8. Work completed with fixed allocation using CBA.

Table 2. Energy and carbon used when deploying each policy while computing the full workload when using CBA.

Policy	Energy used (MWh)	Carbon used ($KgCO_2e$)
Greedy	338	**186**
Energy	**320**	190
Mixed	393	214
EFT	484	262
Runtime	501	261

4.5 Results with Carbon-Based Accounting

Next, we analyze CBA as described in Section 3.3. Although carbon intensity varies throughout the day and by season, we use the yearly average intensity for the simulation. For Figure 8, we allow a user employing *Greedy* to run the same amount of work as in Figure 4. We see that using *Energy*, a user can run comparably fewer jobs than under EBA, while using *Runtime*, a user is able to run comparably more. This results from the high carbon rate of FASTER: while often selected by *Energy* (see Figure 7), it has a much higher embodied carbon rate. Since *Energy* favors FASTER and *Runtime* favors IC, and neither adapts to price, deploying *Runtime*

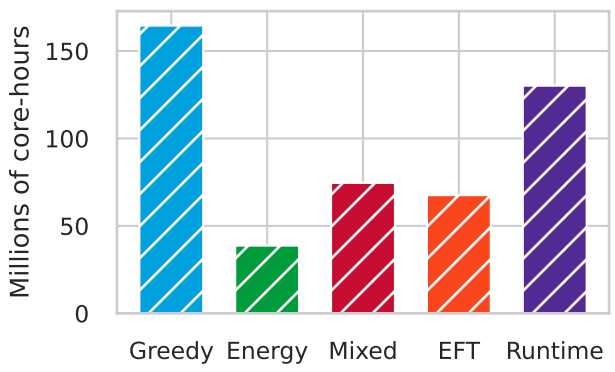


Figure 9. Work completed with a fixed allocation using CBA and with reduced carbon intensity for all machines.

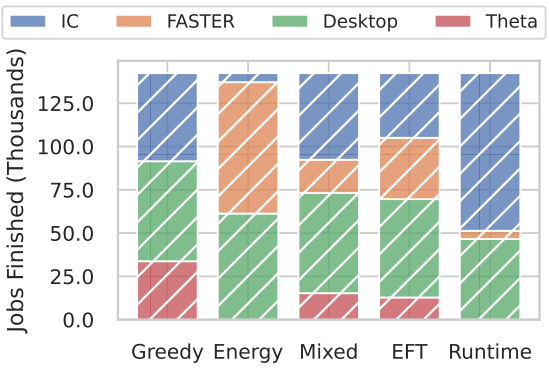


Figure 10. Distribution of functions over machines with reduced carbon intensity for the full workload.

policy allows a user to complete more work in the same allocation. *Greedy* adapts to the new accounting scheme and submits 50% of its workload to IC and only 11% of its workload to FASTER. Although Theta has the lowest embodied carbon rate, its higher energy use still renders it a costly choice. The energy and carbon consumed with CBA are detailed in Table 2. With CBA, we focus on a different metric, but the results are analogous to EBA. **Minimizing cost allows a user to compute the most work while using the least carbon, incentivizing the selection of efficient machines with less embodied carbon.**

4.6 Results with Reduced Carbon Intensity

Carbon intensity can vary considerably across both locales and time—from as low as 16 gCO_2e/kWh in northern Sweden to 600 gCO_2e/kWh in parts of the US and almost 800 gCO_2e/kWh in Poland [17]. We simulate a futuristic scenario involving a low-carbon grid, using Sweden as an example. Mining, manufacturing. and transportation are also transitioning to lower-carbon methods which can reduce the embodied carbon of new devices, but in this scenario, we do not adjust the embodied carbon rate.

We show in in Figure 9 the amount of work performed under each policy with a fixed CBA allocation and in Figure 10 the distribution of jobs across machines. We see in Figure 10 that, relative to previous results, *Greedy* runs a larger proportion of jobs on Theta, the oldest machine with the lowest embodied carbon rate, but never selects FASTER because the energy efficiency improvements of this new hardware are not worth the embodied carbon when electricity generation is very "green". The effects of this trade-off are shown by Table 3, where *Greedy* lowers the carbon consumed but increases the energy consumed compared to the other users. This is in line with previous work arguing low-carbon electricity means more energy must be saved by new hardware to justify upgrading [23]. **In this scenario, CBA incentivizes use of older machines to delay obsolescence, increase refresh times, and reduce embodied carbon in the future.**

Table 3. Energy and carbon used when deploying each policy while computing the full workload when using CBA and with reduced carbon intensity on all machines.

Policy	Energy (MWh)	Carbon ($KgCO_2e$)
Greedy	513	**19**
Energy	**320**	67
Mixed	492	34
EFT	484	42
Runtime	501	21

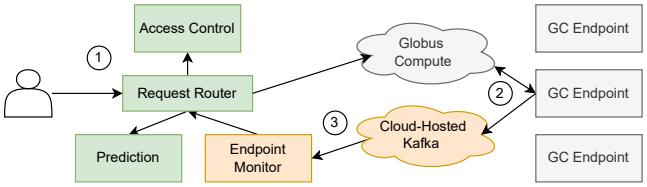


Figure 11. Architecture of the *green-ACCESS* prototype. The three principal system components, shaded green, grey, and orange, are described in the text.

5 Prototype

We implemented *green-ACCESS*, a prototype HPC-FaaS platform with a CBA scheme.

