Cost and Carbon Reduction for Microsoft Azure Virtual Machines Using Workload Analysis

1st Daisy Wong
Richard Montgomery High School
United States
w6daisy@gmail.com

2nd Oliver Zhang
Strake Jesuit College Preparatory
United States
oliver.zhang0712@gmail.com

3rd Jacky Huang
St. Anne's-Belfield School
United States
zzandkun@gmail.com

***Abstract*—Cloud computing has rapidly become the dominant platform for businesses across various sectors. However, many companies find it challenging to effectively control costs, often resulting in suboptimal resource allocation and unnecessary overspending. Moreover, the expansion of cloud computing has spurred a surge in electricity consumption, causing a corresponding rise in greenhouse gas emissions. This paper aims to reduce both the cost associated with Virtual Machines (VMs) in the cloud and the carbon footprint generated by cloud computing activities. To tackle this issue, we analyze the 2019 Azure cloud trace, which incorporates data from more than 2.6 million VMs and historical records of grid emissions intensity from the California ISO Northern Region. We also devise a machine learning model to predict costs based on core and memory size and formulate a waste metric that captured over 90% of the wastage in cloud workloads. In addition, we propose a cost reduction algorithm that helps to save nearly 4 million dollars. Furthermore, we developed a carbon awareness algorithm that could substantially reduce the carbon emissions of VMs by 51%.**

***Index Terms*—cloud computing, Microsoft Azure, virtual machines, cost reduction, carbon reduction, workload analysis**

I. Introduction

Cloud computing has experienced significant growth in recent years, driven by the growing demand for cloud computing infrastructure. According to a report by FMI [1], the global VM market is expected to grow at a Compound Annual Growth Rate (CAGR) of 20.3% between 2023 and 2033. The market was valued at US $5,174.3 million in 2022 and is expected to reach US $26,042.8 million by 2033.

As cloud computing continues gaining momentum, organizations increasingly rely on cloud-based infrastructure to support their computing needs. However, cloud costs can spiral out of control if not managed properly, with VM usage significantly contributing to cloud costs. The report by Flexera [2] has indicated that enterprises wasting over 30% of their cloud spending, with wasted spending totaling $14.1 billion annually. As a result, reducing cloud expenditure has become a top priority for organizations using the cloud.

To address this issue, we investigate the problem of reducing costs and carbon emissions of cloud computing. To our knowledge, only a few works have attempted to reduce costs using real VM workloads, such as [3]–[5]. Furthermore, most of the works on energy-efficient management systems for VMs in the cloud proposed various migration algorithms [12] [13]. Our study deviates from existing literature and instead looks at carbon reduction through VM scheduling. Specifically, we make the following contributions.

Real-world workload analysis. To understand the waste problem, we analyze the characteristics of approximately 2.7 million VMs in the Azure Public Dataset [14] to verify the existence of waste in cloud computing. Our workload analysis shows that the average utilization rate of all VMs is only 15.59%. Moreover, we observe a consistent pattern that as the VM requests more resources (such as memory and cores), its average utilization rate tends to decrease.

VM Cost Prediction Model. As the dataset does not indicate cost per VM, our study utilizes a linear regression algorithm to estimate the cost of each VM using their specific characteristics. The algorithm considers various factors, including public pricing models by Microsoft [15], how many gigabytes of memory the VM has, and the number of cores the VM has. By accurately predicting the cost of each VM, our algorithm reveals the estimated cost of each VM.

Waste Quantification. Building on top of the proposed cost prediction model, we further propose a metric to quantify the waste of each VM. In this study, we quantify the waste of each VM by considering the total cost and resource utilization. This metric provides users with a clear understanding of the extent of waste in the cloud infrastructure and enables them to prioritize cost optimization efforts. Our waste quantification offers key insights into users of the cloud and our analysis confirms that many users waste significant cloud resources, leading to unnecessary costs. This finding highlights the importance of cost optimization for VMs in cloud computing environments.

