Attributing Embodied Carbon in Computing

Leo Han
lxh4@cornell.edu
Cornell Tech
New York, New York, USA

Udit Gupta
ugupta@cornell.edu
Cornell Tech
New York, New York, USA

ABSTRACT

The carbon footprint of computing is xxx percent of global total CO2 emissions per year, on par with that of the aviation industry []. The footprint of computing is only expected to double over the next decade, making it ever more important for computing academics and industry professionals to devote effort towards understanding computing's sustainability impact and taking measures to mitigate such impact []. Existing work has looked ways of reducing the carbon footprint of devices from a hardware and system design perspective []. However, from the user and application side, carbon attribution efforts have only focused on operational carbon emissions []. Attribution of embodied carbon footprint allows software developers and users of compute services to optimize applications to reduce the embodied carbon footprint of computing hardware from the demand side. The few existing methodologies available for attributing embodied carbon ignore the effects of temporal demand and also fail to attribute most of the embodied carbon.

CCS CONCEPTS

• **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

ACM Reference Format:
Leo Han and Udit Gupta. 2024. Attributing Embodied Carbon in Computing. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Computing has a large carbon footprint. The carbon footprint of computing can be divided into operational and embodied emissions. Operational emissions result from the power consumed using the computer hardware. Embodied emissions are the emissions that result from the manufacturing and transport of the hardware; it is all the emissions that were entailed to get the computer hardware to its final usable state.

Although there is much research on improving power-efficiency, and thus operational emissions, of computing hardware, only recently have researchers and industry professionals began to look at embodied emissions for computing. Embodied emissions compose a significant portion, and sometimes a majority, of total emissions of computing []. Moreover, due to the rising demand on computing, computing's carbon footprint is expected to double within the next decade []. Moreover, with continued improvements in power efficiency and the increase in carbon emissions per unit area for newer process nodes, embodied emissions are expected to take a larger and larger portion of computing's total carbon footprint. Due to the significant impact of embodied carbon emissions in computing, the computing community needs to take timely and broad action to mitigate its impact and growth.

Existing work on the embodied carbon has focused on the hardware aspect and focused on how can hardware architectures be designed to entail less embodied carbon []. Despite the users and applications being the demand that drives hardware creation, little work exists that address the problem of how to reduce the embodied carbon footprint of computing hardware from the demand side.

Current attribution methods for embodied carbon fail account for the effects of temporal demand on computing hardware capacity and fail to attribute a majority of total embodied carbon [] (Microsoft, Green Software Foundation, ACT).

This paper proposes a way of thinking about the responsibilities of different stakeholders in the computing environment. This paper also proposes a new model of attributing embodied carbon based on the impact of an application on the effective demand for computing capacity. The model also attributes the totality of embodied carbon emissions to different stakeholders according their respective responsibilities. Throughout the paper, we will use real Azure VM traces as case studies [].

2 FLAWS IN CURRENT EMBODIED CARBON ATTRIBUTION METHODS

Existing methods (Microsoft, ACT, Green Software Foundation) attribute embodied carbon naively by looking at the use of a resource compared to that resource's lifetime, per equation 1[]. Naive attribution models attribute embodied carbon as if all hardware resources are utilized 100% of the time which is far from reality []. The naive attribution model suffers from two key consequences.

$$C_{emb\_app} = \frac{T_{app}}{T_{life}} \times C_{emb} \times Q_{allocated} \quad (1)$$

2.1 Effects of Peak Demand are Ignored

An application that adds demand to the peak demand period will increase the minimum resource capacity needed to accommodate the resource demand whereas an application that adds demand to an off-peak period will not have an effect on the minimum resource capacity needed.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/XXXXXXX.XXXXXXX

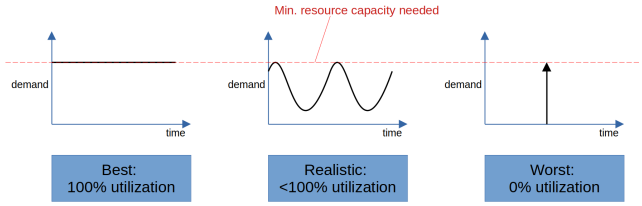

Figure 1: Minimum resource capacity required is determined by the peak demand. In the three scenarios shown above, the minimum resource capacity is the same despite the drastically different levels of total resource utilization.