5.1 Design

We built *green-ACCESS* with a FaaS interface that provides a flexible and adaptive runtime, allowing us to efficiently realize impact-based allocations. Recent work shows that FaaS can improve resource utilization and accelerate applications on HPC systems [31–33]. While FaaS simplifies migration of applications between different systems, EBA and CBA can be

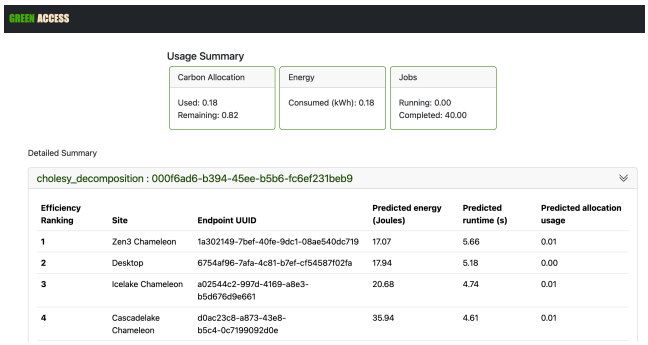


Figure 12. Web interface for *green-ACCESS* prototype.

implemented independently of FaaS. Our system comprises three major components (see Figure 11).

① The frontend supports user interactions. An access control system implements accounting and admission control. Users can employ a Web interface (see Figure 12) or Python interface to access a prediction service that provides estimates of the energy consumption of their jobs. Jobs are submitted via a FaaS interface.

② *green-ACCESS* uses Globus Compute as the FaaS platform to run Python functions on HPC systems [31]. *green-ACCESS* calculates expected CBA charges before forwarding requests to Globus Compute. Registering a machine with *green-ACCESS* requires deploying a Globus Compute Endpoint (GCE) equipped with a monitor that polls data from the RAPL interface, reads hardware counters, and communicates those data back to *green-ACCESS*. This integration with Globus Compute allows *green-ACCESS* to be deployed within (or beside) existing cluster management systems.

③ Energy and performance counter data are transferred via Kafka back to *green-ACCESS*, where they are consumed by the *green-ACCESS* endpoint monitor, a streaming consumer based on the Faust library [34]. The endpoint monitor is responsible for disaggregating per-node power measurements from the RAPL subsystem [21, 35] into user jobs. To this end, we collect, in addition to the energy information, per-process hardware performance counters and periodically fit a power model between performance counters and measured energy [36, 37]. For GPUs, we assume that an entire device is allocated to a single job. The per-process estimates are aggregated to obtain the energy used by a task. These values are used to charge users and refine predictions for future instances of that function. Estimation and accounting are performed asynchronously from job completion.

5.2 Comparison of Pricing Models

Using the prototype, we evaluate the price of executing a function with five different accounting methods:

- **Runtime**: Price is determined only by the core-time used, not accounting for heterogeneity. This is similar to the model used by Chameleon Cloud [19].
- **Energy**: Price is determined only by the energy used, without accounting for device capacity.
- **Peak**: Price is determined by core-time used, multiplied by machine peak performance. This metric accounts for heterogeneous devices by charging more for higher performance systems—a similar model to that used by ACCESS, although the mechanisms to determine the charging factors are not transparent [18].
- **EBA**: The proposed Energy-Based Accounting.
- **CBA**: The proposed Carbon-Based Accounting.

Table 4. Runtime, energy consumption, and costs on different CPU nodes running a Cholesky Decomposition.

Machine	Metrics		Normalized Costs		
	Time (s)	Energy (J)	EBA	CBA	Perf.
Desktop	5.20	18.3	**1.0**	**1.0**	1.43
Cascade Lake	4.68	35.8	1.90	1.20	**1.0**
Ice Lake	**4.60**	19.8	1.10	1.10	1.06
Zen3	5.65	**16.8**	1.05	1.15	1.36

The prototype platform uses CBA, and we calculate EBA and the other accounting methods post-facto for comparison.

We first execute a Cholesky decomposition of a 0.5 GB single precision matrix on CPU nodes. We select Chameleon Cloud nodes [19] resembling the systems of Table 1: A Desktop matching the above specifications, a *Cascade Lake* node matching the specifications of IC, and an *Ice Lake* node with 2 Intel Platinum 8380 CPUs, similar to FASTER. As Chameleon lacks a node similar to Theta, we instead use a *Zen3* node with 2 AMD EPYC 7763 processors with 64 cores each.

The results in Table 4 show the undesirable properties of the Runtime and Peak pricing models. As the Cholesky application runs fastest on the Cascade Lake and Ice Lake systems, charging based solely on **Runtime** assigns the lowest cost on those machines. But those two machines consume the most energy. If usage is weighted by peak performance, as in **Peak**, then as Zen3 and Desktop have the highest peak performance per thread [38] but are the slowest systems for this task, they cost the most. Indirectly, both pricing models incentivize higher energy use.

In contrast, the **EBA** price mirrors task energy usage. The notable discrepancy between **Energy** and **EBA** is that although Zen3 uses the least energy, it has a slightly higher **EBA** cost than Desktop. This difference results from pricing for utilization in Equation 1: Zen3 has a higher TDP per core than Desktop, so it costs more to use per time. **CBA** shares some properties of EBA, with Desktop having the lowest cost and Cascade Lake the highest. However, the newer Zen3 is charged a higher price for embodied carbon. **Overall, we see that running on a more efficient machine, which**

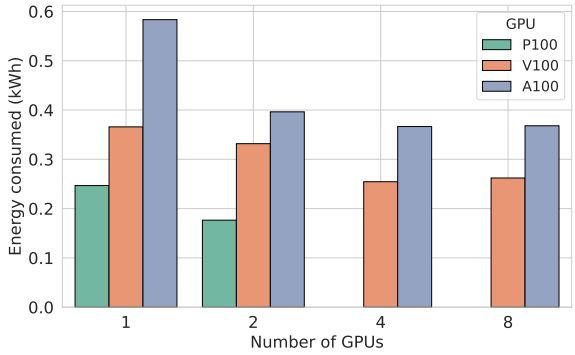


Figure 13. Energy consumed by different GPUs when running a tiled Cholesky Factorization.

would cost more with existing pricing models, costs 47% and 17% less under EBA and CBA, respectively.

