THE UNIVERSITY OF CHICAGO

TOLERATING CAPACITY VARIATION: CLOUD RESOURCE MANAGEMENT TO AVOID TERMINATIONS

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
MASTER

DEPARTMENT OF COMPUTER SCIENCE

BY
RAJINI WIJAYAWARDANA

CHICAGO, ILLINOIS
FEBURARY 2025

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

There are so many people who have supported and inspired me throughout this journey. While I cannot name everyone, I want to express my gratitude to those whose contributions have been transformative.

I am deeply grateful to my advisor, Professor Andrew Chien. Your unwavering guidance, encouragement, and intellectual rigor have been a cornerstone of my growth as a researcher. It is an honor to learn from you.

To my committee members, Professors Sanjay Krishnan, Junchen Jiang, and Yves Robert, thank you for your support and encouragement throughout this process.

I am profoundly thankful to my collaborators from ENS Lyon – Professors Yves Robert, Anne Benoit, and Frédéric Vivien – for inspiring and enriching my research.

I extend my heartfelt gratitude to the LSSG team for their invaluable feedback, insightful discussions, and steadfast motivation throughout this journey.

To my husband, Chalitha – your love, patience, and unwavering belief in me have been an endless source of encouragement and comfort. To my parents, Gamini and Priyani Wijayawardana, who have been my greatest pillars of strength, offering unconditional love and support every step of the way. To my brother, Rajitha, my aunts, Sriyani and Shalini, and my uncle, Neville – your kindness and belief in me have carried me through every challenge. Finally, to my friends, thank you for your laughter and constant presence.

This journey has been profoundly rewarding, and I could not have done it without each of you. Thank you from the bottom of my heart.

ABSTRACT

The growth of cloud data centers has raised significant sustainability concerns and led to a power grid crisis. In this landscape, data center capacity is increasingly determined by external constraints, including power availability, carbon intensity and power prices.

We characterize the impact of variable capacity on cloud workload performance by evaluating commercial and synthetic cloud workloads under traditional scheduling. With capacity change frequency of 0.25–8 per hour, Microsoft Azure and Google's Borg cloud workloads can suffer goodput losses of 12–24% and 5–19%. Generalizing, we study synthetic heavy-tailed workloads and a range of carbon intensity-driven capacity variation. Our studies show that goodput loss increases with heaviness of tail and with increased capacity variation. We identify job terminations as the crucial phenomena and focus on it as the critical performance metric and scheduling objective.

We derive the machine interval distribution from imposed capacity variation and exploit it to reduce terminations under variable capacity. We explore four heuristics with varying levels of machine interval information, including stable machine indices, time between capacity changes, statistical and probabilistic information of interval time remaining. The best performing heuristic uses uptime of an interval to estimate the conditional probability of remaining time. This information is used to improve the assignment of jobs to intervals, reducing terminations by 32%, with 0.9% goodput loss.

We later combine these heuristics to build the Interval-Aware Scheduler (IAS) that reduces terminations across a range of heavy-tailed workloads. On a synthetic workload, IAS lowers terminations by 98% over First-Fit, with a 1.25% goodput loss. IAS can achieve 5% greater goodput, with 32% lower terminations. In the extremely heavy-tailed Azure workload, IAS lowers terminations by 92%, with a 0.05% goodput loss.

CHAPTER 1

INTRODUCTION

1.1 Motivation

Data centers continue to be a rapidly growing segment of the economy, with a CAGR of 9% through 2030 [65]. According to estimates from the Lawrence Berkeley National Laboratory, the data center load is projected to increase from 4.4% of the total US electricity demand in 2023 to 6.7–12% by 2028 [58]. This unprecedented growth in cloud data centers has raised sustainability concerns and led to a power grid crisis. Amid mounting public pressure, hyperscalers have set ambitious sustainability goals. Google aims to operate on 24/7 carbon-free energy across all power grids they are located in by 2030 [45] and Microsoft aims to be carbon negative by 2030 [18]. Moreover, power grids are becoming increasingly strained to meet the energy demands of new data centers [51, 53]. Several power grids have halted and slowed new data center connections to ensure grid reliability [41, 63, 71]. In this landscape, data center flexibility is a crucial consideration [46, 22].

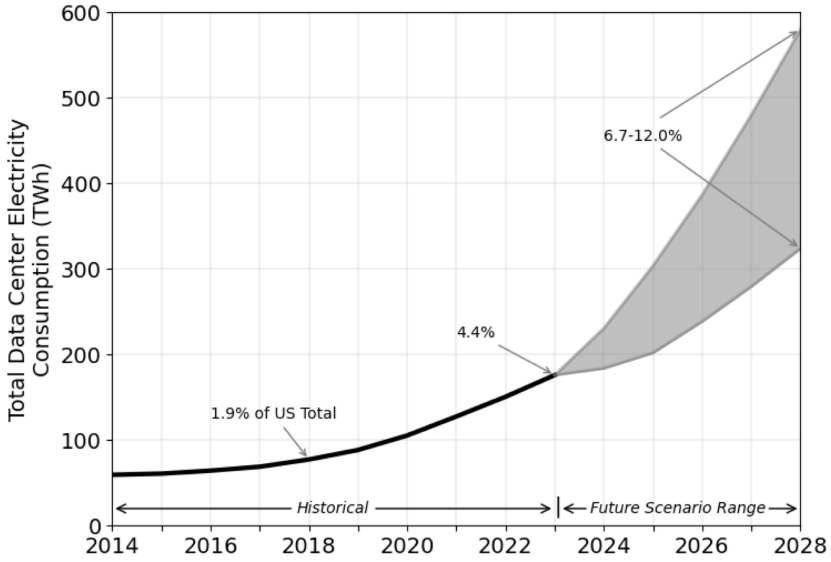


Figure 1.1: Total U.S. data center electricity use from 2014 to 2028, as reported in [58].

A data center's participation in a power grid's demand response program [25, 10], the operation of a renewable-powered data center [31, 37], and the opportunistic use of stranded renewable generation by the data center [13] are several practical applications of variable capacity data centers. Data centers are traditionally viewed as fixed capacity entities, where resource management would not take into account fluctuations in resource capacity. However, in the variable capacity model, a data center's resource capacity is driven by external constraints that are beyond the data center operator's control. This adds a layer of complexity to resource management. If not managed, this would lead to lost opportunities from unexpected capacity increases and performance losses from unexpected capacity decreases, impacting the system's overall performance [73, 72].

The underlying capacity variation structure determines the availability of individual machines. This gives rise to a set of machine intervals, where the machine is continuously powered on and available for scheduling. There is potentially a wealth of information related to machine intervals that could inform the scheduler of a machine's future availability. In this thesis, we explore how information on machine intervals could be effectively exploited to improve variable capacity scheduling.

1.2 Thesis Statement

Novel scheduling algorithms can restore the performance loss in cloud workloads resulting from externally-driven variable capacity, by exploiting machine interval characteristics to predict time remaining in an interval.

1.3 Approach

We evaluate the impact of variable capacity on cloud workload performance. We determine that job terminations are the primary cause of performance degradation, leading to goodput

loss and higher scheduling latency. In the cloud service model, this has additional implications, including lost application state, checkpointing overhead, SLO violations, and degraded user experience. Therefore, we propose job terminations as a critical workload performance metric and primary scheduling objective.

We identify machine interval characteristics that could inform the scheduler of the time remaining in an interval. Using this information, we propose and evaluate a class of scheduling heuristics that aim to avoid job terminations under variable capacity, by better aligning jobs-to-machine intervals, while maintaining the system's goodput and scheduling latency.

1.4 Contributions

We identify and address the challenges of workload scheduling under variable capacity. The key contributions of this thesis are:

- Characterize cloud workload performance under traditional scheduling on a variable capacity platform. Variable capacity leads to significant performance degradation, with goodput losses of 12–24% in Azure and 5–19% in Borg, with increasing change frequency. We determine that interval ends under variable capacity cause job terminations, resulting in goodput loss. Under observed variable capacity in the MISO, ERCOT, SPP and CAISO grids, the Azure workload suffers goodput losses between 3.72–30.75%.

- Examine a set of synthetic workloads on the dimensions of job duration distribution and job resource size, to understand the impact of workload characteristics on performance under variable capacity. We find that heavier-tailed workloads experience greater goodput loss under variable capacity. As the mean job duration increases by approximately 2×, the job failure rate increases by an average of 1.93× and the mean scheduling latency increases by an average of 1.46×.

- Identify job terminations as the crucial performance metric and a primary scheduling objective for cloud workloads under variable capacity, as cloud applications rely on continuous service and immediate response.

- Analyze the impact of the underlying capacity variation structure on machine interval characteristics, and identify information that can be potentially exploited to predict interval time remaining allowing better job-to-interval alignment.

- Propose and evaluate a class of scheduling heuristics that avoid terminations by exploiting machine interval characteristics. We explore heuristics that prioritize big jobs on stable machines, aligns scheduling with capacity changes and leverage the statistics and probability distribution of interval time remaining to determine whether an interval would last long enough to complete the job. Combining these heuristics further improves performance, through structuring them to target distinct job populations. The best performing heuristic combination lowers terminations by 95% with a 0.2% goodput loss, while maintaining lower scheduling latency than First-Fit.

- Based on these heuristics, we build the Interval-Aware Scheduler (IAS) that drastically reduces terminations across a range of heavy-tailed workloads. IAS achieves 5% greater goodput than First-Fit, with a 32% reduction in terminations. We can further reduce terminations by up to 98% over First-Fit, if a 1.25% goodput loss is tolerable. For the extremely heavy-tailed Azure workload, IAS achieves 92% lower terminations under variable capacity, while maintaining similar goodput to First-Fit (-0.05%).

1.5 Thesis Organization

The remainder of the thesis is organized as follows. Chapter 2 provides a brief background on variable capacity platforms, cloud resource management and cloud workloads. In Chapter 3, we describe our research problem and scheduling approach. Chapter 4 highlights the

challenges of variable capacity on cloud workload performance under traditional scheduling. We further explore the impact of workload characteristics on performance under variable capacity. In Chapter 5, we examine how the underlying capacity variation structure affects machine interval characteristics. Chapter 6 proposes and evaluates a class of scheduling heuristics that exploit machine interval characteristics to avoid terminations under variable capacity. In Chapter 7, we discuss related research literature. Finally, we summarize our results and outline future research directions in Chapter 8.

CHAPTER 2

BACKGROUND

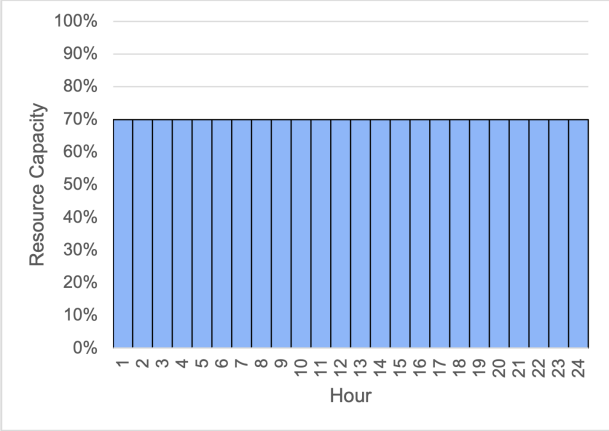
In this chapter, we provide the necessary context for understanding variable capacity platforms and its implication on cloud workload scheduling. Section 2.1 provides an overview of variable capacity platforms and examines the dimensions of variable capacity. Section 2.2 explores cloud data centers as candidates for variable capacity, and provides a brief overview of cloud resource management and cloud workload characteristics.

2.1 Variable Capacity Platforms

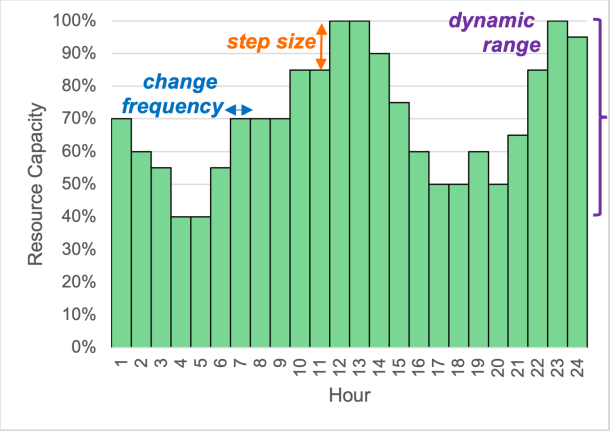
Data centers have traditionally been viewed as fixed capacity entities that require constant and continuous power supply. However, variable capacity platforms are characterized by their ability to dynamically adjust resource availability, driven by external constraints such as power availability, carbon content of power and power prices [73, 13, 54]. Understanding the characteristics of the underlying capacity variation is critical for effective workload scheduling on variable capacity platforms.

2.1.1 Synthetic Models of Variable Capacity

Fixed capacity platforms are straightforward. They require constant and continuous power supply, which ensures a fixed capacity during each time interval (see Figure 2.1a). However, in variable capacity platforms, the resource capacity could fluctuate between time intervals. To systematically analyze the effects of variable capacity on workload performance, prior work has characterized key dimensions of variable capacity [73]. Below, we examine these dimensions, which form an abstract framework that allows specific examples to be characterized and generalized.



(a) Fixed Capacity platform, with average capacity = 70%



(b) Variable Capacity platform under a random walk structure, with average capacity = 70%, change frequency = 1, step size = 0.15, dynamic range = 60%

Figure 2.1: Contrasting Fixed Capacity and Variable Capacity platforms

- ***Change frequency***: Within the abstract framework for capacity capacity, we model capacity changes to occur at the boundaries of discrete time periods. The change frequency defines the number of such boundaries every hour.