Cost Reduction Algorithm. Our study proposes an effective solution for reducing the cost of cloud computing. Using the established waste metrics, we find VMs with high wastage frequently possess larger cores that are underutilized. Therefore, our proposed strategy involves downgrading VMs by reducing their core size until it reaches the minimum core size or the utilization surpasses 100%. By implementing this approach, users can effectively reduce costs without compromising performance. The results from our simulations confirm that we have achieved cost savings of up to 17%.

Carbon Reduction Algorithm. Our research introduces an innovative carbon reduction algorithm that aims to maximize the efficiency of cloud computing while minimizing carbon emissions. To achieve this, we utilize the Carbon Intensity

 ISBN: 978-1-68558-097-1

Data provided by WattTime [17] to assess the relative carbon reductions achievable by each VM during its operation. This data measures the emission rate of electricity generators on the local grid at a specific time, represented in units of total pounds of emissions per megawatt-hour. The dataset we utilize contains carbon intensity levels for California ISO (CAISO) region in July, recorded at 5-minute intervals. Leveraging this information, our carbon reduction algorithm makes predictions about the carbon intensity expected in the next 24 hours. It then strategically assigns VMs to a time window where they can contribute to carbon emissions reduction effectively. By adopting this approach, users can actively minimize their carbon footprint without compromising the quality of service.

The rest of the paper is organized as follows. Section II discusses related work, and Section III presents a thorough analysis of the workload. The cost reduction algorithms are presented in Section IV, and Section V presents the carbon reduction algorithm. Section VI provides an in-depth discussion of the experimental results obtained. Finally, Section VII presents the conclusions drawn from this research and potential future work.

II. Related Work

Several studies have investigated cost reduction for VMs in the cloud, offering valuable insights into optimizing resource allocation and managing costs. For instance, Flexera's State of the Cloud 2022 report [2] highlighted the increasing usage of cloud services and provided recommendations for cost management from a company's perspective. Others such as Cortez et al. [3] explored resource management and proposed a workload prediction model that can leverage machine learning techniques to enhance resource allocation in large cloud platforms. Similarly, Hadary et al. [6] developed a system that employed machine learning algorithms to automate VM allocation for large-scale cloud deployments. However, these studies did not address cost reduction from the user's standpoint.

In the realm of published literature focusing on carbon reduction strategies, several approaches have been explored, primarily centered around individual or multiple data centers. Notable examples include [7] and [8]. However, some researchers, as demonstrated by [9], have taken into consideration distributed data centers with varying carbon footprints and power usage effectiveness. Meanwhile, Beloglazov and Buyya [10] have evaluated heuristics aimed at dynamically reallocating VMs to minimize energy consumption. Furthermore, several other studies have examined alternative methodologies, such as migration algorithms that may pose challenges for users rather than enterprises [11] [12].

In contrast, our study analyzes the specific workloads of individual users and presents a comprehensive solution tailored to their needs. We introduce a program that automatically leverages historical data, utilization rates, and other factors to recommend the most cost-effective VM for a given workload. By directly addressing user requirements, our solution bridges the gap in existing research and offers a practical approach to cost reduction. We also propose a program that suggests users to reschedule work when possible to improve environmental sustainability. Users can significantly enhance their cost and carbon management practices, achieving substantial cost reductions without compromising performance or quality. The unique focus of our paper on individual users sets it apart from previous studies, making it a valuable contribution to the cloud computing cost and carbon reduction field.

III. Workload Analysis

In this section, we characterize the workload of Azure VMs. Exploring their characteristics and focusing on which intrinsic aspects of application and function will help enable numerous platform optimizations.

A. Microsoft Azure Workload

Dataset. In this paper, we utilize the Azure Public Dataset Version 2 [13], a cloud trace generated in 2019 from the Microsoft Azure platform. The dataset includes detailed information of approximately 2.7 million VMs created by over 6,600 users in July 2019, such as the timestamp in seconds when the VM is first created (starting at 0), the timestamp in seconds when the VM is deleted, VM size in terms of maximum core and memory (GBs), the average and maximum CPU utilization, as well as the string IDs of users.