The pricing models and trade-offs that we have demonstrated on different CPU machines extend to GPUs. For this experiment, we run a tiled Cholesky decomposition on a 42 GB single precision matrix, using the StarPU runtime system to orchestrate the application across multiple GPUs [39]. Figure 13 shows the energy consumed by different generations and numbers of GPUs. Different compute nodes host up to 8 GPUs. We observe that: 1) Energy consumption initially decreases as the number of GPUs increases and then stabilises when more than four GPUs are used. With four or more GPUs, the application is too small to utilise all GPUs fully, preventing further speedup or energy savings. 2) Recent GPUs consume more energy. The latest, such as the A100, have far more cores, leading to a greater increase in energy consumption than in performance. For the Cholesky application, the critical path and data transfers limit the benefits of additional cores. Indeed, the newest GPU (A100) solves the problem 50% faster than the oldest (P100), but consumes 118% more energy.

Table 6 illustrates how EBA and CBA manage this trade-off compared to other accounting methods. Eight A100 devices provide the best performance, but consume twice the energy of the P100s. On the other hand, EBA and CBA both prioritize using 2 P100 GPUs, which provide the lowest energy consumption and have the lowest embodied carbon rate (shown in Table 5) compared to newer GPUs. Meanwhile, a peak performance pricing model charges the least for using 1 V100 GPU even as it uses twice the energy of 2 P100 GPUs. **EBA and CBA balance the trade-offs between the higher performance of modern GPUs and the lower energy/carbon use of older GPUs.**

5.3 Running a Workload

We next examine the cost and resource consumption to run a workload using the prototype. We experiment with users deploying *Greedy*, *Energy*, and *Runtime* policies to run a workload using *green-ACCESS*. We create a workload of 80 functions from linear algebra (MatMul, Cholesky decomposition [41]), graph processing (Breadth First Search, Minimum Spanning Tree, Graph Pagerank [30]), and scientific computing (Molecular Dynamics Ionization Energy calculation [42], DNA Visualization [30]). All users execute simultaneously, and each task is allocated resources by core. We measure the performance of each function before running the workload to obtain predictions of energy and runtime.

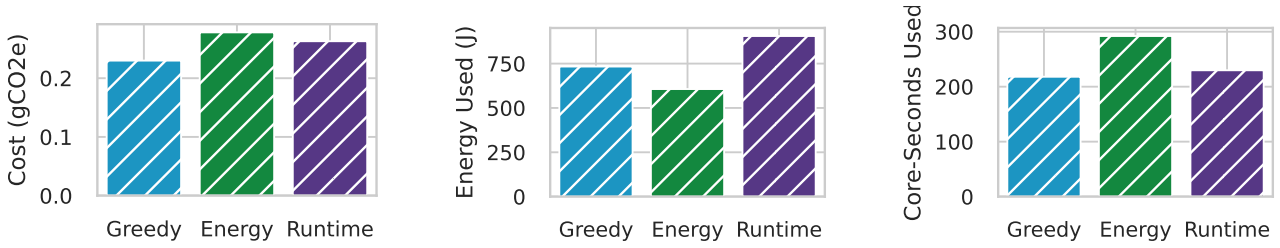
Figure 14 shows the carbon-accounting cost, energy, and core-seconds consumed. Over the whole workload, selecting for minimum price reduces cost by 12% and energy by 19%. We see in Figure 15 that *Greedy* employs three of the four machines for this workload. Here, the Ice Lake node was predicted to be less efficient than the Zen3 node and have a higher embodied carbon cost than the Desktop and Cascade Lake node. **The prototype illustrates that minimizing the price of real jobs under CBA prioritizes energy efficient and lower-carbon machines and validates the trade-offs seen in the simulation.**

Table 5. Specifications of GPU nodes. GFlops is the FLOPS per GPU running GEMM. The average carbon intensity of all nodes was of 53 gCO_2e/kWh. Embodied carbon was calculated according to [40].

GPU	Year	GFlop/s	TDP	Carbon Rate			
# GPUs				1	2	4	8
P100	2018	6700	250	8.5	9.1		
V100	2019	14000	250	19	20	23	28
A100	2021	18000	400	87	93	106	131

Table 6. Comparison of the runtime, energy consumption, and costs under different accounting schemes for the Cholesky Decomposition with different Nvidia GPUs.

	#	Metrics		Normalized Costs		
		Time (s)	Energy (kJ)	EBA	CBA	Perf.
P100	1	2321	889	1.20	1.40	1.55
	2	1396	**635**	**1.0**	**1.0**	1.87
V100	1	1494	1316	1.23	2.07	**1.0**
	2	1190	1194	1.26	1.88	3.23
	4	917	916	1.25	1.44	5.13
	8	926	944	1.85	1.49	10.4
A100	1	1405	2100	1.83	3.35	2.53
	2	926	1427	1.46	2.28	3.33
	4	841	1320	1.76	2.11	6.05
	8	**838**	1325	2.59	2.13	12.0



(a) Cost under CBA **(b)** Energy consumed **(c)** Core-seconds consumed

Figure 14. Resource usage of different users while running the same workload on the prototype system.