- ***Step size***: To prevent large instantaneous capacity changes between time periods and to reflect physical constraints of real systems, we bound the capacity change between two consecutive time periods to a maximum step size.

- ***Dynamic range***: The range of machines over which capacity changes occur. This defines the upper and lower bounds of resource capacity.

2.1.2 Observed Variable Capacity

Amid mounting public pressure, hyper-scale cloud providers have set ambitious sustainability goals. Google, for instance, aims to operate on 24/7 carbon-free energy across all power grids they are located in by 2030. Data centers consume power from the power grid, and the associated carbon emissions depend on the grid's fuel mix and the characteristics of power

generation sources. In order to manage the operational carbon footprint of data centers, it is vital to consider the characteristics of the power grid they are located in.

We derive resource capacity traces based on power grids' carbon intensity, in order to model fluctuations in data center resource capacity driven by the carbon intensity of its power supply. We consider a simple constant carbon budget, where the resource capacity during each time period is constrained by a constant carbon budget and determined by the time period's average carbon intensity [73].

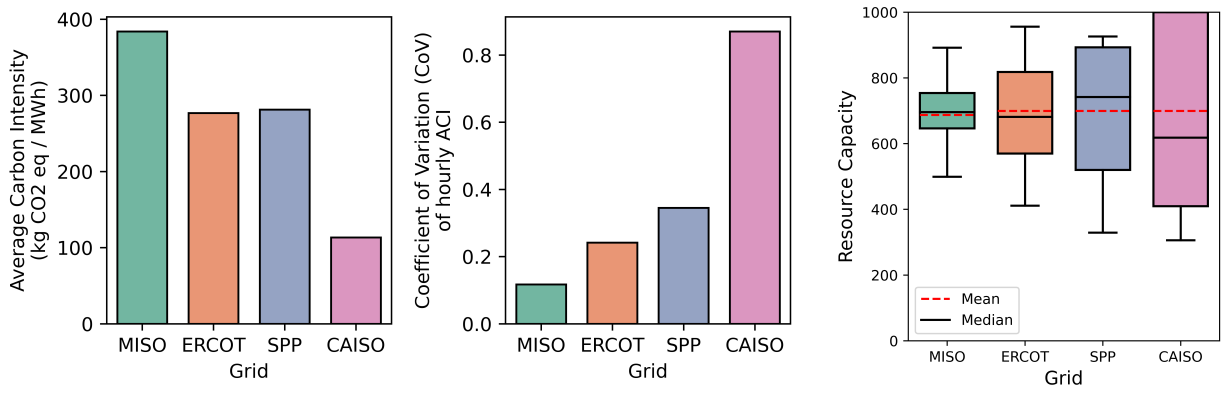
We consider four power grids with distinct characteristics, MISO (Midwest, USA) [49], ERCOT (Texas, USA) [24], SPP (Southwest, USA) [61], and CAISO (California, USA) [9]. Table 2.1 summarizes the main generation sources across the chosen power grids. MISO is a fossil fuel-heavy grid, with 66% of its generation from gas and coal. In contrast, CAISO has the largest fraction of renewable generation, with 19% from solar, 11% from wind and 13% from hydro. ERCOT reflects a mix of CAISO and MISO in terms of its generation sources, with 7% of generation from solar, 24% from wind, and 59% from gas and coal. SPP is a wind-heavy power grid, with 37% of its generation from wind.

Power grid	**Generation source**				
	Gas + coal	**Solar**	**Wind**	**Hydro**	**Nuclear**
MISO [27]	66%	–	16%	2%	15%
ERCOT [23]	59%	7%	24%	–	9%
SPP [55]	54%	–	37%	3%	6%
CAISO [17]	38%	17%	11%	13%	9%

Table 2.1: Generation mix across power grids (2023)

The properties of a grid's underlying generation sources affect its average carbon intensity and temporal fluctuations in carbon intensity. As a grid's renewable penetration increases, its average carbon intensity decreases. However, the intermittent properties of renewable generation sources leads to large fluctuations in carbon intensity between consecutive time steps. This results in a higher coefficient of variation of carbon intensity in grids with higher

renewable penetration, as shown in Figure 2.2a. Thereby, we order the four grids on a spectrum of increasing coefficient of variation of carbon intensity as, MISO, ERCOT, SPP and CAISO. The resulting distribution of variable capacity in a data center located in the grid would show increasing spread in resource capacity across this spectrum. In Figure 2.2b, we observe a dynamic range of 39.3% in MISO, 54.5% in ERCOT, 59.7% in SPP and 69.4% in CAISO. It is evident that grid's with higher renewable penetration experience greater capacity variation, due to the high intermittence of renewable generation.



(a) Power grid characteristics

(b) Distribution of variable capacity

Figure 2.2: The impact of power grid characteristics on the distribution of variable capacity in data centers located within the power grid

2.2 Cloud Data Centers and Workloads

In a landscape where data center growth is limited by power availability, variable capacity could assist hyperscale cloud providers in their growth and sustainability efforts. Our work focuses on cloud workload scheduling based on the fundamental properties of cloud schedulers. In this section, we provide a brief overview of cloud resource management and examine cloud workload properties.

2.2.1 Cloud Resource Management

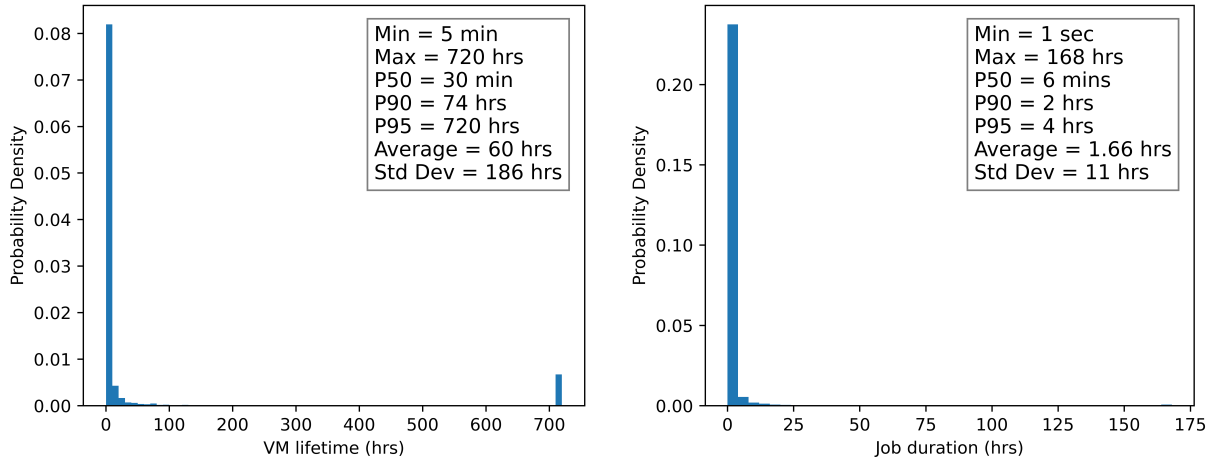
The objective of cloud resource management is to efficiently place jobs on cloud resources (machines), while meeting resource utilization and scheduling latency requirements of the platform. Fundamentally, this is an optimization problem formulated to find a good 'fit' between jobs and machines. Due to the proprietary nature of cloud schedulers, First-Come, First-Served (FCFS/First-Fit) is often considered a simplified approach to cloud job scheduling [7].

Cloud data centers host a variety of applications, including web hosting, content delivery, AI, Internet of Things (IoT), enterprise applications, and streaming services. A significant fraction of cloud workloads is interactive, requiring continuous service and immediate response. 28% of Microsoft Azure's workload is categorized as interactive [19]. In the Google Borg workload, the production tier accounts for approximately 30% of compute resource utilization [64]. Architecting cloud applications is a complex process, involving a multitude of services, including VMs, storage, network connections, REST API connections, databases, etc. These cloud services are governed by Service Level Objectives (SLOs), which outline specific performance metrics such as availability, uptime, and response times that cloud providers are contractually required to guarantee to their customers [47, 16, 57]. Failure to meet SLOs harms user experience and could impose financial penalties on the cloud provider.

The cloud's average resource utilization (compute) is approximately between 55–65% [64, 19]. A significant fraction of cloud resources remain idle due to resource fragmentation and the requirement to maintain low scheduling delays. For instance, in Google's Borg scheduler, 80% of jobs experience less than 5 seconds of scheduling delay [64], enabled by the platform's low resource utilization.

2.2.2 Cloud Workloads

Cloud workloads can be defined in various abstractions, such as Virtual Machines (VMs), containers, tasks, and bare-metal jobs. Microsoft Azure [48] and Google's Borg traces [32] are exemplar commercial cloud workloads, offering rich detail on job characteristics, resource utilization, and cloud scheduling. Microsoft Azure trace provides VM creation and deletion times, requested virtual cores and memory, and actual CPU utilization in 5-minute intervals. The Borg workload trace includes task start times, end times, requested CPU and memory (normalized to maximum machine size), and actual CPU and memory usage in 5-minute intervals. Given the vast amount of data in the traces, we study representative samples from the Azure and Borg workloads.



(a) Azure (b) Borg

Figure 2.3: Distribution of VM lifetime / job duration in the Azure and Borg workload traces

The distribution of job duration in both cloud workloads is heavy-tailed (see Figure 2.3), with a large number of short jobs and relatively few long jobs. The Azure workload trace contains a substantial number of continuously running VMs that span the entire duration of the trace, while the Borg workload contains mostly smaller duration jobs. As summarized in Table 2.2, the runtime of VMs in the Azure workload has an average of 60 hours, a median

of 30 minutes, and a P95 of 30 days. In the Borg workload, the average runtime is 1.66 hours, with a median of 6 minutes and a P95 of 4 hours. Table 2.2 further summarizes the characteristics of the representative samples from Azure and Borg workload traces, including requested parallelism and memory. It is evident that the Azure trace contains longer running jobs than Borg. This imposes implications on scheduling performance under variable capacity, as the distribution of job durations might be incompatible with machine interval lengths under variable capacity. We explore this relationship in detail in Sections 4.1, 4.3.1 and 6.3.

Workload	**Azure**	**Borg**
Trace year	2019	2019
Abstraction	Virtual Machines (VMs)	Jobs
Trace duration	30 days	7 days
Number of jobs	60,000	605,503
Runtime	Avg = 60 hrs P50 = 30 mins P95 = 30 days	Avg = 1.66 hrs P50 = 6 mins P95 = 4 hrs
Parallelism	Avg = 3.7 cores P50 = 2 cores P95 = 8 cores	Avg = 0.01 NCU P50 = 0.007 NCU P95 = 0.02 NCU
Memory	Avg = 16 GB P50 = 8 GB P95 = 32 GB	Avg = 0.004 NMU P50 = 0.002 NMU P95 = 0.015 NMU

Table 2.2: Key statistics of representative samples of cloud workload traces

2.3 Summary

Variable capacity platforms can be systematically analyzed across the dimensions of change frequency, step size and dynamic range. We examined variable capacity under a constant carbon budget across four power grids and found that the characteristics of the underlying capacity variation source determines observed variable capacity. The unprecedented growth of

cloud data centers positions them as a suitable candidate for variable capacity. We explored the characteristics of cloud resource management, including First-Fit scheduling, SLOs, and cloud utilization levels. Finally, we contrast the properties of the chosen commercial cloud workloads from Microsoft Azure and Google's Borg. The Azure workload has relatively longer jobs due to continuously running VMs, whereas the Borg workload contains a large number of short duration jobs.

CHAPTER 3

PROBLEM AND APPROACH

We explore the research problem, detailing the research questions we seek to address in this thesis. We provide a brief overview of our scheduling approach. Further, we formally define the scheduling framework, in terms of the variable capacity platform, machine intervals, the workload and the scheduling objective. Finally, we outline the experimental setup and define performance metrics.

3.1 Problem

The primary objective of a scheduling system is to make the best use of available resources. However, operating a variable capacity system adds a layer of complexity to achieving this objective. Resource capacity in a variable capacity system is driven by external phenomena. Resource capacity changes directly translate to machines being powered on/off based on power availability. Capacity loss results in the termination of jobs and VMs running on the powered off machine. From a scheduler's point of view, this is extremely undesirable, as it would result in goodput loss, violated SLOs, increased scheduling latency and a host of other workload performance implications [54, 73].

Traditional schedulers provide poor scheduling performance under variable capacity [54, 73], making variable capacity data centers infeasible for cloud workloads. Our research seeks to address the scheduling challenges of variable capacity, answering the following research questions.

- How does the introduction of variable capacity affect cloud workload performance under traditional scheduling?
- How do workload characteristics impact workload performance under variable capacity?

- How does the underlying capacity variation structure affect machine interval characteristics and what properties are potentially exploitable for scheduling?
- How can machine interval characteristics be effectively exploited to improve variable capacity scheduling?