B. Cost Analysis

Tools. While the Azure Public Dataset Version 2 [13] provided relevant information concerning VM characteristics, it excluded essential information regarding the price of each VM, which is essential for calculating the cost of each VM. To solve this problem, we utilize third-party tools such as the pricing calculator offered by Azure Microsoft [14], Orange [15], and SciKit Learn [16] to develop a cost prediction model.

Core/CPU	Mem	Cost
2	2	--
2	4	0.091
2	8	0.134
4	8	0.191
4	32	0.297
8	32	0.537
8	64	0.594
24	64	--
30	70	--

Figure 1. Data used to calculate cost

Pricing Calculator. To analyze the cost of all VMs, we first need to find the pricing of each VM. The pricing calculator offered by Azure Microsoft [14] calculates the prices of each

VM based on numerous factors. We set the VMs to be in the region of the East US, Windows operating system, OS-only type, standard tier, and memory-optimized category for 730 hours. In choosing instances, we picked instances that matched the core and memory with the lowest temporary storage. However, not every VM reported on the Azure Public dataset Version 2 was provided on the pricing calculator. As seen in Figure 1, there were instances of core and memory pairs that were not provided on the pricing calculator. The cost in the figure was measured in cost per hour.

Core/CPU	Mem	Cost
1	0.75	0.02
2	3.5	0.12
4	7	0.24
8	14	0.25
12	48	0.595
16	64	0.896
32	128	1.792
...	...	...

Figure 2. First few row of our training dataset

Linear Regression. We use the linear regression tool from Sckikit Learn [16] to solve this problem. In addition to the data shown in Figure 1, we included other data from the pricing calculator in order to train our our model. The first few rows of our training dataset can be seen in Figure 2. We chose linear regression because the variables, namely core size and memory space, exhibit a linear relationship. Hence, linear regression is more suitable for capturing this linear association, as opposed to quadratic or alternative regression models. Through our linear regression model, we obtain the coefficient of determination R^2 as 0.98566, signaling the accuracy of our prediction model. Our model is determined through the following equation:

$$price = 0.01530 + 0.00055 \times core + 0.014222 \times mem$$

Where 0.01530 represents the intercept, 0.00055 constitutes the slope for the core variable, and 0.014222 forms the slope for the memory variable.

Additionally, compared to other machine learning techniques, linear regression handles overfitting relatively well using different procedures such as reduction techniques, regularization, and cross validation. While comparing our linear regression results to other machine learning techniques, we found the linear regression results to be more accurate.

Finally, with the price of VMs of different core sizes and memory space combinations, we are able to calculate the cost of each VM based on runtime using the following equation:

$$cost = price \times (\frac{runtime}{3600})$$

The VM cost is calculated as the product of the VM's price, which depends on its core size and memory space, and the duration in hours. The runtime is calculated by subtracting the deletion timestamp from the creation timestamp. To convert this duration into hours, we divided the result by 3600, as there are 3600 seconds in an hour.