2.2 Most of the Embodied Carbon is Never Attributed

In the extreme case where there are no users, then no hardware is needed and there is zero embodied carbon. In the actual case where there is a demand for compute capacity from users, the hardware provider provides compute capacity and that results in the embodied carbon emissions from the compute capacity needed. Therefore, the entirety of the embodied carbon costs should be attributed among the users and the cloud service provider.

However, with the simple use-time based method of embodied carbon attribution (1), significant portions of embodied carbon are never attributed because attribution is based on portion of use time over lifetime, as illustrated in the simple example in 2. Therefore, embodied carbon is only fully attributed when resource utilization is 100% over the entire hardware lifetime. In practice, utilization rates are far below 100%; data centers are often less than 50% utilized [].

Since the sum of use time of a resource across all applications is less than the lifetime of the hardware resource2, the sum of embodied carbon attributed across all applications is less than embodied carbon footprint of the hardware resource. If the applications are not responsible for the missing carbon, then who is?

$$\sum T_{app} < T_{life} \tag{2}$$

$$\Rightarrow \sum C_{emb\_app} < C_{emb} \tag{3}$$

3 COARSE-GRAINED CARBON ATTRIBUTION

To help users and compute providers break down the contributors to embodied carbon, these three metrics identify inefficiencies in system-level behavior that lead to increased embodied carbon emissions from computing hardware. These coarse-grained metrics are useful because they are simple to understand while providing insight into identifying bottlenecks in reducing data center embodied carbon emissions. Moreover, they can be calculated with only system-wide coarse-grained telemetry on resource allocation and utilization. No application-level or user-level telemetry is needed.

3.1 Resource Provisioning Efficiency (RPE)

RPE indicates how efficiently a compute infrastructure providers such as a public cloud service provider has provisioned hardware

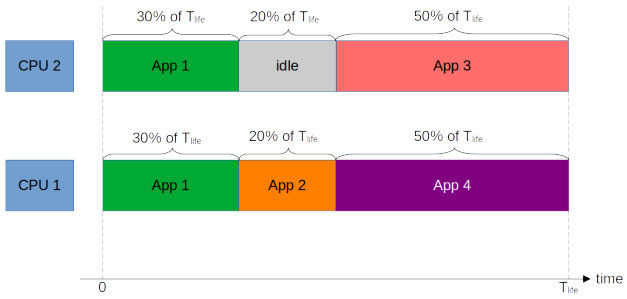

Figure 2: Simple example of two identical CPUs, where CPU 2 is idle 20% of its lifetime.

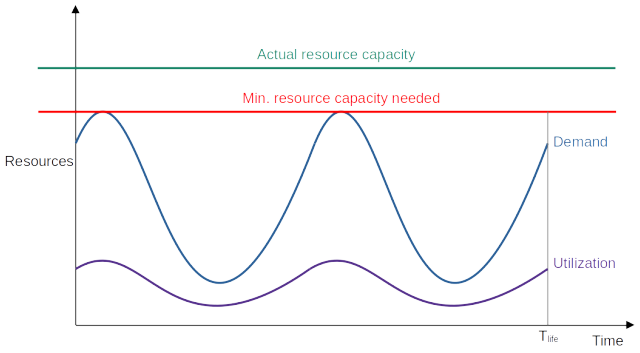

Figure 3: True hardware utilization is often much lower than resource capacity. Resource capacity must be higher than peak resource demand.

for user demand. RPE is the ratio of peak hardware resource demand to total hardware capacity, per 4. For example, a RPE of 80% means that 20% of the hardware capacity did not contribute to meeting user demand. Improving RPE is the responsibility of the compute infrastructure provider; a low RPE indicates that the compute hardware capacity is oversized for user demand and has incurred unnecessary amounts of embodied carbon emissions.

$$RPE = \frac{Q_{min}}{Q_{capacity}} \tag{4}$$

$$Q_{min} = max(Q_{demand}) \tag{5}$$

3.2 Resource Demand Efficiency (RDE)

RDE indicates how efficiently or inefficiently a resource demand curve can be met with the minimum possible hardware capacity. In RDE is the area under the demand curve divided by the area under the hardware capacity line. A resource demand curve with sharper peaks (curve A in **??**) will tend to have worse RDE than more constant demand curves (curve B in **??**).