3.2 Approach

The underlying capacity variation structure of a variable capacity platform determines the availability of individual machines. Analyzing the availability of individual machines in such a platform would produce a set of intervals, where each interval would correspond to a contiguous block of time during which a machine is powered on and available for scheduling. Profiling the underlying source of variable capacity provides insight into the future availability of a machine interval. We propose the use of such machine interval characteristics to avoid terminations under variable capacity. In our study,

- We characterize cloud workload performance under traditional scheduling on a variable capacity platform, in order to demonstrate the challenges in variable capacity scheduling. (see Sections 4.1 and 4.2)
- We examine a set of synthetic heavy-tailed workloads on dimensions of job duration distribution and job resource size, to understand the impact of workload characteristics on performance under variable capacity. (see Section 4.3)
- We analyze the impact of the underlying variation structure on machine interval characteristics, identifying information that can be potentially exploited for scheduling. (see Chapter 5)
- We propose and evaluate a class of scheduling heuristics that better aligns jobs to machine intervals, to reduce terminations under variable capacity. (see Chapter 6)

3.3 Formal Definition of the Scheduling Framework

Let M be the set of machines in the platform, where each machine $m_i \in M$ has $m_i.res$ resources. The platform experiences variable resource capacity, with $M(t)$ number of machines available at time t with $M(t) \leq |M|$. The platform follows a 'Last In, First Out' (LIFO) machine termination policy. Machines ($m_i \in M$) are ordered by index i, which is a measure of its relative risk for capacity variation.

Capacity changes happen at times $t' \in T$, such that $M(t') < M(t'-1)$ (capacity decrease) or $M(t') > M(t'-1)$ (capacity increase) or $M(t') = M(t'-1)$ (no change in capacity). The granularity of t' depends on the ***change frequency*** ($freq$), with capacity changes occurring every $\frac{1}{freq}$ hour time points. The magnitude of capacity change between two consecutive time units $t'-1$ and t' is bounded by the ***maximum step size*** ($step$), with $M(t') - M(t'-1) \in [M(t'-1) - step, M(t'-1) + step]$. The ***dynamic range*** (dyn) of variable capacity is the range of machines over which capacity changes occur. We define this is as $dyn = ub - lb$, where ub and lb represent upper and lower bounds of variable capacity. The resource capacity at any time t is bounded by the dynamic range, $lb \leq M(t) \leq ub$.

Variable capacity gives rise to ***machine intervals***, where each interval $w_{j,i}$ corresponds to a contiguous block of time with length $w_{j,i}.len$, during which machine m_i is alive and available for scheduling. Machine intervals have length $w_{j,i}.len \in \{\frac{\ell}{freq} \mid \ell \in \mathbb{Z}, 1 \leq \ell \leq T \times freq\}$. The underlying capacity variation structure determines the set of machine intervals W, and the distribution of $w_{j,i}.len$.

Each machine interval $w_{j,i}$ would be at a certain point in time within its progression at time t. Let $w_{j,i}.uptime(t)$ be the ***uptime*** of the interval $w_{j,i}$ at time t. This is the amount of time in the interval $w_{j,i}$ that precedes time t, with $w_{j,i}.uptime(t) = t - w_{j,i}.start$. Let $w_{j,i}.remaining(t)$ denote the ***time remaining*** in the interval $w_{j,i}$ conditioned on uptime $w_{j,i}.uptime(t)$. This is the amount of time until the machine that the interval corresponds to is switched off under variable capacity with $w_{j,i}.remaining(t) = w_{j,i}.len -$

$w_{j,i}.uptime(t)$. The distribution of the time remaining in an interval $w_{j,i}$ varies with its uptime $w_{j,i}.uptime(t)$. Figure 3.1 depicts a machine interval at time t in its progression.

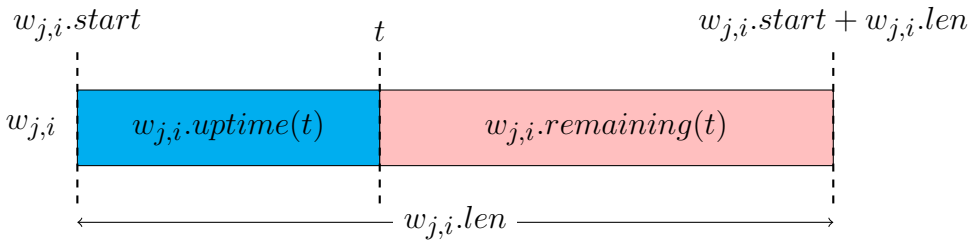


Figure 3.1: Graphical representation of a machine interval

Each job $j_k \in J$ has release time $j_k.release$, resource requirement $j_k.res$ and duration of $j_k.exec$. Jobs enter a FCFS queue upon arrival and are re-queued at the back if terminated. The baseline scheduler, FCFS/First-Fit (FF), iterates over available resources ordered on machine index i, and assigns the job to the first resource that has sufficient capacity to accommodate it, making scheduling decisions online. If terminated, the job is added to the back of the queue.

Table 3.2 summarizes the notation used in formally defining the scheduling problem.

Notation	**Description**
Variable Capacity Platform	
M	The set of machines
m_i	$m_i \in M$ represents a single machine
i	Index of machine m_i, which is a measure of the machine's relative risk of termination; $1 \leq i \leq \lvert M \rvert$
$m_i.res$	The resource capacity of machine m_i (cores/memory)
$M(t)$	The number of machines available during time unit t; $M(t) \leq \lvert M \rvert$
Dimensions of Variable Capacity	
$freq$	Change frequency, with a capacity changes occurring every $\frac{1}{freq}$ hr
$step$	The maximum step size between two consecutive capacity changes; $M(t) - M(t-1) \in [M(t-1) - step, M(t) + step]$

ub and lb	Upper and lower bounds of variable capacity; $lb \leq M(t) \leq ub$
dyn	Dynamic range of variable capacity; $dyn = ub - lb$.
Machine intervals	
$w_{j,i}$	A machine interval on machine m_i
$w_{j,i}.len$	Length of the machine interval $w_{j,i}$; $w_{j,i}.len \in \{\frac{\ell}{freq} \mid \ell \in \mathbb{Z}, 1 \leq \ell \leq T \times freq\}$
$w_{j,i}.start$	Start time of machine interval $w_{j,i}$. $(w_{j,i}.start + w_{j,i}.len)$ is the end time of interval $w_{j,i}$.
$w_{j,i}.uptime(t)$	Amount of time in machine interval $w_{j,i}$ that precedes time t; $w_{j,i}.uptime(t) = t - w_{j,i}.start$
$w_{j,i}.remaining(t)$	Time remaining in the machine interval $w_{j,i}$ conditioned on uptime $w_{j,i}.uptime(t)$; $w_{j,i}.remaining(t) = w_{j,i}.len - w_{j,i}.uptime(t)$
W	The set of machine intervals; $w_{j,i} \in W$
Workload	
J	The set of jobs
j_k	A single job, $j_k \in J$
$j_k.release$	The submit (i.e. release) time of job j_k
$j_k.res$	The resource requirement of job j_k
$j_k.exec$	The execution time of job j_k

Table 3.2: Notation for the formal definition of the scheduling problem

The scheduling objective is to minimize the number of job terminations, while limiting goodput loss to 2% and maintaining the scheduling latency distribution, compared to the baseline online First-fit scheduler. In practice, the constraint on goodput could be a tunable parameter, depending on the workload's requirements. Equation 3.1 formally defines this scheduling objective, denoting the improved scheduler as S and the baseline as FF.

$$\forall w_{j,i} \in W, \forall j_k \in J, \forall t \in T, \quad \min \left\{ \sum_{w_{j,i} \in W} \sum_{j_k \in J} \sum_{t \in T} X_{w_{j,i},j_k,t} \right\}$$

$$\text{where} \quad X_{w_{j,i},j_k,t} = \begin{cases} 1 & \text{if } t + j_k.\text{exec} > w_{j,i}.\text{start} + w_{j,i}.\text{len}, \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

$$\text{subject to} \quad goodput_{FF} - goodput_S \leq 2\%$$

$$\text{and} \quad latency_{FF} \sim latency_S$$

3.4 Experimental Methodology

We study representative samples of commercial cloud workloads from Microsoft Azure and Google's Borg, detailed in Section 2.2.2. We examine synthetic variable capacity platforms considering change frequencies of 0.25, 0.5, 1, 2, 4 and 8 changes/hour and step size values of 0.15, 0.3, 0.45 and 0.6. Dynamic range is constrained to 60%, since dynamic range impacts the number of intervals, but not machine interval characteristics. Capacity changes are modeled as random walks, with equal probability of capacity increase, decrease and no change in capacity between consecutive time periods. We examine observed variable capacity, based on a constant carbon budget, in the MISO, ERCOT, SPP and CAISO grids, over the representative month of April 2024 (previously discussed in Section 2.1.2).

As shown in Figure 3.2, we assume a 70% average available resource capacity in the platform, in order to model both upward and downward flexibility for variable capacity. Further, we size the workload to approximately 80% of the average resource capacity, which relates to the system's goodput. This results in an overall utilization of 56% in the platform, which is consistent with cloud utilization levels. For the Microsoft Azure workload, we model a cloud platform with 24,000 cores (1000 machines $\times$ 24 cores/machine), as exact machine information is not available. For Google's Borg workload, we rely on the associated machine

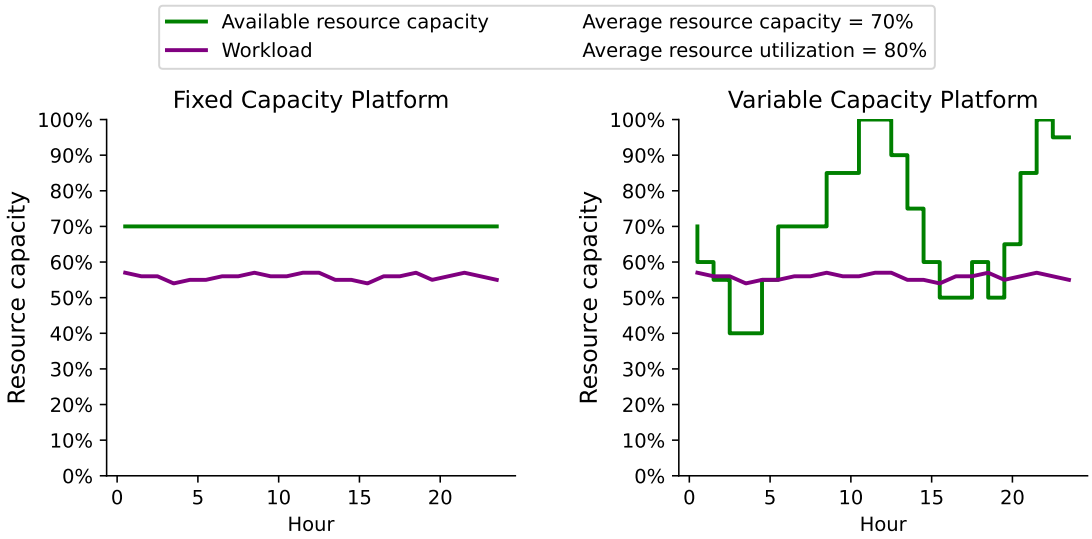


Figure 3.2: Configuration of resource capacity and utilization, in fixed and variable capacity platforms

trace [32], sampling 151.7 Normalized Compute Units (NCUs) comprising 200 machines.

The baseline scheduler is online First-Fit(FF), a commonly-used simplified approach to cloud job scheduling [7]. The FF scheduler iterates over available resources ordered on machine index i, and assigns the job to the first resource that has sufficient capacity to accommodate it. The following performance metrics are used to evaluate a scheduler's effectiveness on a workload. Goodput, the number of job terminations, and scheduling latency are key performance metrics, while the remaining metrics provide a holistic view of the system's performance.

1. ***Goodput***: The useful resource utilization of the platform. This is the ratio between the total computation of successful job completions to the total available resource capacity.

2. ***Number of job terminations***: The number of jobs terminated due to capacity loss.

3. ***Scheduling latency***: The time from job submission until the first start of a job, calculated over jobs that were scheduled. This is a measure of user experience.

4. ***Wasted resource fraction***: The fraction of resources that were wasted due to job

terminations. This is the ratio between the computation performed on terminated jobs to the total available resource capacity.

5. ***Idle resource fraction***: The fraction of platform resources that remain idle, after accounting for goodput and the wasted resource fraction. This is a byproduct of maintaining low scheduling latency.

6. ***Failure rate***: The ratio between the number of terminations and the number of jobs in the workload.

7. ***Jobs not scheduled***: The number of jobs in the workload that were unable to find a suitable resource to be scheduled on.

3.5 Summary

In this chapter, we describe our problem statement and scheduling approach. Under traditional scheduling, variable capacity leads to degraded workload performance. The underlying capacity variation structure determines the characteristics of machine intervals in a variable capacity platform, particularly the time remaining in an interval given its current point in its progression (uptime). Such information on machine intervals could potentially be exploited to improve variable capacity scheduling. Further, we formally define the scheduling framework and scheduling objective, forming the basis for following chapters. Finally, we detail our experimental setup, including workloads, variable capacity platforms and performance metrics.