C. Utilization Analysis

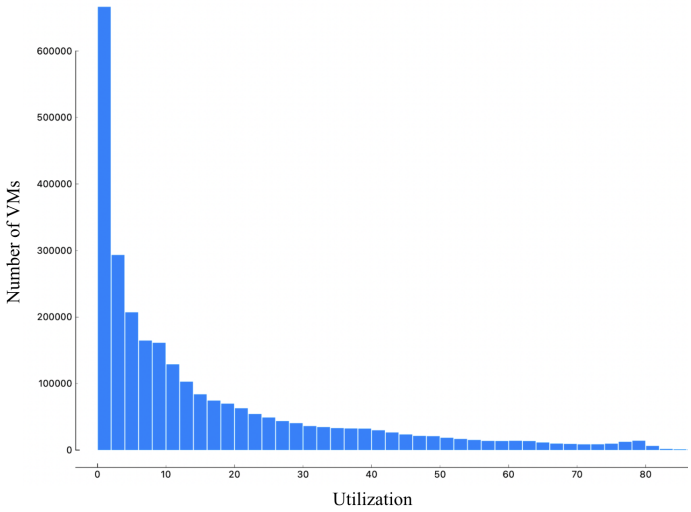

Figure 3. Utilization of VMs

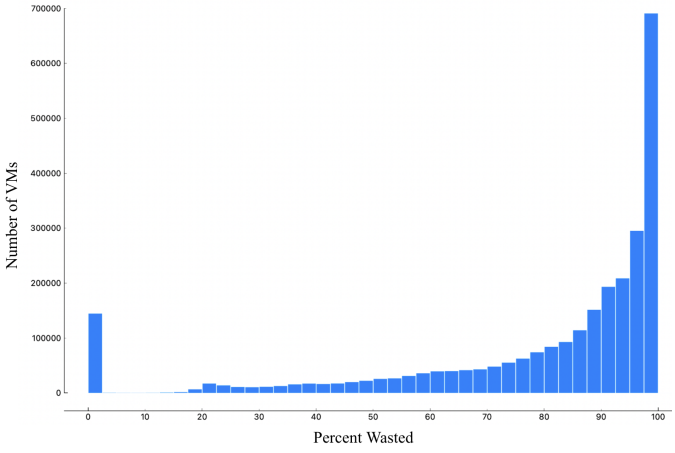

Figure 4. Percent wasted for VMs

User Analysis. Many Azure users created more than one VM. Based on different users, some VMs averaged with low core and memory count, while other VMs averaged with higher core and memory count. Although we observe that some users can efficiently utilize cloud resources, a large group of users also seem to be managing their resources ineffectively. We will refer to the first group of users as the "experienced" users and the second group as the "inexperienced users."

Experienced users all utilized VMs in an indistinguishable manner. Most importantly, their VMs consisted of low core

 ISBN: 978-1-68558-097-1

and memory count (often two cores and two GB). Their VMs ran for a short time rather than the entire 30 days analyzed in the dataset. Their experience was also evident in their high utilization rates, which signified that these experienced users were using their VM for a limited time before deleting it immediately. Inexperienced users utilized VMs in opposition to experienced users. Their VMs consisted of high core and memory count (many of which used the maximum 64 cores and 70 GB).

Additionally, these VMs had a duration averaging 30 days with extremely low utilization rates, sometimes running below 1%. The characteristics of the VMs created by inexperienced users portray the typical scenario of creating a VM and forgetting to shut it down when it is no longer needed. Consequently, these VMs cause significant waste.

VM Analysis. The distribution chart depicted in Figure 3 illustrates a negative exponential curve. Analyzing the chart more in-depth reveals that more than half (55%) of VMs had a utilization rate below 10%. The largest bar indicates about 24% of VMs being utilized at below 2%. More significantly, 24% of VMs had a utilization rate of less than 2%. The analysis shows cause for concern since most VMs show wasteful spending, and many of these VMs are generated by inexperienced users. The trends in Figure 3 correlate with the trends depicted in Figure 4. The percentage wasted for VMs was calculated through a cost over waste ratio. The data in Figure 4 acknowledges that 51% of VMs wasted more than 90% of their expenses. Moreover, 25% of VMs drained more than 97.5% of costs. Note that 5% of VMs had zero percent wasted due to their time of use being zero seconds. Overall the two trends detailed in Figures 3 and 4 emphasize the potential for a significant cost reduction, which this paper seeks to contribute.

D. Waste Analysis

The waste of cloud expenditure is calculated using the formula below:

$$waste = \frac{cost \times (100 - util)}{100}$$

According to our calculations, the combined cost of all VMs exceeds $23 million, with an estimated waste of over $20 million. These figures highlight that customers are squandering approximately 90% of their VM resources. The disparity between waste and cost emphasizes the urgency to reduce cloud cost and waste.

IV. Cost Reduction Algorithms

Recall Section III. C, inexperienced users utilize large VMs at lower utilization rates, leading to some of the most significant waste in the cloud. Inspired by this observation, we propose two cost reduction algorithms that are referred to as aggressive downgrading approach and passive downgrading approach.

Downgrading a VM involves reducing its core size to the smallest possible core. Instead of considering memory space, we prioritize core size when downgrading VMs since utilization issues usually relate to core size rather than memory space. The aggressive approach focuses on downgrading VMs based on cores and the lowest memory, while the passive approach considers downgrading VMs based on the core with the highest memory.