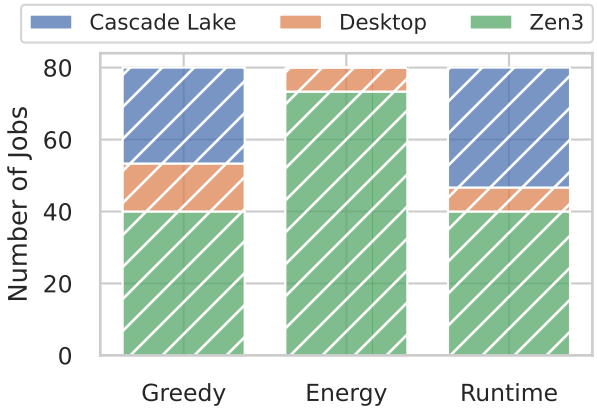


Figure 15. Distribution of tasks to machines on the prototype depending on policy.

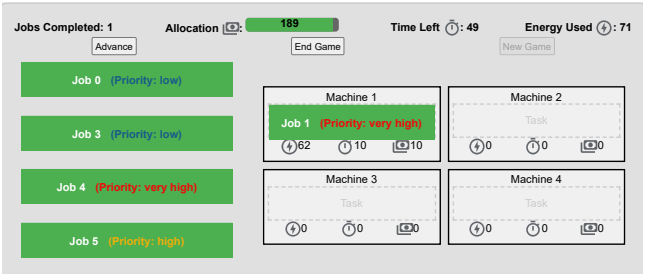


Figure 16. Scheduling game interface. The green boxes are "jobs" that can be "scheduled" by dragging them onto the light-grey "machine" boxes. Job runtime, cost, and energy information were available by hovering over a job.

6 User Study

Finally, we focus on the question of how user behavior may change under a modified pricing scheme. As it was infeasible to acquire the resources for a large-scale deployment of *green-ACCESS*, and changing the allocations of existing systems may have significant user impact, we developed a simulated environment to conduct a user study.

6.1 Design

To conduct the study we built a JavaScript game that emulated the decisions that users of a *green-ACCESS*-like platform would face (see Figure 16). Participants were asked to imagine that they were a computational scientist who had to finish all of their jobs within a time and allocation limit, and they had four machines to use to do so. These instructions are a proxy for realistic HPC use cases where a researcher is granted a limited allocation but faces time pressure to meet a deadline. In our game, a job could be "scheduled" by dragging and dropping it onto a specific machine. Every job was randomly assigned a priority between "Low" and "Very High," although no instructions were given on how to treat jobs of different priorities. This ambiguity was intentional. Users had to individually weigh a job's priority against its time, energy, and cost. To simplify the effect of queues, only one job could be scheduled on a machine at a time. The machines reflected those used in the simulation, and the resources a job used were inferred using the same mechanism as the simulation. More jobs "arrived" as jobs were scheduled to reflect the streaming or time-dependent nature of real workloads. The jobs, and order of jobs was the same for all participants. Each participant was randomly assigned to one of three game versions:

- **V1**: Job cost is proportional to runtime, and no information on job energy consumption is shown. This reflects current standard practice: see Section 2.2.
- **V2**: Cost is still based on runtime, but job energy consumption is displayed next to time and cost.
- **V3**: Cost is given based on the EBA formula.

Participants played the game at least twice, with the first iteration, intended for familiarization with the game and interface, not used for analysis. The version remained the same between the first and second play of the game, but was randomized after that. The game was distributed using the same channels as the survey.

6.2 Results

After discarding every users first time playing the game, we received 207 instances of the game played by 90 unique users.

We discarded 15 instances in which participants finished the game in less than one minute with a disproportionately high amount of allocation remaining.

We first assess the impact of displaying energy information (V2) and using EBA (V3) on total energy consumption: see Figure 17. On average, participants using EBA (V3) consumed 1928 kWh in the simulation, while those with V1 or V2 consumed 3262 kWh and 3142 kWh, respectively. V3 is significantly lower than V1 or V2 (p=0.00), but there was no significant difference between V1 (the control) and V2 (with energy information). In line with survey results, **information regarding energy consumption alone created no change in how much energy a participant consumed.**

However, in V3 (using EBA), participants also completed fewer jobs overall. Users with the changed allocation scheme completed an average of 9.7 jobs compared to 14.5 in V1 and 14.9 in V2. Although we attempted to give an equivalent sized allocation to users in V3 as V1 and V2, the differing pricing scheme meant an exact conversion was impossible. To distinguish the effect of EBA from the effects of a smaller allocation, in Figure 18 we examine the amount of energy used, stratified by the number of jobs completed. The figure shows for any number of jobs completed, **users who saw an EBA version of the game used significantly less energy**.

There were two ways a user could save energy: by choosing not to run a job, or by using a more efficient machine to run a job. To distinguish these two mechanisms, we looked at the probability that a user selected to run a job. Since users may have ran out of time or allocation before seeing all the jobs, we calculate this probability as (# of users who completed job *i* / # of users who saw job *i*). If participants reduced energy by not running the most energy intensive jobs, we would expect a (negative) correlation between the probability of a job being run and its energy consumption. We plot this correlation in Figure 19. While V3 participants were less likely to run a job in general (since they completed fewer jobs on average), there is no correlation between job energy consumption and the probability that a user ran a job, in any game version. Thus, **the decision to run a job was not influenced by the energy consumed, even though job cost depended on the energy.**

The other mechanism by which users could consume less energy while completing the same jobs was to employ more efficient resources. In Figure 20 we illustrate these decisions by plotting the average energy used to run each job across different versions. Note that for each job and each participant, there are only four possible values for the energy consumed by a job, based on the four different machines. A lower average energy consumption means more participants chose more efficient machines. For 16 of the 20 jobs, the average energy consumed across participants in V3 of the game was the lowest out of the three versions. This suggests that **under EBA, when participants elected to run a job, they selected a more efficient machine to do so.**

7 Related Work

Sustainable data-centers: Prior work examined adapting data-center capacity or demand to reduce power, cost, and carbon emissions [9, 43, 44], for example by scheduling jobs to data-centers currently powered by renewable energy. The Zero-Carbon Cloud project further examines running data-centers on "stranded" power — excess power generated when grid supply exceeds load [11]. Such scheduling requires users to provide jobs that can be delayed or moved.