CHAPTER 4

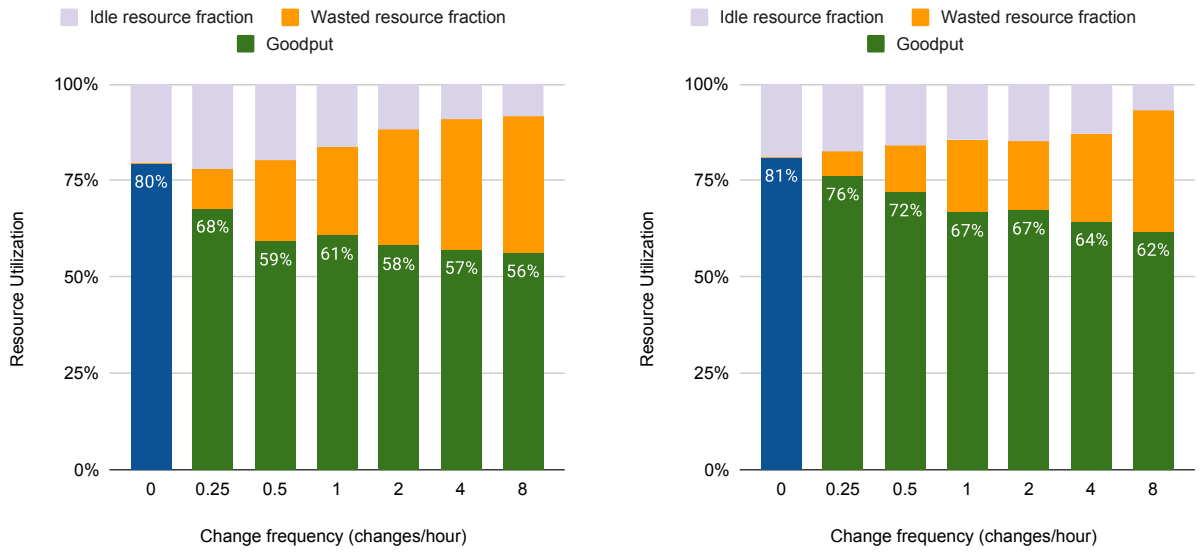
UNDERSTANDING WORKLOAD PERFORMANCE UNDER VARIABLE CAPACITY

In this chapter, we highlight the problem of variable capacity on cloud workload performance under traditional scheduling. Section 4.1 systematically investigates the impact of the synthetic variable capacity on commercial cloud workloads from Microsoft Azure and Google's Borg. In Section 4.2, we evaluate cloud workload performance under observed variable capacity, driven by variations in the power grid's carbon intensity. Section 4.3 examines a set of synthetic workloads, in order to understand the impact of workload properties on performance under variable capacity.

4.1 The Impact of Synthetic Variable Capacity on Cloud Workload Performance

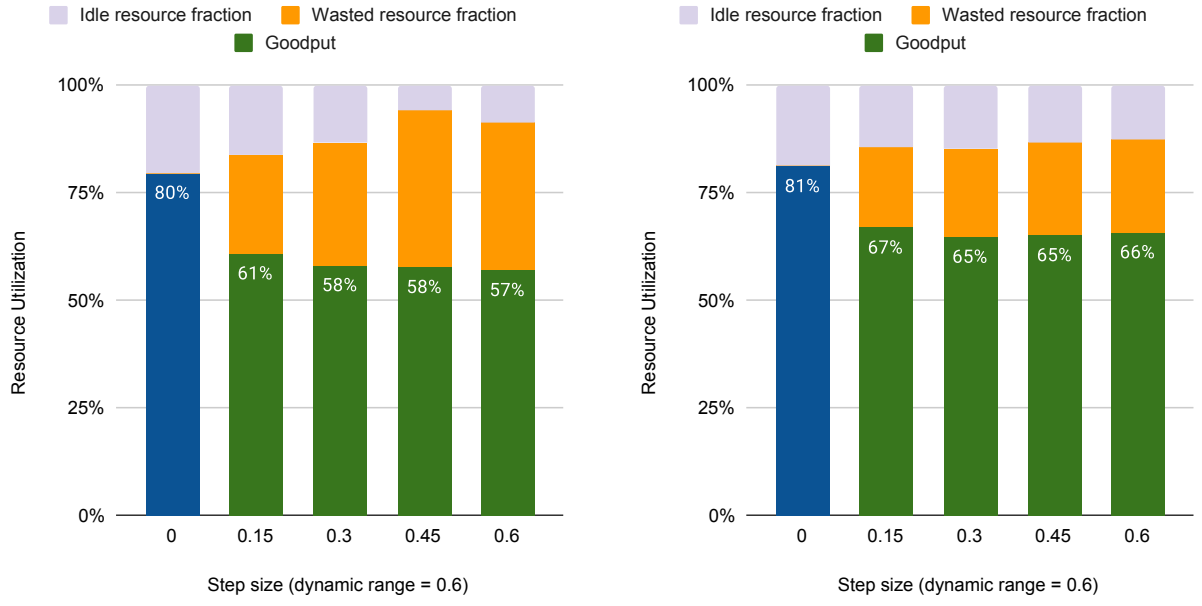
To systematically understand the impact of variable capacity on workload performance, we synthetically model the dimensions of variable capacity — step size and change frequency — and evaluate their impact of variable capacity on two commercial cloud workloads from Microsoft Azure and Google's Borg (detailed in Section 2.2.2), under traditional online First-Fit scheduling. We investigate the effects of variable capacity on the system's goodput, the number of job terminations, and the distribution of scheduling latency.

In both cloud workloads we observe a severe loss in goodput as capacity variation increases. In Figure 4.1, we evaluate the changes to the system's resource utilization as we successively increase the change frequency by 2×, while keeping the step size constant at 0.15. As the change frequency increases to 8 changes/hour, the Azure workload experiences a goodput loss of 24% (see Figure 4.1a) and the Borg workload experiences a goodput loss of 19% (see Figure 4.1b). In Figure 4.2, we evaluate the changes to the system's resource



(a) Azure (b) Borg

Figure 4.1: ***Goodput*** of cloud workloads against ***change frequency***, under traditional scheduling

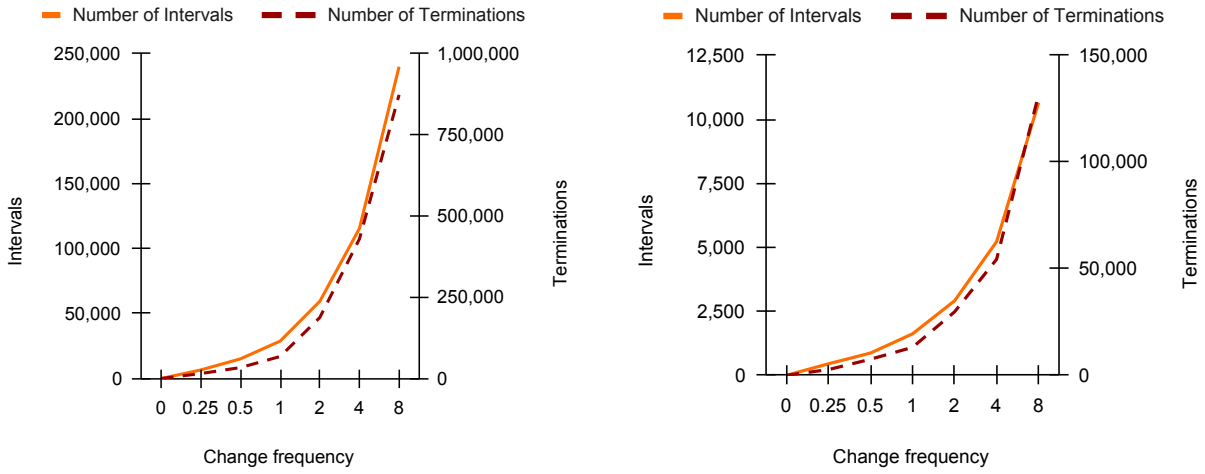


(a) Azure (b) Borg

Figure 4.2: ***Goodput*** of cloud workloads against ***step size***, under traditional scheduling

utilization as we successively increase the step size by 2×, while maintaining a change frequency of 1 change/hour. As the step size increases to 0.6, the Azure workload experiences a goodput loss of 23% (see Figure 4.2a) and the Borg workload experiences a goodput loss of 15% (see Figure 4.2b).

Goodput loss is a result of jobs being terminated in the face of variable capacity. To further understand this effect, we observe changes in the secondary resource utilization metrics, the wasted resource fraction and the idle resource fraction. The wasted resource fraction increases, and the idle resource fraction decreases with increasing variation. Therefore, it is evident that the higher goodput degradation from increasing variable capacity is a result of increasing wasted computation from job terminations. Due to its heavier-tailed in job duration distribution, the Azure workload experiences greater goodput loss than the less heavy-tailed Borg workload. This effect is further detailed in Section 4.3.

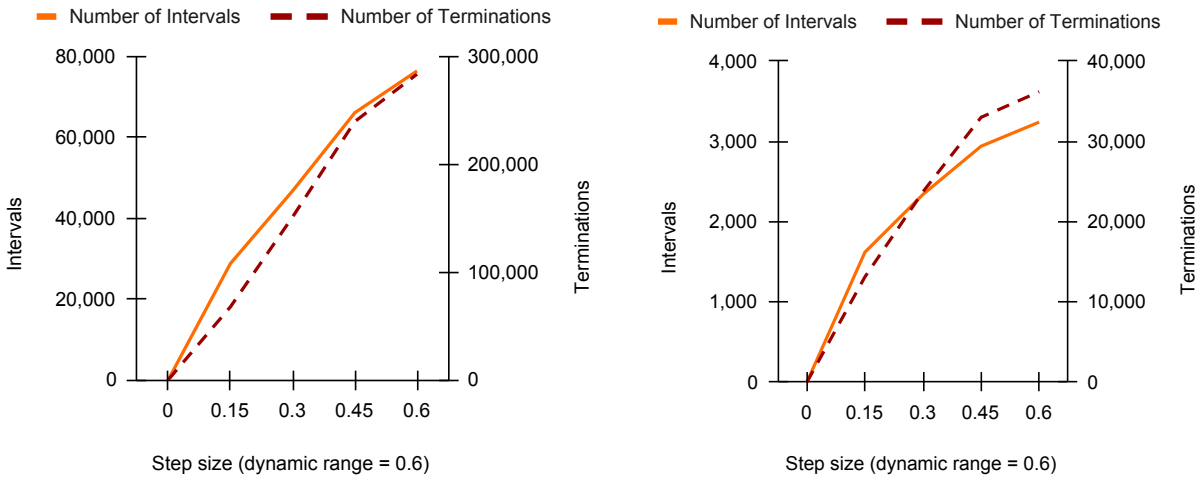


(a) Azure

(b) Borg

Figure 4.3: ***Job terminations*** in cloud workloads against ***change frequency***, under traditional scheduling

Figures 4.3 and 4.4 show the corresponding number of job terminations (right y-axis) and the number of machine intervals under the variable capacity profile (left y-axis), with increasing change frequency and step size. We observe that the number of jobs terminations is



(a) Azure

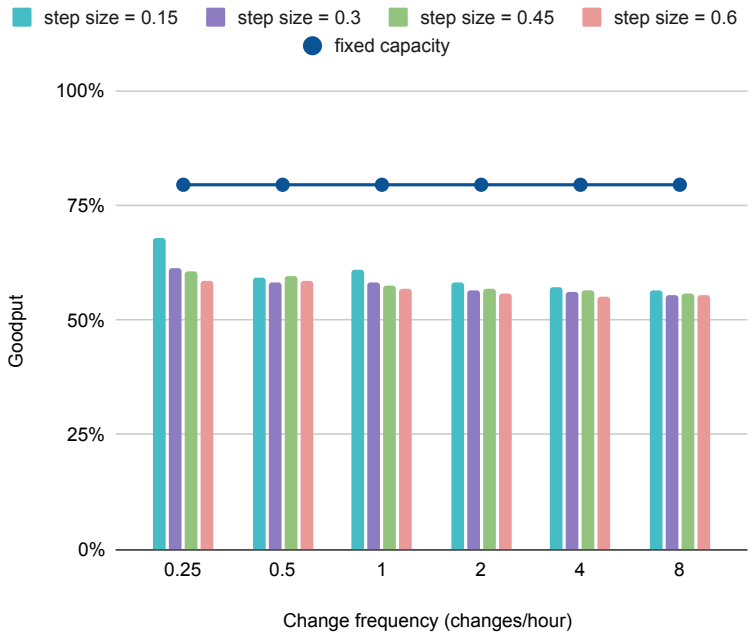
(b) Borg

Figure 4.4: ***Job terminations*** in cloud workloads against ***step size***, under traditional scheduling

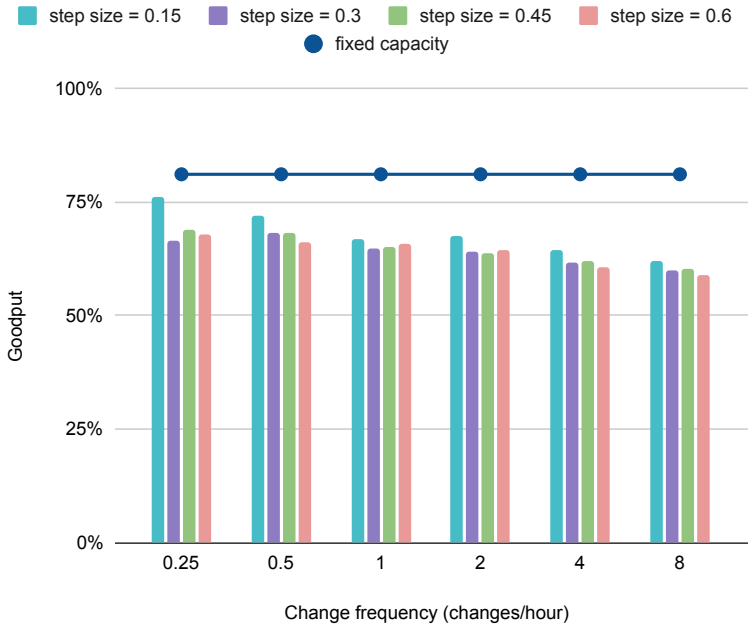
proportional to the number of intervals under the variable capacity profile in both workloads, since interval ends lead to job terminations.

We further evaluate the impact of variable capacity to cloud workload performance over a more comprehensive range of change frequency and step size parameters in Figures 4.5 and 4.6. In line with our previous conclusions, we observe that goodput loss increases as the change frequency and step size increases. Further, the number of job terminations follow the number of intervals in the variable capacity profile.