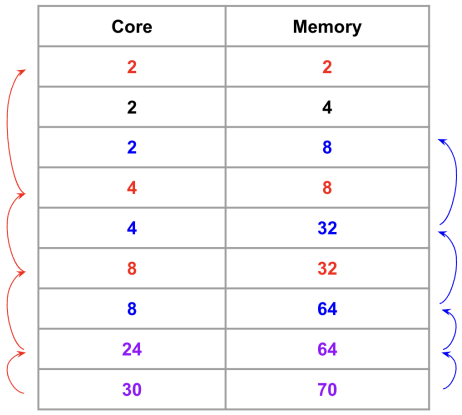

Figure 5. A list of VM core/memory combination

A. Sample Cases and Approaches

The two approaches are illustrated in Figure 5 through the pathways on the left and right sides of the table, respectively. In the first approach, which is more aggressive, the downgrading is performed based on the cores with the lowest memory. Let us consider an example where we attempt to downgrade a VM with a core size of 30 and 70 GB of memory. Following the “aggressive” approach (indicated by the red arrows on the left), the VM is downgraded to the core level with the smallest memory space.

On the other hand, the second approach adopts a passive strategy by downgrading based on the cores with the highest memory. If we were to apply the “passive” approach to downgrade a VM with a core size of 30 and 70 GB of memory, it would follow the pathway depicted on the right side of Figure 5, indicated by the blue arrows, resulting in a downgrade to the core level with the largest memory space.

B. Threshold Selection

It would be highly inefficient to downgrade every VM in the trace. To identify VMs eligible for downgrading, we established a criterion that selected VMs must have exceeded a 10% waste threshold, as indicated by the distribution chart shown in Figure 4.

Considering the limitations imposed by maximum utilization, average utilization, and core sizes, our proposed approaches necessitate VM utilization to fall within a range of 0% to 33%-50%. Let us assume a VM with 8 cores is downgraded to 4 cores. To calculate the new utilization, we divide the two core sizes, resulting in a quotient of 2, and multiply it by the old utilization rate. Since the maximum utilization possible for all VMs is 100%, the theoretical maximum utilization can only reach 50%. Similarly, if we

 ISBN: 978-1-68558-097-1

Algorithm 1 Aggressive Downgrading

Require: $percentWasted \geq 10$
  while $canDowngrade$ *and* $core >= 2$ **do**
    if $core = 2$ **then**
      $newMaxUtil = 2/2 \times maxUtil$
      $newUtil = 2/2 \times util$
      $core = 2$
      $mem = 2$
    else
      $newMaxUtil = oldCore/newCore \times maxUtil$
      $newUtil = oldCore/newCore \times util$
      if $newMaxUtil < 100$ **then**
        $maxUtil = newMaxUtil$
        $util = newUtil$
      else
        $canDowngrade = false$
      end if
    end if
    if $newMaxUtil > 100$ *or* $newUtil > 100$ **then**
      $core = coreLevel$
      $mem = memLevel$
    end if
  end while

Algorithm 2 Passive Downgrading

Require:
  $percentWasted \geq 10$
  $core \geq 2$ and $mem \geq 8$
  while $canDowngrade$ and $core >= 2$ **do**
    if $core = 2$ **then**
      $newMaxUtil = 2/2 \times maxUtil$
      $newUtil = 2/2 \times util$
      $core = 2$
      $mem = 8$
    else
      $newMaxUtil = oldCore/newCore \times maxUtil$
      $newUtil = oldCore/newCore \times util$
      if $newMaxUtil < 100$ **then**
        $maxUtil = newMaxUtil$
        $util = newUtil$
      else
        $canDowngrade = false$
      end if
    end if
    if $newMaxUtil > 100$ or $newUtil > 100$ **then**
      $core = coreLevel$
      $mem = memLevel$
    end if
  end while

consider a VM with 24 cores downgraded to the next level, 8 cores, following the same principle, the maximum utilization can only be 33.3%. These restrictions significantly contribute to the efficiency of targeting and downgrading VMs.