Runtimes: Numerous works on energy-efficient computing focused on adapting dynamic voltage and frequency scaling (DVFS) to high performance environments [45, 46]. DVFS allows software to lower CPU frequency to reduce power consumption at the cost of a lower processing speed. These works attempt to improve efficiency or meet a power budget while minimizing performance impacts [6, 47]. However, if users are not charged for energy consumption, they have limited incentive to adopt these techniques.

Green serverless: The problem of emissions has been noted by the FaaS community [48]. GreenCourier implements carbon-aware scheduling of serverless functions based on cluster location [49], and GreenWhisk [50] extends load balancing with the grid's carbon intensity and energy status of off-grid executors. In contrast, we exploit both heterogeneity and geographic distribution, account for embodied carbon, and engage users into scheduling decisions. Lin and Mohammed similarly propose carbon aware pricing for serverless, but consider optimizations to function configuration rather than heterogeneity between systems [51]. Concurrently with this work, EcoLife [52] uses multi-generation hardware to reduce the carbon footprint of serverless platforms. While EcoLife focuses on optimizing scheduling and keep-alive time, our work focuses on user behavior.

Motivating sustainable behavior: Little work has studied incentives for adopting the many techniques for developing energy-efficient software. Di Pietrantonio et al. suggest that the Thermal Design Power (TDP) of a device can be used to calculate a static cost-ratio between GPU and CPU nodes [53]. Slurm provides energy accounting capabilities [22], but these have not been incorporated into computing allocations. Georgiou et al. suggest incorporating energy into the cluster scheduling algorithm to prioritize energy-efficient users [54]. Other works have proposed incorporating the price of electricity into the cost of cloud VMs [55–58]. These works emphasize deferring costs rather than incentivizing sustainability, and do not examine the same trade-offs as here. Guyon et. al. study the effect of choosing lower performance VMs on overall data-center efficiency [59]. Margery et al. propose attributing CO_2 emissions to VMs [60], but lack details, and do not account for emissions other than electricity generation.

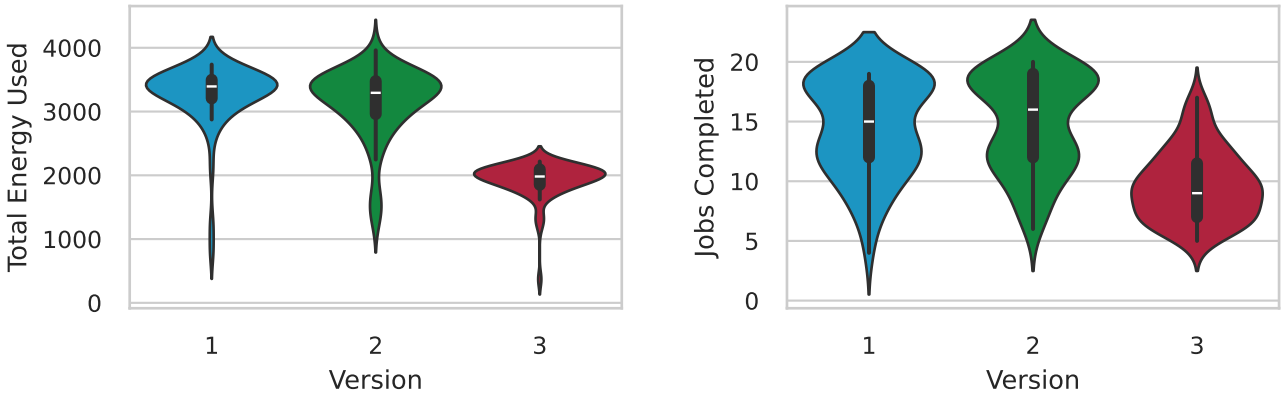


Figure 17. Energy used and jobs completed with different versions of the game. In the EBA version of the game (V3) participants used less energy but also completed fewer jobs.

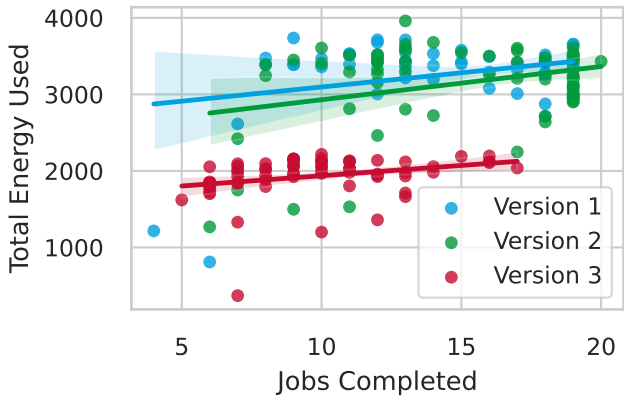


Figure 18. Total energy used for each version of the game, broken down by number of jobs completed. Those playing V3 used less energy to complete the same number of jobs.