We conclude that different variable capacity profiles produce varying machine interval counts, which determine the number of job terminations and the resulting goodput. Irrespective of the variable capacity parameters, variable capacity profiles that produce the same number of intervals, have the same impact on job terminations and goodput. For instance, in the Borg workload, the variable capacity profile of 2 changes/hour with step size of 0.15 produces 2,910 intervals and 29,749 job termination with a goodput of 67%. In comparison, the variable capacity profile of 1 change/hour with step size of 0.45 produces 2,940 intervals and 33,001 job termination with a goodput of 65%.

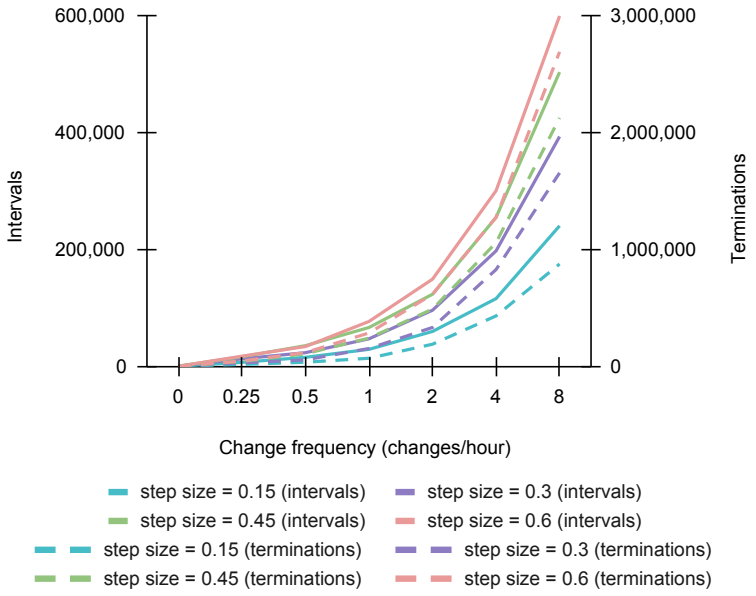


(a) Azure

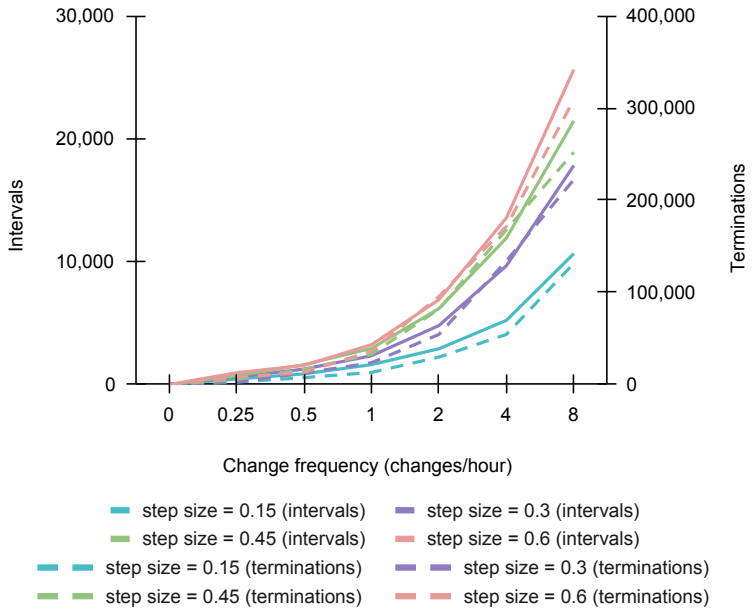


(b) Borg

Figure 4.5: ***Goodput*** of cloud workloads across the full spectrum of synthetic variable capacity, under traditional scheduling

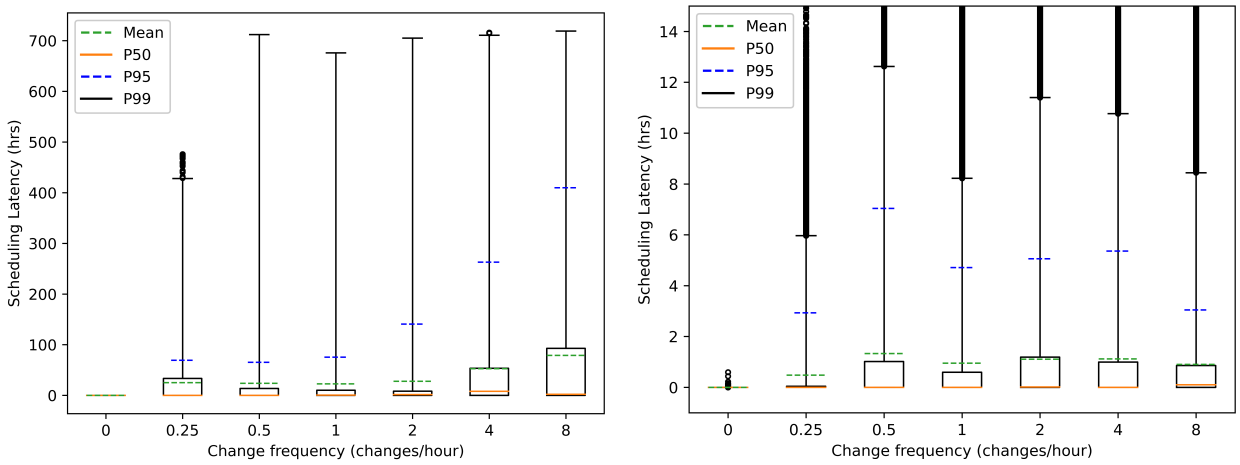


(a) Azure



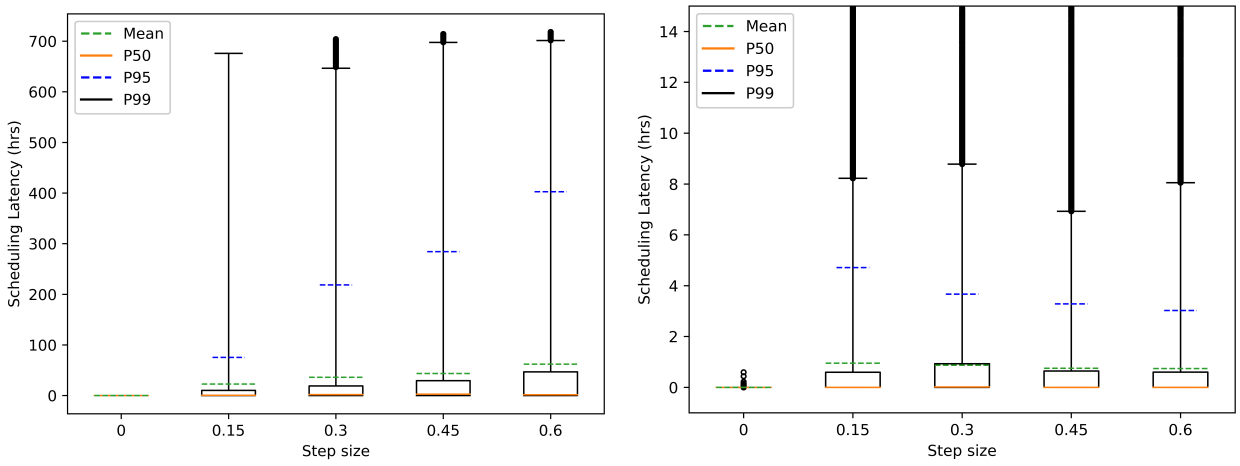
(b) Borg

Figure 4.6: ***Job terminations*** in cloud workloads across the full spectrum of synthetic variable capacity, under traditional scheduling



(a) Azure (b) Borg

Figure 4.7: Distribution of ***scheduling latency*** in cloud workloads against ***change frequency***, under traditional scheduling



(a) Azure (b) Borg

Figure 4.8: Distribution of ***scheduling latency*** in cloud workloads against ***step size***, under traditional scheduling

In addition to the system's goodput and job terminations, increasing capacity variation also impacts scheduling latency. In Figures 4.7 and 4.8, we show the distribution of scheduling latency with increasing change frequency and step size. We observe that variable capacity increases scheduling latency. However, the overall impact to scheduling latency is lower in the less heavy-tailed Borg workload (Figures 4.7b and 4.8b) than the heavier-tailed Azure workload (Figures 4.7a and 4.8a). Further, in the heavier-tailed Azure workload, the P95 tail latency increases with higher variable capacity. Due to the Azure workload's heavier-tail, jobs experience higher scheduling latency than the less heavy-tailed Borg workload.

Insights:

- As the change frequency and step size increase, goodput decreases, the number of intervals increase and the number of job terminations increase.
- Different variable capacity profiles produce varying interval counts, which determine the number of job terminations and the resulting goodput.
  - Irrespective of the variable capacity parameters, variable capacity profiles that produce the same number of intervals have a similar impact on job terminations and goodput.
- Variable capacity increases scheduling latency. Heavier-tailed workloads experience higher scheduling latency under variable capacity.

4.2 The Impact of Observed Variable Capacity on Cloud Workload Performance

In the previous section, we considered workload performance under synthetic variable capacity modeled on real variation properties, with the objective of systematically evaluating the

impacts of variable capacity on workload performance. In this section, we extend this work to observed variable capacity, driven by variations in power grids' carbon intensity. This section presents a case study of scheduling under variable capacity, aiming to understand the effect of the underlying variable capacity source's properties on workload performance.

As previously described in Section 2.1.2, we consider data centers located in four power grids with distinct characteristics — MISO, ERCOT, SPP, and CAISO, ordered on a spectrum of increasing coefficient of variation in carbon intensity — and evaluate the impact of variable capacity under a constant carbon budget on the Azure workload's performance, over the representative month of April 2024.

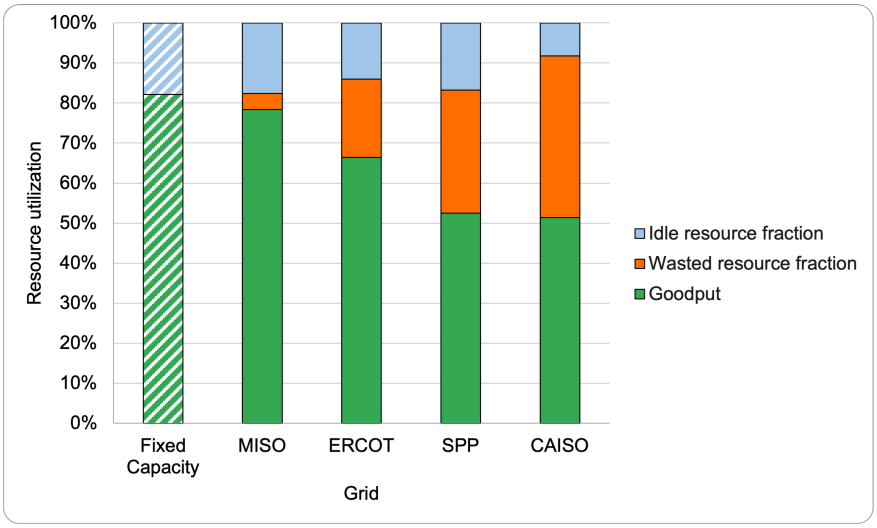
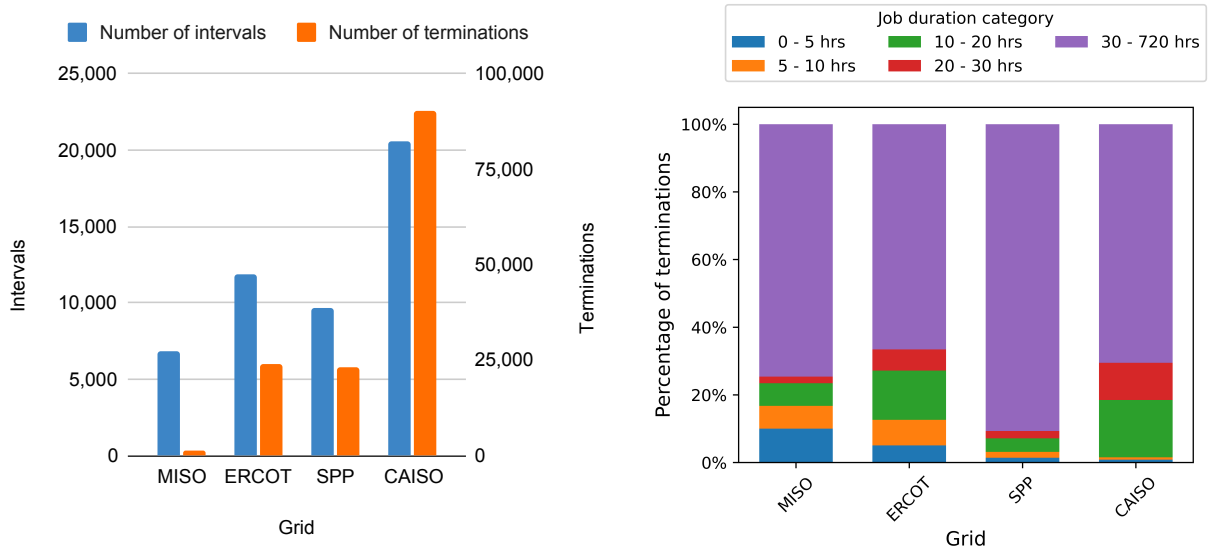


Figure 4.9: ***Goodput*** of the Azure workload with observed variable capacity, under traditional scheduling

We evaluate the Azure workload's performance under observed variable capacity in Figures 4.9 and 4.10. We see the highest goodput loss and the highest number of job terminations under variable capacity in CAISO. The higher renewable penetration in CAISO and the resulting high capacity variation over the larger dynamic range leads to a higher number of intervals. This, in turn, results in a higher number of job terminations and greater goodput loss. In contrast, MISO's lower renewable penetration produces a smaller dynamic range, a lower number of intervals, and, thus, lower goodput loss than CAISO.