When the VM initiates the downgrade process, it computes and assigns the new maximum utilization, average utilization, memory, and cores to their respective variables. If either the maximum or average utilization surpasses 100%, the downgrade process is halted since utilization cannot exceed 100%. Likewise, if the VM reaches the minimum downgrade level with a core size of 2, the downgrade process is also concluded.

V. Carbon Reduction Algorithm

VMs consume massive amounts of energy, often from fossil fuel-based sources, thus leading to significant carbon emissions. This section presents a carbon reduction algorithm that identifies proper VMs and reschedules them to a low-carbon window to reduce their carbon footprint.

A. Carbon Intensity

Carbon intensity is the measurement of emissions generated relative to the energy consumed. In this paper, we use the Marginal Operating Emissions Rate (MOER), a widely used metric, to measure the carbon intensity in units of pounds of emissions per megawatt-hour. MOER measures the emissions rate of electricity generators on a certain grid at a certain time. Low MOER indicates that electricity is more environmentally friendly and vice versa. The Azure trace [13] was created with data from July 2019. Consequently, we utilized the corresponding MOER dataset from CAISO North during July 2019 as our carbon intensity data.

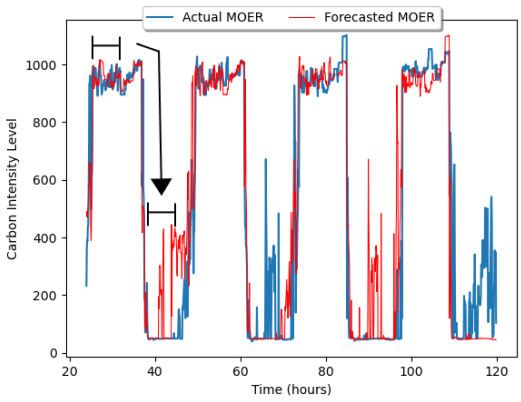

Figure 6. Forecasted and actual MOER from July 2-5, 2019

As illustrated by the blue line in Figure 6, MOER values in the CAISO North region vary greatly, but typically surges during the day when electricity demand is higher. This increased energy consumption by businesses and households often prompts grids to deploy additional generators, including those with higher emissions, to meet the heightened demand. Conversely, demand and MOER tend to decrease during the night, enabling a shift towards more sustainable electricity generation. This variability in MOER arises from the interplay

 ISBN: 978-1-68558-097-1

of numerous external factors that shape the dynamics of the electricity grid, which include weather and the makeup of generators within the grid (renewable or non-renewable).

B. Carbon Intensity Forecasting

Forecasting MOER allows workloads to be delayed until times of low MOER to reduce carbon emissions. Our forecasting algorithm uses a robust prediction methodology. Our methodology operates under the assumption that the MOER levels observed today will persist unchanged into tomorrow. Alternative forecasting models, such as the moving average approach, were also evaluated and tested. Nonetheless, due to challenges posed by data availability, the moving average model encountered limitations in precisely anticipating MOER levels.

The subsequent component of the algorithm centers on optimizing carbon reduction. This is achieved by identifying the interval with the lowest MOER level averages over the next 24 hours to execute the workload. Operational activities are postponed and rescheduled until the interval mentioned above. As shown in Figure 6, workloads during the high MOER times are shifted to times of lower MOER levels, thereby reducing carbon emissions.

There are three types of workloads in the Azure Public Dataset "Delay-sensitive," "Delay-insensitive," and "Unknown." As their names suggest, delay-sensitive workloads are time-sensitive, delay-insensitive workloads are not, and unknown workloads do not specify whether it is time-sensitive. Because users did not mark unknown type workloads as delay-sensitive, we assumed they could be postponed. For our simulation, we ran both "Delay-insensitive" and "Unknown" type workloads through the optimization algorithm as they were more amenable to rescheduling and could tolerate delays.