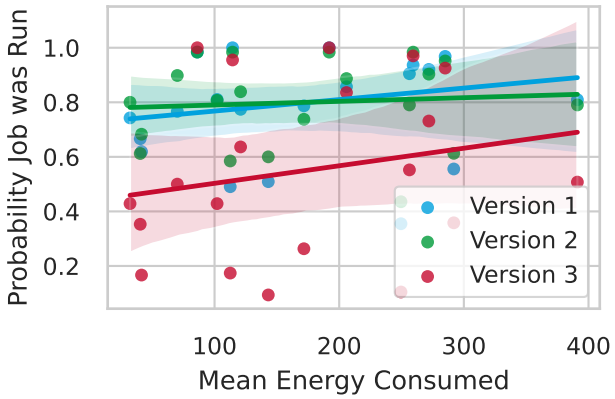


Figure 19. Proportion of users who ran a job vs. average energy consumed by job. Energy use was not correlated with probability of running a job in any version of the game.

Figure 20. Average energy per job by game version. For 16 of the 20 jobs, participants using EBA (V3) achieved the lowest average energy consumed.

8 Summary

Computing is a significant source of energy demand and greenhouse gas emissions. The responsibility for sustainable computing is shared by both producers and consumers. Yet more than 70% of users remain unaware of energy use and energy efficiency is among users' lowest priorities. With the goal of realizing this shared responsibility, we introduced Energy- and Carbon-Based Accounting (EBA and CBA), two mechanisms for charging for computing based on environmental impact. To evaluate these mechanisms, first, we used simulations to show that impact-based pricing can allow an energy-conscious user to process 28% more of a workload that a performance-focused user with the same allocation. Next, we build a prototype FaaS-HPC platform to implement CBA and demonstrate that CBA incentivizes more sustainable decisions which are disincentivized under current accounting models. Finally, we demonstrate that users in a simulated environment use less energy under EBA. By linking costs to energy use and carbon emissions, EBA and CBA incentivize users to prioritize sustainable computing.

References

[1] AWS. (2023, October) Sustainability Pillar - AWS Well-Architected Framework. [Online]. Available: https://docs.aws.amazon.com/wellarchitected/latest/sustainability-pillar/sustainability-pillar.html

[2] "2022 Amazon Sustainability Report," 2022, https://sustainability.aboutamazon.com/2022-sustainability-report.pdf.

[3] "Google Environmental Report 2023," 2023, https://www.gstatic.com/gumdrop/sustainability/google-2023-environmental-report.pdf.

[4] "LUMI: Sustainable future," 2024. [Online]. Available: https://www.lumi-supercomputer.eu/sustainable-future/

[5] J. Eastep, S. Sylvester, C. Cantalupo, B. Geltz, F. Ardanaz, A. Al-Rawi, K. Livingston, F. Keceli, M. Maiterth, and S. Jana, "Global extensible open power manager: A vehicle for HPC community collaboration on co-designed energy management solutions," in *High Performance Computing*. Cham: Springer International Publishing, 2017, pp. 394–412.

[6] D. C. Wilson, S. Jana, A. Marathe, S. Brink, C. M. Cantalupo, D. R. Guttman, B. Geltz, L. H. Lawson, A. H. Al-rawi, A. Mohammad, F. Keceli, F. Ardanaz, J. M. Eastep, and A. K. Coskun, "Introducing application awareness into a unified power management stack," in *IEEE International Parallel and Distributed Processing Symposium*. Virtual: IEEE, 2021, pp. 320–329.

[7] J. You, J.-W. Chung, and M. Chowdhury, "Zeus: Understanding and optimizing GPU energy consumption of DNN training," in *USENIX NSDI*, 2023.

[8] J.-W. Chung, Y. Gu, I. Jang, L. Meng, N. Bansal, and M. Chowdhury, "Perseus: Reducing energy bloat in large model training," 2024. [Online]. Available: https://arxiv.org/abs/2312.06902

[9] T. Sukprasert, A. Souza, N. Bashir, D. Irwin, and P. Shenoy, "Quantifying the benefits of carbon-aware temporal and spatial workload shifting in the cloud," 2023.

[10] P. Wiesner, I. Behnke, D. Scheinert, K. Gontarska, and L. Thamsen, "Let's wait awhile: How temporal workload shifting can reduce carbon emissions in the cloud," in *22nd International Middleware Conference*, ser. Middleware '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 260–272. [Online]. Available: https://doi.org/10.1145/3464298.3493399

[11] F. Yang and A. A. Chien, "ZCCloud: Exploring wasted green power for high-performance computing," in *IEEE International Parallel and Distributed Processing Symposium*, 2016, pp. 1051–1060.

[12] A. Radovanovic. (2020, April) Our data centers now work harder when the sun shines and wind blows. [Online]. Available: https://blog.google/inside-google/infrastructure/data-centers-work-harder-sun-shines-wind-blows/

[13] L. Lin, R. Wijayawardana, V. Rao, H. Nguyen, W. E. Gnibga, and A. A. Chien, "Exploding AI power use: An opportunity to rethink grid planning and management," 2024. [Online]. Available: https://arxiv.org/abs/2311.11645

[14] W.-c. Feng and K. Cameron, "The Green500 list: Encouraging sustainable supercomputing," *Computer*, vol. 40, no. 12, pp. 50–55, 2007.

[15] T. Hoefler, "The Green Graph500," Jul. 2012.