(a) Number of intervals and job terminations

(b) Drilldown on terminated job durations

Figure 4.10: ***Job terminations*** in the Azure workload with observed variable capacity, under traditional scheduling

The underlying generation sources of the power grid impact workload performance under observed variable capacity. This is evident in the performance metrics of ERCOT and SPP. In Figure 4.10a, we observe a similar number of terminations across ERCOT and SPP. However, Figure 4.10b shows that the mix of the terminated job durations vary substantially across the two grids. Under variable capacity in ERCOT, 33% of terminations are from jobs with duration less than 30 hours. In SPP, this fraction is much lower, at 9%. Due to the longer periodicity of wind generation, the wind-heavy SPP grid has longer intervals. Therefore, in SPP, shorter jobs could complete successfully within its intervals, disproportionately affecting longer jobs. Although the number of terminations are relatively similar, in Figure 4.9, we observe higher goodput loss in SPP, which is a result of the larger fraction of long job terminations. The effects of the underlying generation sources on machine intervals are discussed in detail in Section 5.1.

Insights:

- Grids with higher coefficient of variation in carbon intensity experience a greater impact to workload performance under variable capacity.
  - Higher coefficient of variation in carbon intensity in the power grid leads to increased capacity variation and a higher number of intervals, resulting in greater goodput loss.
- The underlying generation sources of the power grid impacts the distribution of terminated job durations, and thus, goodput under variable capacity.

4.3 The Impact of Workload Characteristics on Performance under Variable Capacity

Prior sections evaluated the performance of real cloud workload traces under variable capacity. In this section, we seek to understand the impact of workload characteristics on performance under variable capacity. We construct and evaluate a set of synthetic workloads, considering the dimensions of job duration distribution and job resource size.

4.3.1 Job Duration Distribution and its Impact on Workload Performance

In order to understand the impact of the job duration distribution on workload performance under variable capacity, we construct a set of workloads in steady state with varying job duration distributions. The distribution of job durations in cloud workload is heavy-tailed [19, 64, 34]. Therefore, we model the job durations based on a Zipf distribution [52], with varying skew parameters. In a Zipf distribution, a lower skew parameter produces a heavier tail. Thus, a workload with a lower Zipf skew parameter has a higher fraction of longer jobs.

Table 4.1 shows the statistics of the set of heavy-tailed synthetic workloads with varying

Workload	Zipf skew	Mean (hrs)	P50	P90 (hrs)	P95 (hrs)
Zipf-1.8	1.8	1.51	5 mins	0.92	2.17
Zipf-1.7	1.7	2.90	10 mins	1.33	3.67
Zipf-1.6	1.6	5.43	10 mins	2.25	7.50
Zipf-1.5	1.5	11.73	10 mins	5.00	20.70
Zipf-1.4	1.4	26.07	15 mins	15.00	92.97
Zipf-1.3	1.3	58.86	25 mins	99.05	720.00
Zipf-1.2	1.2	123.24	1.25 hrs	720.00	720.00
Zipf-1.1	1.1	288.09	30.75 hrs	720.00	720.00

Table 4.1: Statistics of job duration in the set of synthetic heavy-tailed workloads with increasing skew in job duration

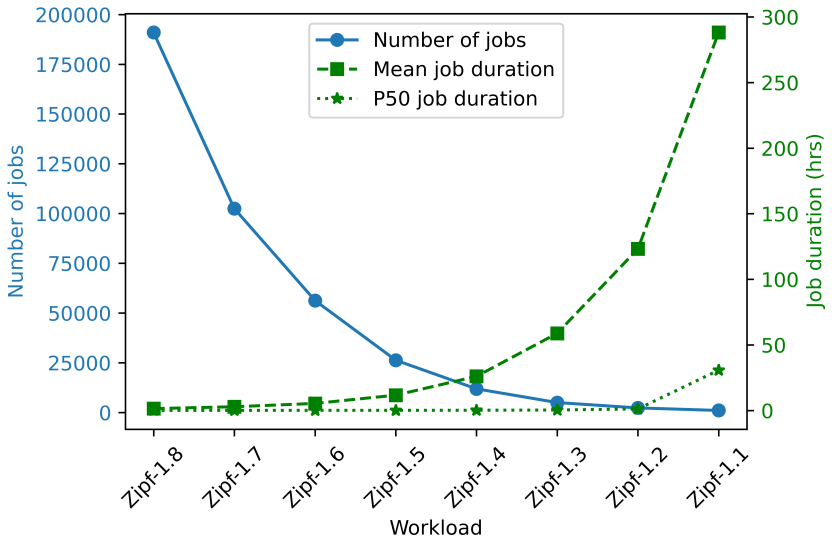


Figure 4.11: Workload characteristics with increasing skew in job duration distribution

skew parameters. In each successive workload, the mean job duration increases by approximately 2× (1.87× – 2.34×). Figure 4.11 shows the interplay between job duration and the number of jobs in the workload. As the workload becomes more heavy-tailed, the fraction of longer jobs in the workload increases, as evidenced by the mean and P50 job duration. Consequently, the number of jobs in the workload decreases.

It is important to note that the Azure workload evaluated in previous sections exhibits the job duration statistics of the synthetic workload with skew parameter of 1.3, in terms of mean and P50 job duration. In contrast, the Borg workload has much shorter jobs and displays the properties of the synthetic workload with a skew of 1.8. Therefore, the results in this section closely tie with our observations in Section 4.1, based on real cloud workload performance under variable capacity.

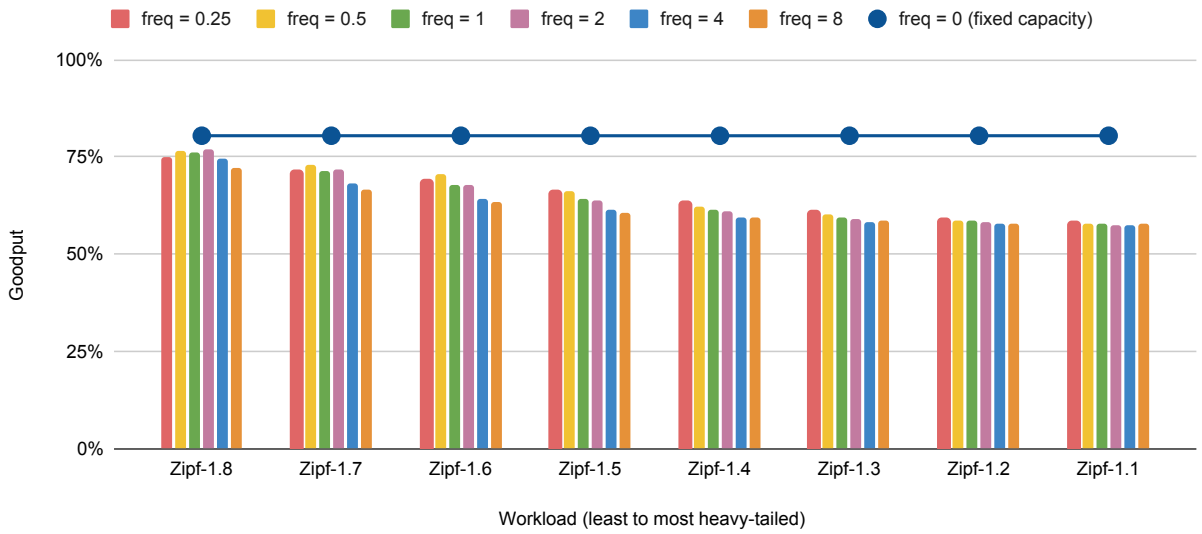


Figure 4.12: ***Goodput*** of synthetic workloads with increasing skew in job duration, against ***change frequency***, under traditional scheduling

Figures 4.12 and 4.13 shows the impact to goodput across the set of heavy-tailed workloads, under increasing change frequency and step size. We observe that heavier-tailed workloads experience greater goodput loss under variable capacity.

In Figures 4.14a and 4.14b, we evaluate the impact of the workload's job duration distri-

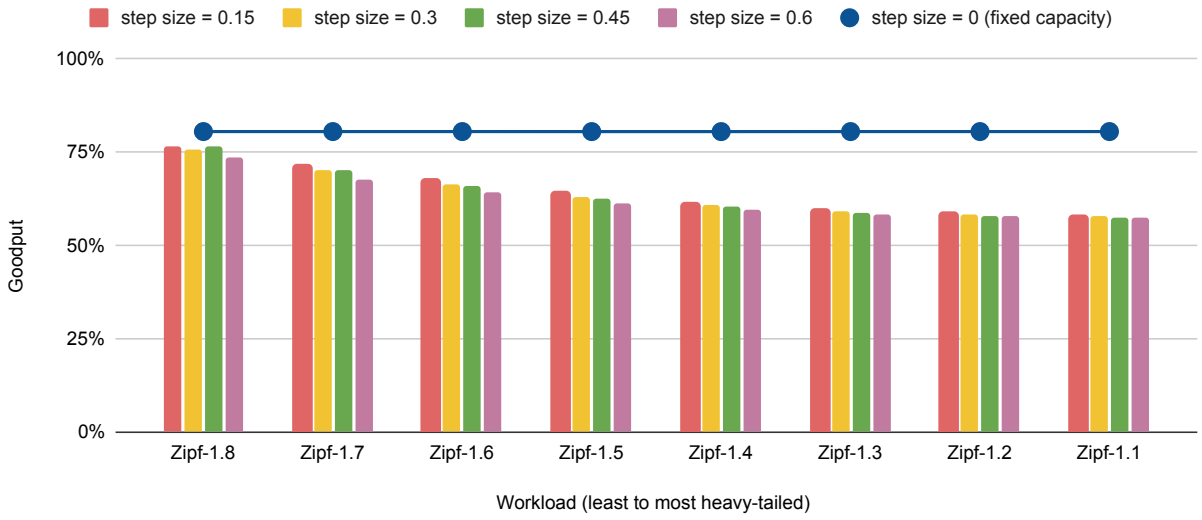
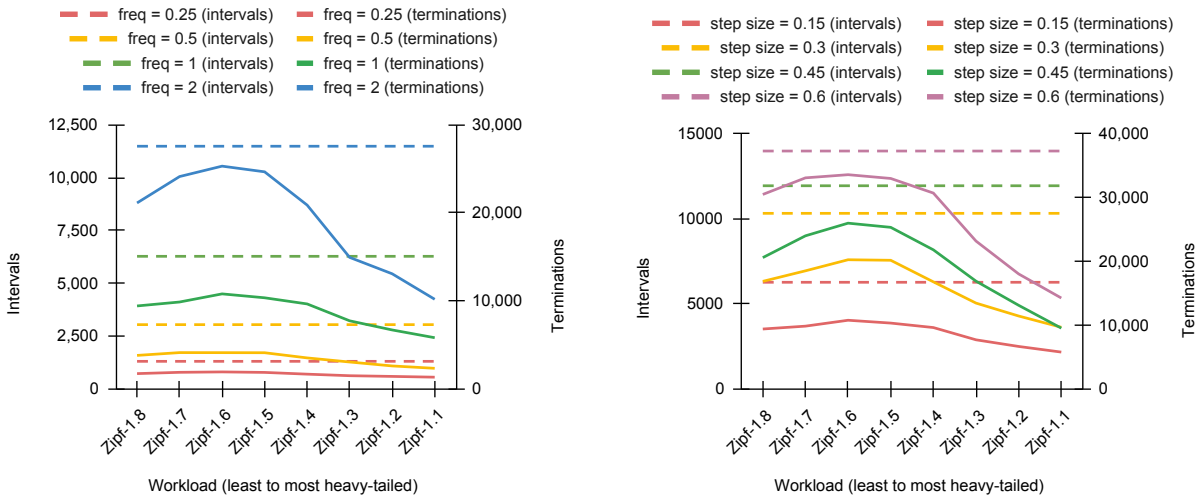


Figure 4.13: ***Goodput*** of synthetic workloads with increasing skew in job duration, against ***step size***, under traditional scheduling

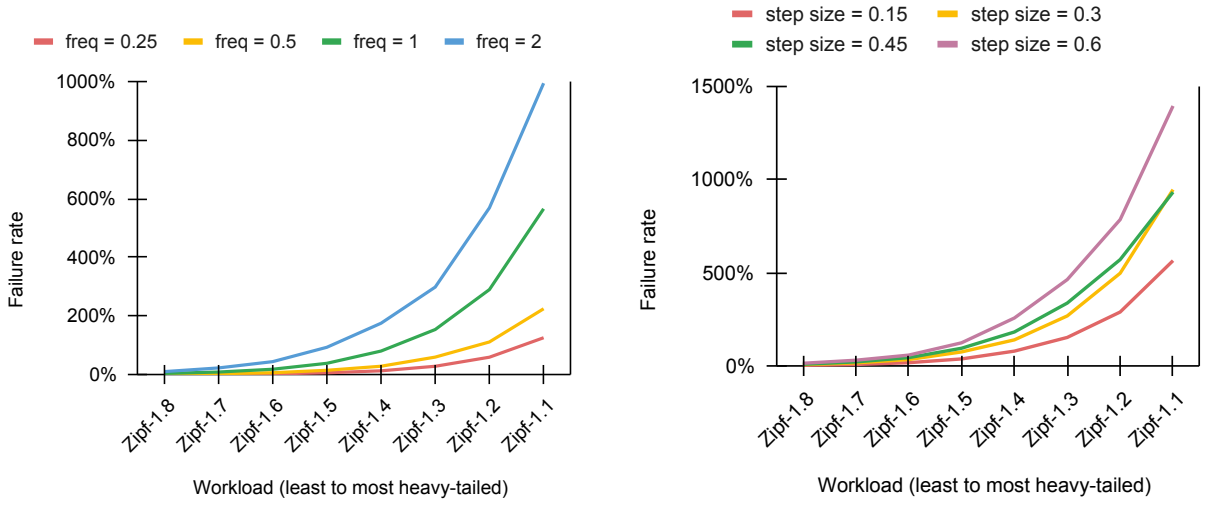


(a) Change frequency

(b) Step size

Figure 4.14: ***Job terminations*** in synthetic workloads with increasing skew in job duration, against ***change frequency*** and ***step size***, under traditional scheduling

bution on the number of job terminations under increasing change frequency and step size, respectively. We observe that the number of intervals remains constant across the workloads, and increases with the change frequency. At lower change frequency, the number of terminations remain relatively constant, since terminations are correlated with the number of intervals. However, as the change frequency and step size increases, the number of terminations increase for less heavy tailed workloads. This is because the time between capacity changes become smaller, increasing the likelihood of shorter jobs in less heavy-tailed workloads being terminated within shorter intervals. 4.14b shows similar results with increasing step size. It's important to note that although terminations decrease for heavier-tailed workloads, the overall impact to resource utilization is much higher. As heavier tailed workloads contain a smaller number of relatively longer jobs, a termination in a heavier-tailed workloads results in greater wasted resources and subsequent goodput loss.