Algorithm 3 Carbon Reduction

Require: $RunTime > 0$
  $AverageMOER = Average\ MOER\ in\ Window$
  $Window \leftarrow Start\ Time\ to\ EndTime$
  for $\left(\frac{1Day}{5minutes}\right)$ **do**
    $Shift\ Window\ By\ 5\ Minutes$
    $NewAverage \leftarrow Average\ of\ Modified\ Window$
    if $NewAverageMOER < AverageMOER$ **then**
      $AverageMOER \leftarrow NewAverageMOER$
      $BestWindow \leftarrow Window$
    end if
  end for
  $Reschedule\ Task\ to\ BestWindow$
  $Savings = DefaultEmission - AverageMOER$
  $PercentSavings = 100 \times \left(\frac{Savings}{DefaultEmission}\right)$

Our carbon reduction program, found as Algorithm 3, works by identifying the length of the window and shifting it until it finds the best duration with the lowest MOER levels. The workload depicted in Figure 6 shows the results of optimization. The program matched the window length to the corresponding MOER levels. It then shifts the window and calculates the new average; if the new average is lower than the past lowest average, the new average is set as the lowest average, and the best window is set to that duration. The window shifts every 5 minutes to match the MOER data that only provides data in five minute intervals. The program then loops 720 times, the number of 5 minute intervals in a day, to find the lowest average and the corresponding window. Then, it calculates the percent saving after optimization.

VI. Experimental Results

In this section, we will discuss the experimental results from the simulations on the cost and carbon reduction programs.

A. Cost Reduction Results

We ran our simulations on the entirety of the 2019 Azure cloud trace [13]. First, we calculated the total cost of over 2.6 million VMs in the trace, resulting in a total of $23,144,128.53 before downgrading. Then, we ran two simulations to compare user savings using the two cost reduction algorithms presented in Section IV. After downgrading, the cost savings were astonishing. Through our first downgrading approach, 1,975,282 VMs were aggressively downgraded and saved users a total of almost $4 million or 17% in savings. Through our second downgrading approach, 730,436 VMs were passively downgraded and saved users a total of about $950,000 or 4% in savings.

B. Carbon Reduction Results

We ran the carbon reduction simulation on both"Delay-insensitive" workloads and "Unknown" type workloads. In the"Delay-insensitive" group, we observed savings ranging from 0 to 8%, with mean savings around 0.2%. In the "Unknown" group of workloads, the average savings was 55%. Percent savings overall averaged 51%. The drastic difference in outcomes is explained by the differences in run time each group has. "Delay-insensitive" workloads run for long durations, with an average of 26 days and peaks around 30. These lengthy workloads are too substantial to generate significant savings through optimization, as they are too large to accommodate the dips of MOER. Meanwhile, the average run time of "Unknown" type workloads is only approximately four and a half hours, making them more susceptible to generating sizeable savings during optimization.

VII. Conclusion and Future Work

This paper analyzed the Azure workload of over 2.6 million VMs. We developed two cost reduction algorithms and a carbon reduction algorithm. The experimental results have shown that our proposed algorithms can help reduce costs of up to 17% of cost by efficiently choosing core size and memory space and reducing on average 51% of carbon emissions by rescheduling workloads to a low carbon time. In the future, we would like to explore the relationship between cost savings and carbon reduction.

Our proposals faces a few limitations. While the cost reduction algorithm is limited by sheer number of VMs in

 ISBN: 978-1-68558-097-1

the dataset, the carbon reduction algorithm is limited by the challenge of accurately predicting MOER due to its complex nature influenced by various external factors. Our algorithm adopts a simplified approach, primarily relying on historical MOER data to extrapolate future levels. This simplification enhances computational efficiency and operational feasibility but sacrifices precision by disregarding the intricate dynamics underlying MOER fluctuations. This can be addressed in future work by developing a more sophisticated algorithm incorporating external factors to achieve more precise forecasting of MOER levels.

Our study was conducted with the assumption that we can change VM size and run time whenever it is needed. However, this does not reflect the real world scenarios 100% of the time. Thus although our solution aims to tackle memory space issues through the duo use of the aggressive and passive algorithm, real world applications of the cost reduction algorithm could still be constrained by VM availability and memory space; a VM may still be reduced to a level where the program does not have enough memory to run. Additionally, both the cost and carbon reduction algorithms would need user consent to downgrade their VMs, as some users may require a specific VM size without changes, or other users may not be able to reschedule their VM workloads.