A computer service provider cannot achieve greater average allocation efficiency than the RDE while fulfilling user demand without oversubscribing resources. RDE being a function of the

resource demand curve means that it is the collective user behavior that creates this source of compute utilization inefficiency that contributes to greater embodied carbon emissions. Users can help improve RDE by scheduling their workloads at less busy times or heaving less stringent time requirements for their workloads, and computer service providers can help reduce RUE by providing users with incentives to use compute resources at low demand times.

$$RDE = \frac{\int_{T_{life}} Q_{demand}(t)\,dt}{Q_{min} \times T_{life}} \tag{6}$$

3.3 Resource Utilization Efficiency (RUE)

RUE is the ratio of hardware utilization to hardware allocation. RUE is determined by individual behavior of users and their applications. Users can help reduce RUE by right-sizing hardware allocations and requests so that allocated hardware resources do not sit idle and are better utilized for effective work.

$$RUE = \frac{\int_{T_{life}} Q_{util}(t)\,dt}{\int_{T_{life}} Q_{demand}(t)\,dt} \tag{7}$$

3.4 End-to-end Embodied Carbon Use Efficiency (ECUE)

The three metrics RPE, RDE, and RUE help quickly clarify what is the bottleneck in reducing data center emissions. If RPE dominates, then issue lies in data center hardware overprovisioning. If RDE dominates, then the issue is that resource demand over time fluctuates greatly. If RUE dominates, then the issue is that individual users are not efficiently using allocated resources.

By also considering performance (application dependent) and the embodied carbon emissions of the entire compute infrastructure, we can use RPE, RDE, and RUE to calculate End-to-end Embodied Carbon Use Efficiency as a measure of how much computational work are we achieving.

$$ECUE = perf \times T_{life} \times RPE \times RDE \times RUE \div C_{emb} \tag{8}$$

4 DEMAND-BASED EMBODIED CARBON ATTRIBUTION

Given data on per-app resource allocations, we can attribute the embodied carbon of data center infrastructure to each user. The proposed demand-based attribution method address the shortfalls of existing embodied carbon attribution methods outlined in section 2 by attributing 100% of all resources to a combination of users and the data center provider.

4.1 Attributions to Data Center Provider

As shown in section 3.1, any hardware capacity above the minimum necessary hardware capacity entails embodied emissions that were unnecessary for meeting user demands. We propose that the entirety of the embodied carbon emissions associated with the hardware capacity above the minimum required hardware capacity be attributed to the data center provider.

$$C_{emb\_DC} = (Q_{capacity} - Q_{min}) \times C_{emb} = (1 - RPE) \times Q_{capacity} \times C_{emb} \tag{9}$$

4.2 Attribution to Users

All of the embodied carbon emissions associated with Q_{min} must be attributed to users as Q_{min} is the minimum hardware capacity needed to accommodate all user demand. Intuitively, the users who demand capacity during peak demand periods contribute more to a higher Q_{min} and thus a worse RDE than those who demand capacity during lower demand periods. We amortize the embodied carbon of each unit of hardware over its total allocated time rather than its lifetime.

With the simple two-CPU example shown in figure 2, CPU 2 is idle for 20% of its lifetime. Therefore, it's embodied carbon cost must be fully amortized over 80% of lifetime. Therefore, App 1 will be attributed 30%÷80% = 37.5% of CPU 2's total embodied carbon cost, and App 3 will be attributed 50%÷80% = 62.5% of the embodied carbon cost of CPU 2. CPU 1 is used over its entire lifetime so its embodied carbon cost in amortized over its entire lifetime. Doing similar attribution to apps for CPU 1, we have the following per-app embodied carbon attributions where C_emb is the embodied emissions for one CPU.

- $C_{App1} = 0.675\ C_{emb}$
- $C_{App2} = 0.2\ C_{emb}$
- $C_{App3} = 0.625\ C_{emb}$
- $C_{App4} = 0.5\ C_{emb}$

However, since App 3 and App 4 are using the same amount of resources at the same time, they should be attributed the amount of embodied carbon. Thus, in such situations, we average attribution rates across resources and end up with the following attributions.

- $C_{App1} = 0.675\ C_{emb}$
- $C_{App2} = 0.2\ C_{emb}$
- $C_{App3} = 0.5625\ C_{emb}$
- $C_{App4} = 0.5625\ C_{emb}$

To extend the attribution process we used for the simple two-CPU example to much greater numbers of resources and users, we formalize the process as follows.

Since we attribute carbon to each application by the the quantity of hardware resources neeed along with the time demanded of each of the resources, we can use the notion of carbon intensity, in gCO_2 per unit time per unit hardware, as the rate at which we attribute embodied carbon to applications.