(a) Change frequency

(b) Step size

Figure 4.15: ***Job failure rate*** of synthetic workloads with increasing skew in job duration, against ***change frequency*** and ***step size***, under traditional scheduling

We evaluate the impact of workload characteristics to the failure rate under increasing skew in job duration in Figure 4.15. As the mean job duration increases by approximately

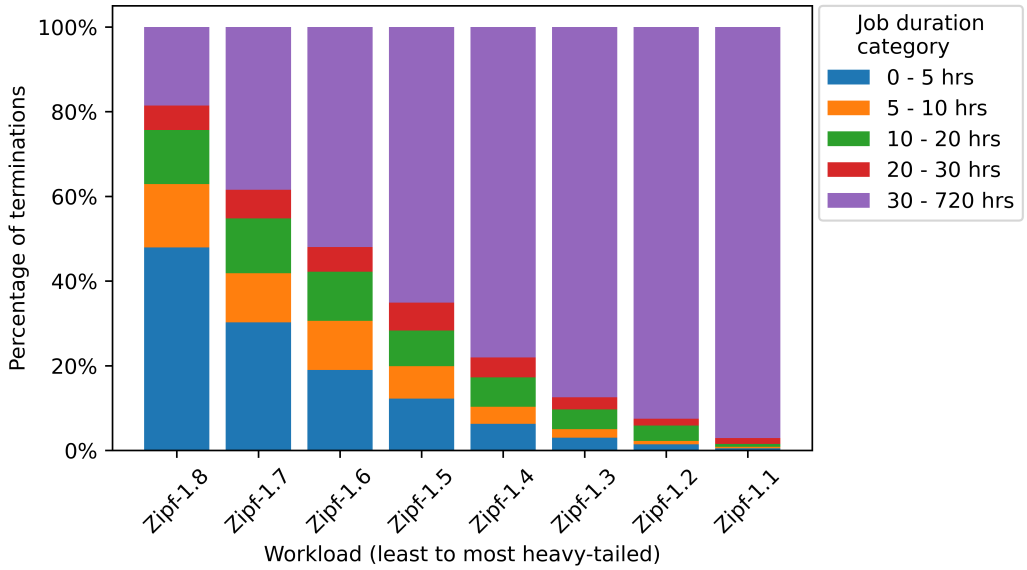


Figure 4.16: Drilldown on ***job terminations*** in synthetic workloads with increasing skew in job duration (change frequency = 1, step size = 0.15)

2×, the job failure rate increases to 1.94× on average (1.48× – 2.17×) across the range of change frequencies, and to 1.93× (1.63× – 2.17×) across the range of step sizes. This is a result of the decreasing number of jobs in heavier-tailed workloads. Therefore, reducing the number of job terminations would disproportionately benefit heavier-tailed workloads. We drill down on terminated job durations across the set of heavy-tailed workloads in Figure 4.16, with change frequency 1 change/hour and step size 0.15. The higher fraction of longer jobs in heavier-tailed workloads results in a higher fraction of long job terminations, leading to greater goodput loss.

In Figure 4.17, we evaluate scheduling latency with increasing skew in the job duration distribution, with change frequency 1 change/hour and step size 0.15. As the mean job duration increases by approximately 2×, the mean scheduling latency increases to 1.46× on average and the P99 scheduling latency increases to 1.56× on average.

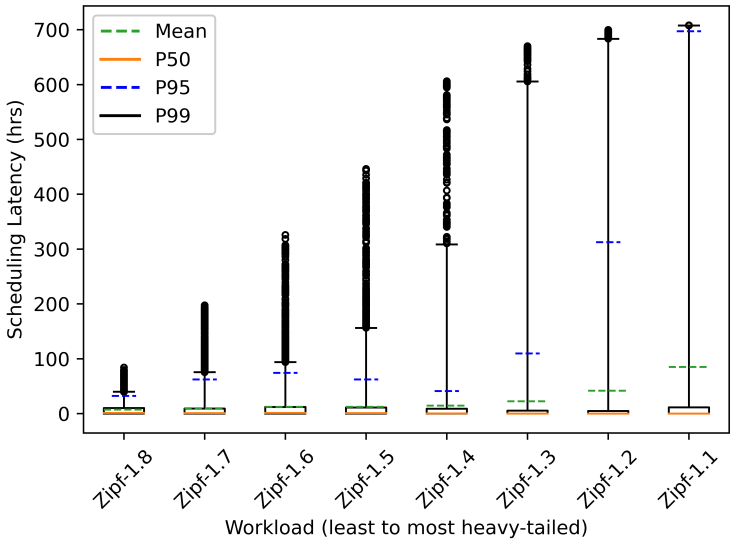


Figure 4.17: ***Scheduling latency*** of synthetic workloads with increasing skew in job duration (change frequency = 1, step size = 0.15)

4.3.2 Job Resource Size and its Impact on Workload Performance

In this section, we evaluate the impact of the workload's job resource size, on performance under variable capacity.

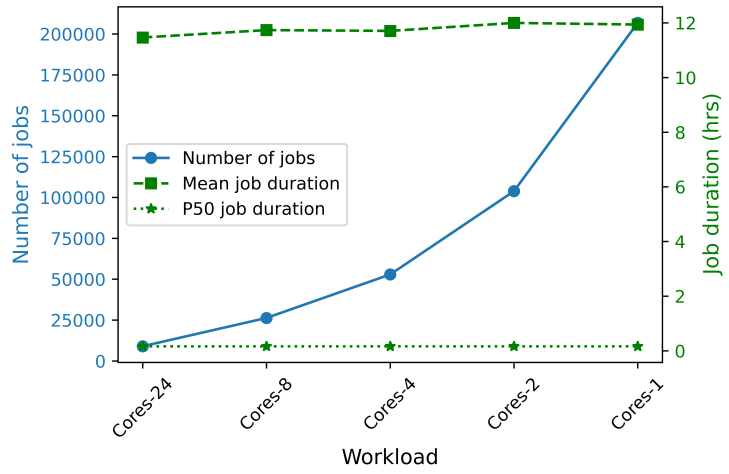


Figure 4.18: Workload characteristics with decreasing job resource size

Figure 4.18 shows the characteristics of the evaluated set of workloads. For simplicity, we assume that all jobs in the workload have constant compute resource sizes (cores). As we

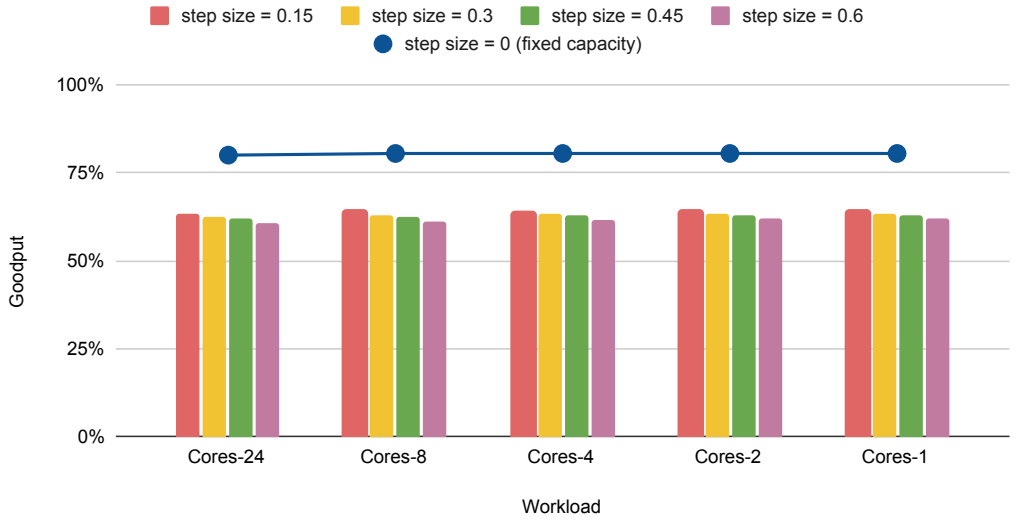
decrease the number of cores allocated to each job, the number of jobs increases proportionately, effectively distributing the workload across multiple jobs. We maintain a heavy-tailed job duration distribution with a skew parameter of 1.5 across the workloads. Thus, the mean and P50 job duration across the workloads remain relatively constant.

In Figure 4.19, we observe that as change frequency and step size increases, the goodput loss across workloads with decreasing resource sizes remains the same. This is because a similar amount of work lost at interval ends across the workloads, as a result of similar job duration distributions. However, the number of job terminations increase with decreasing resource size, as a single interval end would correspond to the termination of a higher number of jobs (see Figure 4.20).

We observe higher failure rates in workloads with higher job resource size, because of the lower number of jobs (Figure 4.21). Workloads with larger job resource sizes consist of a smaller number of larger jobs. Due to the effects of the smaller number of jobs in the workload, the failure rate is higher in such workloads. Thus, reducing job terminations would disproportionately benefit workloads with larger job resource sizes.

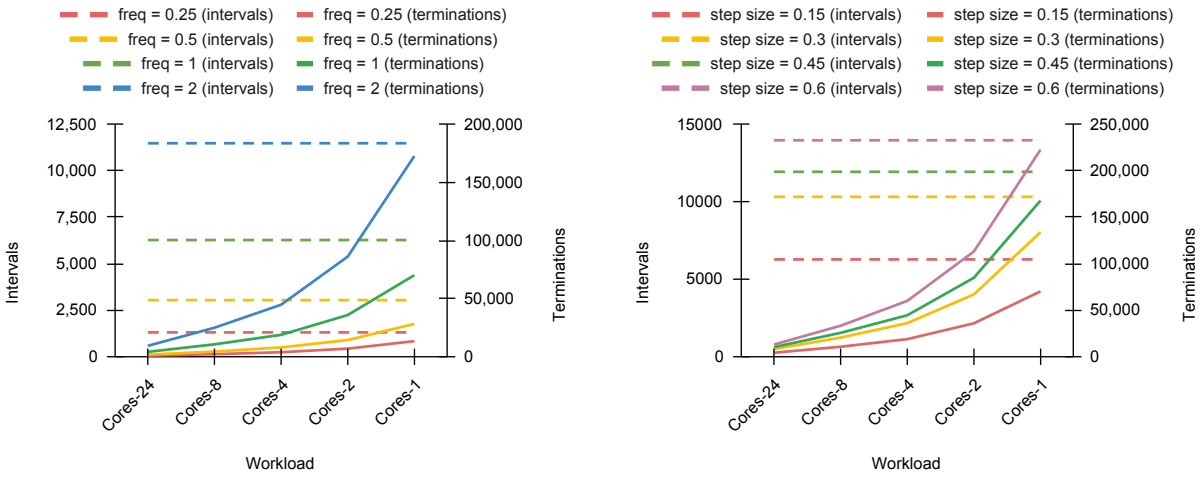
There is very limited impact on scheduling latency with decreasing job resource size. Figure 4.22 shows that the mean scheduling latency remains relatively constant between $0.84\times$ – $1.06\times$ and the P99 scheduling latency remains relatively constant between $0.91\times$ – $1.02\times$, across workloads with varying job resource sizes.