Overall, we believe inexperienced users mentioned in Section III. C would benefit the most from our cost reduction program. Users with workloads that can be rescheduled would save the most carbon reductions. Both the cost and carbon reduction algorithms are recommended to be implemented in real cloud providers such Microsoft Azure, Amazon Web Services (AWS), Alibaba etc.

REFERENCES

[1] "Virtual Machine Market," FMI, Dec. 2022. https://www.futuremarketinsights.com/reports/virtual-machine-market (accessed Jul. 2023)

[2] "State Of The Cloud Report," Flexra, 2022.

[3] E. Cortez, A. Bonde, A. Muzio, M. Russinovich, M. Fontoura, and R. Bianchini, "Resource Central: Understanding and Predicting Workloads for Improved Resource Management in Large Cloud Platforms * CCS CONCEPTS," 2017. Accessed: May 2023. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2017/10/Resource-Central-SOSP17.pdf

[4] M. Shahrad et al., "Serverless in the Wild: Characterizing and Optimizing the Serverless Workload at a Large Cloud Provider," USENIX, 2020. Accessed: May 2023 Available: https://www.usenix.org/conference/atc20/presentation/shahrad

[5] B. Everman, M. Gao, and Z. Zong, "Evaluating and reducing cloud waste and cost—A data-driven case study from Azure workloads," Sustainable Computing: Informatics and Systems, vol. 35, p. 100708, Sep. 2022, Accessed: May 2023. [Online] doi: https://doi.org/10.1016/j.suscom.2022.100708.

[6] O. Hadary et al., "Protean: VM Allocation Service at Scale," www.usenix.org, 2014. Accessed: May 2023. [Online]. https://www.usenix.org/system/files/osdi20-hadary.pdf

[7] A. Berl et al., "Energy-Efficient Cloud Computing," The Computer Journal, vol. 53, no. 7, pp. 1045–1051, Aug. 2009, Accessed: May 2023. [Online] doi: https://doi.org/10.1093/comjnl/bxp080.

[8] A. Uchechukwu, K. Li, and Y. Shen, "Energy Consumption in Cloud Computing Data Centers," International Journal of Cloud Computing and Services Science, vol. 3, no. 3, Jun. 2014.

[9] A. Khosravi, S. Garg, and R. Buyya, "Energy and Carbon-Efficient Placement of Virtual Machines in Distributed Cloud Data Centers," in Proceedings of the 19th international conference on Parallel Processing, Aug. 2013, pp. 317–328.

[10] A. Beloglazov, "Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing," 2010.

[11] M. Giacobbe, A. Celesti, M. Fazio, M. Villari, and A. Puliafito, "An Approach to reduce carbon dioxide emissions through virtual machine migrations in a sustainable cloud federation," IEEE Xplore, Apr. 01, 2015.

[12] S. Supreeth and K. Patil, "VM Scheduling for Efficient Dynamically Migrated Virtual Machines (VMS-EDMVM) in CLoud Computing Enviornment," KSII Transactions on Internet and Information Systems., vol. 16, no. 6, pp. 1892-1912, Jun.2022.

[13] E. Cortez, A. Bonde, A. Muzio, M. Russinovich, M. Fontoura, and R. Bianchini, "AzurePublicDatasetV2," Github, 2017. [Online]. Avaliable: https://github.com/Azure/AzurePublicDataset

[14] Microsoft, "Pricing Calculator — Microsoft Azure," Microsoft.com, 2023. https://azure.microsoft.com/en-us/pricing/calculator/ (accessed Jun. 2023)

[15] J. Demšar et al., "Orange: Data Mining Toolbox in Python," Journal of Machine Learning Research, vol. 14, no. 71, pp. 2349–2353, 2013, Accessed: Jul. 12, 2023. [Online]. Available: http://jmlr.org/papers/v14/demsar13a.html

[16] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python" Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011, Available: https://jmlr.csail.mit.edu/papers/volume12/pedregosa11a/pedregosa11a.pdf

[17] "API Reference," www.watttime.org https://ww.watttime.org/api-documentation/introduction (accessed Jun. 23, 2023).

 ISBN: 978-1-68558-097-1