[16] K.-D. Lange and M. G. Tricker, "The design and development of the server efficiency rating tool (SERT)," in *2nd ACM/SPEC International Conference on Performance Engineering*, ser. ICPE '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 145–150. [Online]. Available: https://doi.org/10.1145/1958746.1958769

[17] "Electricity maps," 2022, https://app.electricitymaps.com/map.

[18] T. J. Boerner, S. Deems, T. R. Furlani, S. L. Knuth, and J. Towns, "ACCESS: Advancing innovation: NSF's advanced cyberinfrastructure coordination ecosystem: Services & support," in *Practice and Experience in Advanced Research Computing*, ser. PEARC '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 173–176. [Online]. Available: https://doi.org/10.1145/3569951.3597559

[19] K. Keahey, J. Anderson, Z. Zhen, P. Riteau, P. Ruth, D. Stanzione, M. Cevik, J. Colleran, H. S. Gunawi, C. Hammock, J. Mambretti, A. Barnes, F. Halbach, A. Rocha, and J. Stubbs, "Lessons learned from the Chameleon testbed," in *USENIX Annual Technical Conference*. USENIX Association, July 2020.

[20] M. Tirmazi, A. Barker, N. Deng, M. E. Haque, Z. G. Qin, S. Hand, M. Harchol-Balter, and J. Wilkes, "Borg: the next generation," in *Proceedings of the Fifteenth European Conference on Computer Systems*, ser. EuroSys '20. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3342195.3387517

[21] K. N. Khan, M. Hirki, T. Niemi, J. K. Nurminen, and Z. Ou, "RAPL in action: Experiences in using RAPL for power measurements," *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, vol. 3, no. 2, mar 2018. [Online]. Available: https://doi.org/10.1145/3177754

[22] D. Auble, T. Cadeau, Y. Georgiou, and M. Perry, "SLURM energy accounting and external sensors plug-in," 2013, https://slurm.schedmd.com/SUG13/energy_sensors.pdf.

[23] B. Li, R. Basu Roy, D. Wang, S. Samsi, V. Gadepally, and D. Tiwari, "Toward sustainable HPC: Carbon footprint estimation and environmental implications of HPC systems," in *International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '23. New York, NY, USA: Association for Computing Machinery, 2023. [Online]. Available: https://doi.org/10.1145/3581784.3607035

[24] "Software carbon intensity (SCI) specification," 2024. [Online]. Available: https://sci.greensoftware.foundation/

[25] J. Lyu, J. Wang, K. Frost, C. Zhang, C. Irvene, E. Choukse, R. Fonseca, R. Bianchini, F. Kazhamiaka, and D. S. Berger, "Myths and misconceptions around reducing carbon embedded in cloud platforms," in *2nd Workshop on Sustainable Computer Systems*, ser. HotCarbon '23. New York, NY, USA: Association for Computing Machinery, 2023. [Online]. Available: https://doi.org/10.1145/3604930.3605717

[26] *How To Depreciate Property*, 2023, https://www.irs.gov/pub/irs-pdf/p946.pdf.

[27] M. Gonthier, E. Larsson, L. Marchal, C. Nettelblad, and S. Thibault, "Data-driven locality-aware batch scheduling," in *26th Workshop on Advances in Parallel and Distributed Computational Models*, San Francisco, United States, May 2024. [Online]. Available: https://inria.hal.science/hal-04500281

[28] T. Patel, A. Wagenhäuser, C. Eibel, T. Hönig, T. Zeiser, and D. Tiwari, "What does power consumption behavior of HPC jobs reveal?: Demystifying, quantifying, and predicting power consumption characteristics," in *IEEE International Parallel and Distributed Processing Symposium*, 2020, pp. 799–809.

[29] T.-P. Pham, J. J. Durillo, and T. Fahringer, "Predicting workflow task execution time in the cloud using a two-stage machine learning approach," *IEEE Transactions on Cloud Computing*, vol. 8, no. 1, pp. 256–268, 2020.

[30] M. Copik, G. Kwasniewski, M. Besta, M. Podstawski, and T. Hoefler, "SeBS: A serverless benchmark suite for function-as-a-service computing," in *22nd International Middleware Conference*, ser. Middleware '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 64–78. [Online]. Available: https://doi.org/10.1145/3464298.3476133

[31] R. Chard, Y. Babuji, Z. Li, T. Skluzacek, A. Woodard, B. Blaiszik, I. Foster, and K. Chard, "FuncX: A federated function serving fabric for science," in *29th International Symposium on High-Performance Parallel and Distributed Computing*, ser. HPDC '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 65–76. [Online]. Available: https://doi.org/10.1145/3369583.3392683

[32] M. Copik, K. Taranov, A. Calotoiu, and T. Hoefler, "rFaaS: Enabling high performance serverless with RDMA and leases," in *IEEE International Parallel and Distributed Processing Symposium*, 2023, pp. 897–907.

[33] A. Kamatar, V. Hayot-Sasson, Y. Babuji, A. Bauer, G. Rattihalli, N. Hogade, D. Milojicic, K. Chard, and I. Foster, "Greenfaas: Maximizing energy efficiency of hpc workloads with faas," *arXiv preprint arXiv:2406.17710*, 2024.

[34] Solem, Ask and Goel, Vineet, *Faust User Manual*, Robinhood Markets, 2019, https://faust.readthedocs.io/en/latest/.

[35] H. David, E. Gorbatov, U. R. Hanebutte, R. Khanna, and C. Le, "RAPL: Memory power estimation and capping," in *16th ACM/IEEE International Symposium on Low Power Electronics and Design*, 2010, pp. 189–194.