Like we did with CPU 2 in the two-CPU example, we must adjust the attribution carbon intensity such that the embodied carbon is only amortized over time that hardware is used. For the sake of analysis, we can assume, without the loss of generality, that the q-th resource is only used if the resource demand Q_{demand} reaches q. E.g., the 5th CPU core is only allocated if 5 or more CPU cores are demanded. Then for the q-th resource, its total allocated time over its lifetime can be found by sum the time intervals in which $Q_{demand} \geq q$.

$$T_{alloc} = \int_0^{T_{life}} u(Q_{demand}(t) - q)\,dt \tag{10}$$

where u is the unit step function and

$$u(Q_{demand}(t) - q) = \begin{cases} 1 & Q_{demand} \geq q \\ 0 & Q_{demand} < q \end{cases} \quad (11)$$

Then for the q-th resource, its carbon intensity is as follows.

$$ci(q) = \frac{C_{emb}}{T_{alloc}} \quad (12)$$

Similar to our 2-CPU example, we must also ensure that the all users within the same time slice get attributed carbon with the same carbon intensity. So to get carbon intensity as a function of time, we average across the $ci(q)$ for each time slice.

$$ci(t) = \frac{\int_0^{Q_{demand}(t)} ci(q)\, dq}{Q_{demand}(t)} \quad (13)$$

In practice, we want to attribute embodied carbon before the entire hardware lifetime is up. We can simply replace T_{life} in the above equations with any period over which we can perform the same attribution method.

5 ACCOUNTING FOR SECOND-LIFE AND OVERSUBSCRIPTION

Placeholder - add section if needed

5.0.1 *RPE with Oversubscription.*

$$Q_{min} = max(Q_{use}) \quad (14)$$

6 CASE STUDIES: PUBLIC CLOUD AND HPC

6.1 Microsoft Azure Public Cloud

From the allocation curve as shown in figure 4, RDE is calculated to be 0.952 for CPU and 0.960 for memory.

From Barbalho+ MLSys'23[], we see that Microsoft has been able to reach around 85% packing efficiency, which means RPE should be around $85\% \div RDE = 0.893$ for CPU and 0.885 for memory. RUE on the other hand, is much lower, at only 0.122.

Under naive attribution, the portion of total embodied emissions of a resource type attributed to users is equal to RPE × RDE (which is equal to the packing efficiency) which is around 85%. However, as illustrated in section 2.1, the entirety of embodied carbon for the minimum required resource capacity should be attributed to the users with the remaining attributed to the data center provider.

Assuming that the Azure data center runs {x type of server} with an embodied carbon cost for its CPU of Y, we can create the carbon intensity curve for embodied carbon attribution based on the allocation curve.

6.2 Marconi 100 HPC

The Marconi 100 HPC has an extensive set of workload traces available including resource allocations and utilizations. We also know the Marconi 100 HPC hardware capacity:

From the traces, we calculate that RPE = x%, RDE = x%, and RUE = x%.

From the resource demand curves and using LCA data for the hardware used in the Marconi 100, we can create carbon intensity functions for CPU, GPU, and memory which can be used to appropriately attribute embodied carbon per workload.