[36] N. Schmitt, L. Iffländer, A. Bauer, and S. Kounev, "Online power consumption estimation for functions in cloud applications," in *IEEE International Conference on Autonomic Computing*. Umea, Sweden: IEEE, 2019, pp. 63–72.

[37] G. Fieni, R. Rouvoy, and L. Seinturier, "SmartWatts: Self-calibrating software-defined power meter for containers," in *20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing*. IEEE, 2020, pp. 479–488.

[38] PassMark, "CPU benchmarks," Surry Hills, NSW, Austrailia, 2024. [Online]. Available: https://www.cpubenchmark.net/

[39] C. Augonnet, S. Thibault, R. Namyst, and P.-A. Wacrenier, "StarPU: A Unified Platform for Task Scheduling on Heterogeneous Multicore Architectures," *Concurrency and Computation: Practice and Experience, Special Issue: Euro-Par 2009*, vol. 23, 2011.

[40] S. Ji, Z. Yang, S. Cahoon, A. K. Jones, and P. Zhou, "SCARIF: Towards carbon modeling of cloud servers with accelerators," in *IEEE Computer Society Annual Symposium on VLSI*, 2024, pp. 1–6.

[41] C. R. Harris, K. J. Millman, S. J. Van Der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.

[42] L. Ward, "ML-in-the-loop molecular design with Parsl," https://github.com/ExaWorks/molecular-design-parsl-demo/tree/main, 2021, accessed: 2024-01-24.

[43] W. Deng, F. Liu, H. Jin, B. Li, and D. Li, "Harnessing renewable energy in cloud datacenters: Opportunities and challenges," *IEEE Network*, vol. 28, no. 1, pp. 48–55, 2014.

[44] Z. Liu, M. Lin, A. Wierman, S. Low, and L. L. H. Andrew, "Greening geographical load balancing," *IEEE/ACM Transactions on Networking*, vol. 23, no. 2, pp. 657–671, 2015.

[45] C.-h. Hsu and W.-c. Feng, "A power-aware run-time system for high-performance computing," in *ACM/IEEE Conference on Supercomputing*. IEEE, 2005, pp. 1–1.

[46] F. Harada, T. Ushio, and Y. Nakamoto, "Power-aware resource allocation with fair QoS guarantee," in *12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*. IEEE, 2006, pp. 287–293.

[47] L. Wang, G. Von Laszewski, J. Dayal, and F. Wang, "Towards energy aware scheduling for precedence constrained parallel tasks in a cluster with DVFS," in *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. Heraklion, Crete, Greece: IEEE, 2010, pp. 368–377.

[48] P. Patros, J. Spillner, A. V. Papadopoulos, B. Varghese, O. Rana, and S. Dustdar, "Toward sustainable serverless computing," *IEEE Internet Computing*, vol. 25, no. 6, pp. 42–50, 2021.

[49] M. Chadha, T. Subramanian, E. Arima, M. Gerndt, M. Schulz, and O. Abboud, "GreenCourier: Carbon-aware scheduling for serverless functions," in *9th International Workshop on Serverless Computing*, 2023, pp. 18–23.

[50] J. Serenari, S. Sreekumar, K. Zhao, S. Sarkar, and S. Lee, "GreenWhisk: Emission-aware computing for serverless platform," *arXiv preprint arXiv:2409.03029*, 2024.

[51] C. Lin and M. Shahrad, "Bridging the sustainability gap in serverless through observability and carbon-aware pricing," *HotCarbon'24*, 2024.

[52] Y. Jiang, R. B. Roy, B. Li, and D. Tiwari, "EcoLife: Carbon-aware serverless function scheduling for sustainable computing," 2024. [Online]. Available: https://arxiv.org/abs/2409.02085

[53] C. Di Pietrantonio, C. Harris, and M. Cytowski, "Energy-based accounting model for heterogeneous supercomputers," *arXiv preprint arXiv:2110.09987*, 2021.

[54] Y. Georgiou, D. Glesser, K. Rzadca, and D. Trystram, "A scheduler-level incentive mechanism for energy efficiency in HPC," in *15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 2015, pp. 617–626.

[55] M. Hinz, G. P. Koslovski, C. C. Miers, L. L. Pilla, and M. A. Pillon, "A cost model for IaaS clouds based on virtual machine energy consumption," *Journal of Grid Computing*, vol. 16, pp. 493–512, 2018.

[56] M. Kurpicz, A.-C. Orgerie, and A. Sobe, "How much does a VM cost? Energy-proportional accounting in VM-based environments," in *24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, 2016, pp. 651–658.

[57] M. Aldossary and K. Djemame, "Energy-based cost model of virtual machines in a cloud environment," in *5th International Symposium on Innovation in Information and Communication Technology*, 2018, pp. 1–8.

[58] A. Narayan and S. Rao, "Power-aware cloud metering," *IEEE Transactions on Services Computing*, vol. 7, no. 3, pp. 440–451, 2014.

[59] D. Guyon, A.-C. Orgerie, C. Morin, and D. Agarwal, "Involving users in energy conservation: A case study in scientific clouds," *International Journal of Grid and Utility Computing*, vol. 10, no. 3, pp. 272–282, 2019.

[60] D. Margery, D. Guyon, A.-C. Orgerie, C. Morin, G. Francis, C. Palansuriya, and K. Kavoussanakis, "A CO2 emissions accounting framework with market-based incentives for cloud infrastructures," in *6th International Conference on Smart Cities and Green ICT Systems*, ser. SMARTGREENS 2017. Setubal, PRT: SCITEPRESS - Science and Technology Publications, Lda, 2017, p. 299–304. [Online]. Available: https://doi.org/10.5220/0006356502990304

Received 22 October 2024