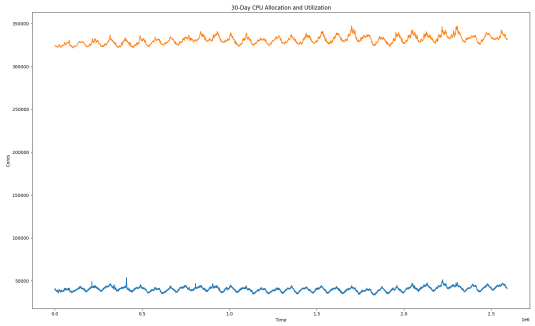

Figure 4: Azure CPU allocation and utilization from Azure 2017 VM traces

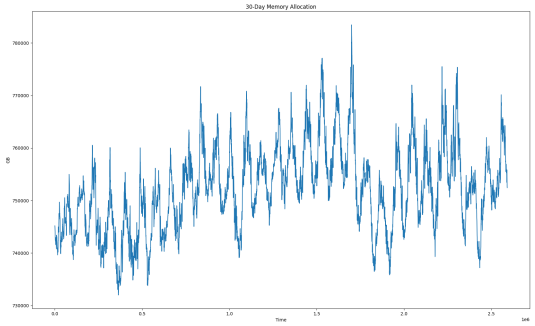

Figure 5: Azure memory allocation from Azure 2017 VM traces

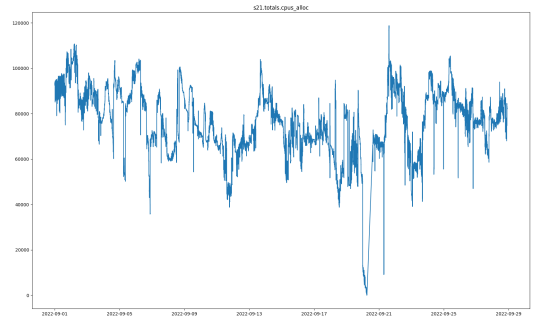

Figure 6: Marconi 100 HPC CPU Allocations in September 2022

7 RELATED WORK

Scheduling and Resource Management: Resource Central, Protean, Pond, Barbalho+ MLSys'23

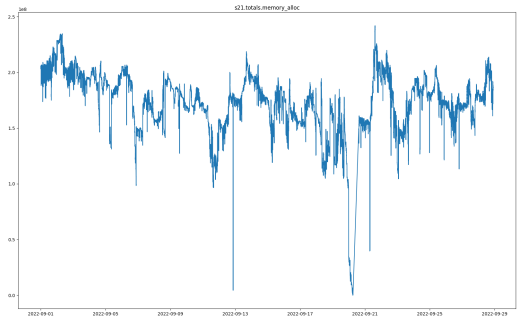

Figure 7: Marconi 100 HPC Memory Allocations in September 2022

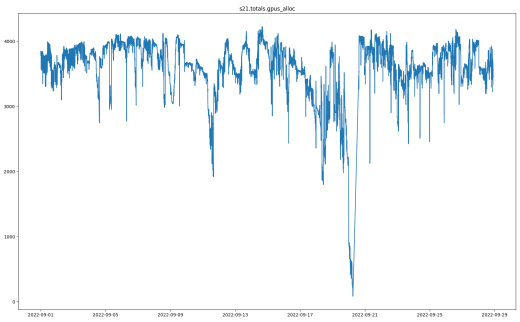

Figure 8: Marconi 100 HPC GPU Allocations in September 2022

Oversubscription: power based oversubscription, Sheng+ WWW'23, Kumbhare+ ATC'21

Carbon-aware workload scheduling: CarbonExplorer, Andrew Chien's work, Google

Sustainable software: GSF, CodeCarbon, Green Metrics Tool, SoftAWERE

Carbon attribution: Google, MSF dashboard

Cloud sustainability: Cloud Jewels,

8 DISCUSSION

We see different problems with public clouds and HPC. Public clouds achieve good RPE (packing efficiency) and RDE, possibly due to sophisticated packing and scheduling systems along with SpotVMs incentivizing users to use spare compute capacity, effectively doing demand-leveling.

As seen with Microsoft Azure, RDE is high at around 95% and RPE (as proxied by packing density) is also high at 85% and on an upward trend as a result of active efforts to improve packing. Fortunately, with public cloud providers like Microsoft Azure, improving RDE and RPE also aligns with business goals; as evidenced by work like Spot VMs[], Harvest VMs [], and advanced schedulers [], significant effort is being put towards improving allocation efficiency. Unfortunately, users are charged by resource allocation not utilization, so cloud providers have little incentive to improve RUE. Users, on the hand, have an economic incentive to demand less resources; however, uncertain loads and compute needs motivate users to oversize resource requests conservatively.

The demand-based embodied carbon attribution method Oversubscription can help with improving overall resource utilization efficiency. Example, Google Borg from 2010 to 2019.

9 FUTURE WORK

Using demand-based attribution model to inform actionable things on the end of cloud providers and users.

Consideration of lifetime extension. Actual lifetime of hardware often differs front the realized lifetime of hardware. Hardware, either in consumer or data center settings, are often prematurely retired. Moreover, in our modeling, we have considered hardware lifetime to be static regardless of utilization. However, high utilization can decrease hardware lifetime, and vice versa.

SLA considerations necessitate data center utilization to be low. Unfortunately, this means that latency-sensitive applications have an out-sized impact on emissions.

Heterogeneity of platforms within a provider.

10 RIGHTS INFORMATION

ACKNOWLEDGMENTS

REFERENCES

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009