(a) Change frequency



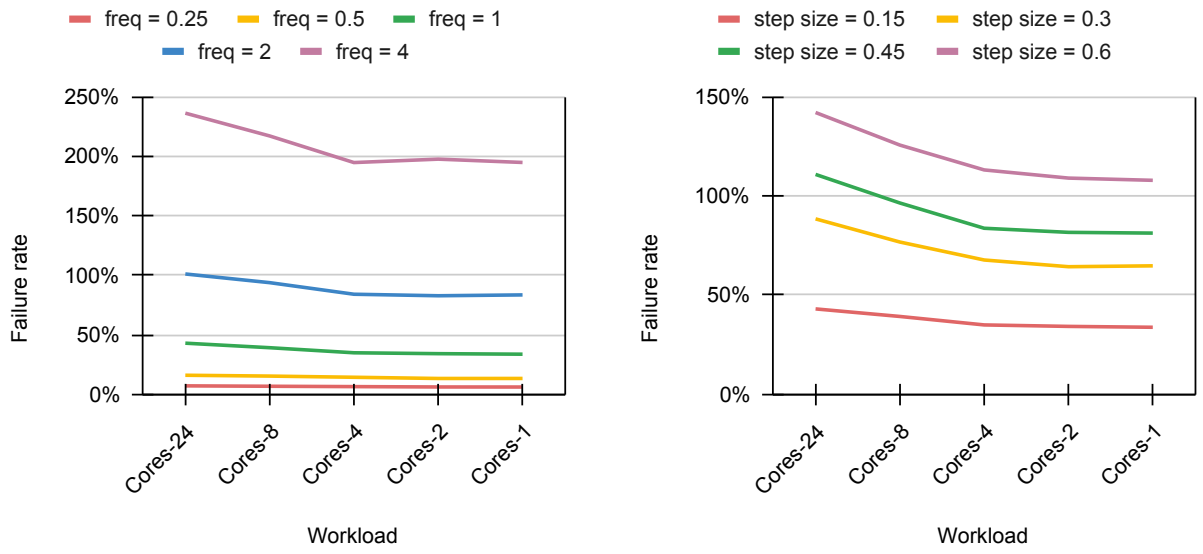
(b) Step size

Figure 4.19: ***Goodput*** of synthetic workloads with decreasing job resource sizes, against ***change frequency*** and ***step size***, under traditional scheduling



(a) Change frequency (b) Step size

Figure 4.20: ***Job terminations*** in synthetic workloads with decreasing job resource sizes, against ***change frequency*** and ***step size***, under traditional scheduling



(a) Change frequency (b) Step size

Figure 4.21: ***Failure rate*** of synthetic workloads with decreasing job resource sizes, against ***change frequency*** and ***step size***, under traditional scheduling

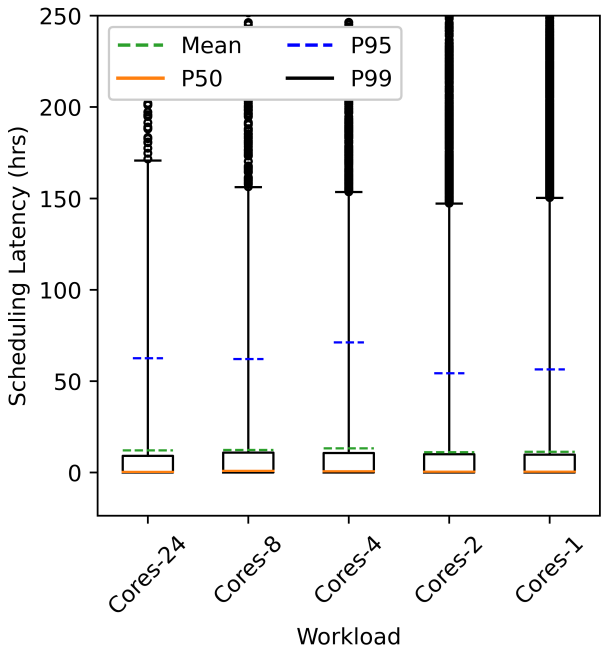


Figure 4.22: ***Scheduling latency*** of synthetic workloads with decreasing job resource size (change frequency = 1, step size = 0.15)

Insights:

- Heavier-tailed workloads experience greater goodput loss under variable capacity.
- As the mean job duration increases by approximately 2×, the job failure rate increases to 1.93× on average, across the range of change frequencies and step sizes.
  - Reducing the number of job terminations would disproportionately benefit heavier-tailed workloads.
- As the mean job duration increases by approximately 2×, the mean scheduling latency increases to 1.46× on average and the P99 scheduling latency increases to 1.56× on average.
- Workloads with larger job resource sizes experience higher failure rate under variable capacity.

4.4 Job Terminations: A Crucial Workload Performance Metric for Variable Capacity

We summarize the fundamental implications of variable capacity on workload performance as,

- Job terminations caused by intervals ends
- Goodput loss as a result of wasted computation from job terminations
- Delay in job scheduling (increased scheduling latency) due to lost capacity

In this section, we posit job terminations as the crucial performance metric for job scheduling under variable capacity. We advocate for the community to use job terminations as the primary scheduling objective for cloud workloads under variable capacity. We further discuss the implications of checkpointing on goodput accounting and overall conclusions.

4.4.1 Implications of Job Terminations in the Cloud Service Model

From a job scheduler's point of view, variable capacity leads to capacity losses during which certain jobs running on the platform would be terminated. We identify consequences of job terminations to cloud providers below:

- ***Lost application state***: In the cloud workload model, jobs typically refer to VMs that host continuous services. Disruptions to these VMs would lead to losses in the application's state on several fronts, including lost storage connections, network connections, authentication, REST API connections, currently in-flight operations, and the overall loss of the service's continuity. A cost-constrained application developer might lack the necessary redundancy and transactional safeguards deployed, leading to substantial losses from disruption of service. Even for an optimized application, this loss is non-zero and could result in loss of service continuity.

- ***SLO violations***: The reliability of cloud services is measured in terms of monthly uptime percentage, as specified by their Service Level Objectives (SLO) [16, 57, 47]. Job terminations under variable capacity could result in SLO violations through loss of service. In the event of an SLO violation, the cloud customer could be entitled to a service credit on the cloud service [4, 26]. Therefore, there is financial incentive for cloud providers to avoid job terminations.

- ***Impact to user experience***: Cloud providers host a range of user-facing applications within their workload. The compute resource utilization of interactive applications is 28% in Microsoft Azure [19] and 30% in Google's Borg [64]. Job terminations caused by variable capacity would be felt by the end users of these user-facing applications.

- ***Checkpointing overhead***: HPC applications rely on checkpointing for resilience under system failures [6, 20]. A similar checkpoint/recovery model could be used to reduce the amount of work wasted under variable capacity. However, even with optimal checkpointing, the overhead of checkpointing is non-zero. Checkpointing, by itself, would have implications on the system's goodput, as further discussed in Section 4.4.2.

Based on our analysis above, we identify job terminations as the crucial performance metric for cloud workload scheduling under variable capacity.

4.4.2 Implications of Checkpointing on Goodput Accounting

In calculating the impact to resource utilization from job terminations throughout this document, we follow Equation 4.1. We calculate the wasted computation from a job termination as the total work lost from the start of the job until its termination.

For job j_k, let $j_k.res$ denote its resource requirement, $j_k.start$ the last started time and $j_k.terminate$ the time of its last termination.

$$\text{wasted computation} = j_k.res \times (j_k.terminate - j_k.start) \tag{4.1}$$

An extended goodput accounting methodology might consider checkpointing, the de-facto standard technique for resilience in HPC [21, 38]. Adapting this model to cloud services, we could consider checkpointing as a constant tax on the job, to account for the overhead associated with continuously maintaining the state of every job. Therefore, job duration should be re-calibrated for the job slowdown resulting from checkpointing. Checkpointing allows a job to restart from its last checkpoint, thereby limiting wasted work to the computation performed since the last checkpoint and some recovery time. Wasted work from job terminations under a checkpointing model can be formulated as Equations 4.2 and 4.3. Let $j_k.exec$ denote job j_k's execution time, C the system's checkpointing overhead, R recovery time and t' the last checkpoint time.

$$\text{job duration under checkpointing} = j_k.exec \times (1 + C) \tag{4.2}$$

$$\text{wasted computation} = j_k.res \times (j_k.terminate - t' + R) \tag{4.3}$$

The impact to resource utilization under checkpointing would be drastically different from the currently considered goodput accounting methodology. The amount of wasted work from job terminations and resulting goodput loss could be substantially lower, since checkpointing allows a large fraction of computation performed before termination to be recovered. However, the cost of checkpointing continuous cloud applications could be prohibitive. Thus, for checkpointing to be feasible, the goodput loss from checkpointing must be lower than the expected goodput loss from capacity change.

While we note the dependency of our evaluation of resource utilization under variable capacity on the goodput accounting methodology, our overall conclusions remain fundamentally unchanged. Goodput is an important performance metric since system utilization translates to the platform's ability to effectively sell compute resources. However, the cloud is not goodput-optimized, as indicated by its lower resource utilization levels [64, 19] that are required to maintain lower scheduling latency and prevent SLO violations. Therefore, in the cloud service model, job terminations are a more crucial performance metric, given its direct impact on SLO violations.

4.5 Summary

In this chapter, we observed that as underlying capacity variation increases, both goodput loss and job terminations increase. We identified that interval ends cause job terminations, as evidenced by the proportionality between job terminations and the number of intervals. We further evaluated workload performance under observed variable capacity, driven by variations in the power grid's carbon intensity. The characteristics of the underlying generation sources in the power grid, including variation and sources of generation, impact workload performance under variable capacity. We observed that workload with heavier-tailed distribution in job duration and larger resource sizes experience greater performance loss under variable capacity. Finally, we identified job terminations as the crucial performance metric for cloud resource management under variable capacity.

CHAPTER 5

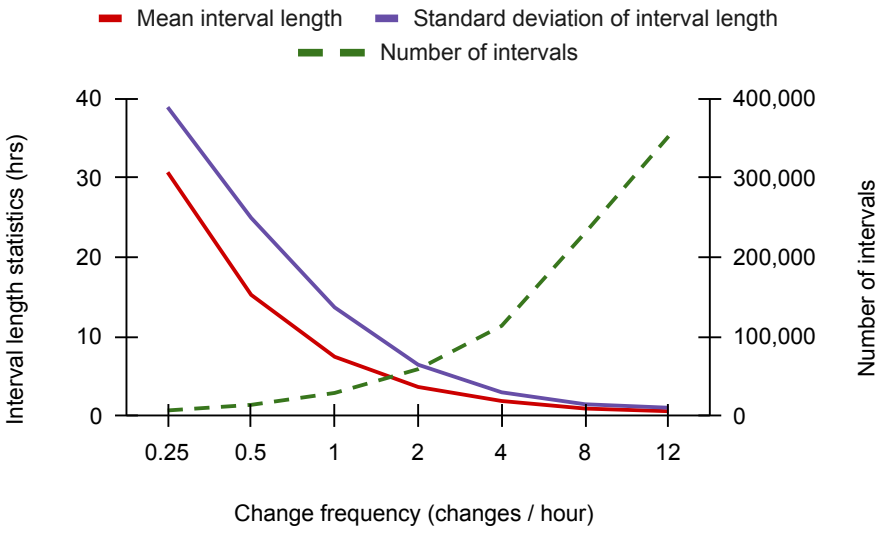
UNDERSTANDING MACHINE INTERVAL CHARACTERISTICS UNDER VARIABLE CAPACITY

In Chapter 4, we observed that machine intervals play a key role in workload performance under variable capacity, affecting job terminations, the system's goodput and scheduling latency. In this chapter, we explore how the underlying capacity variation structure affects machine intervals and build an understanding of interval properties are potentially exploitable to improving scheduling under variable capacity. In Section 5.1, we seek to understand how the underlying capacity variation structure affects machine interval length, across the multi-dimensional space of synthetic variable capacity as well as under observed variable capacity. Section 5.2 evaluates the impact of variable capacity on the time remaining in an interval. In Section 5.3, we identify machine intervals characteristics that could potentially predict interval time remaining, allowing better job-to-interval alignment.

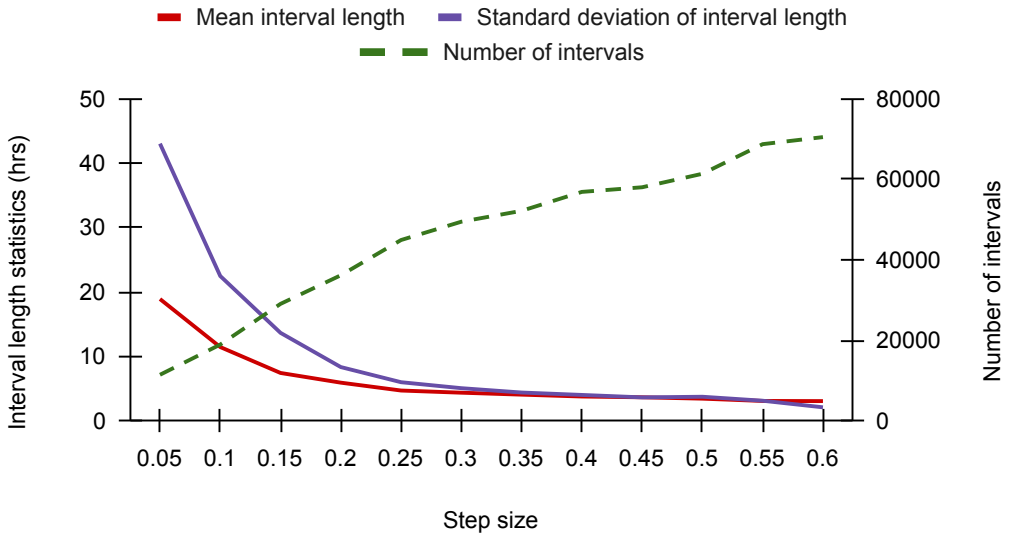
5.1 The Impact of the Variable Capacity on Machine Interval Length

The underlying capacity variation structure determines whether a machine would be alive at a given time step, producing a set of machine intervals. As formally defined in Section 3.3, machine intervals are contiguous blocks of time during which a machine is alive and available for scheduling. In this section, we explore how the underlying variable capacity structure affects the lengths of machine intervals. To build a systematic understanding, we consider the setting of synthetic variable capacity on the dimensions of change frequency and step size. We then extend this study to the subset of observed variable capacity.

We evaluate the impact of variable capacity on machine interval length $w_{j,i}.len$ in Figure 5.1. We observe that as the change frequency and step size increases, the mean interval



(a) Change frequency

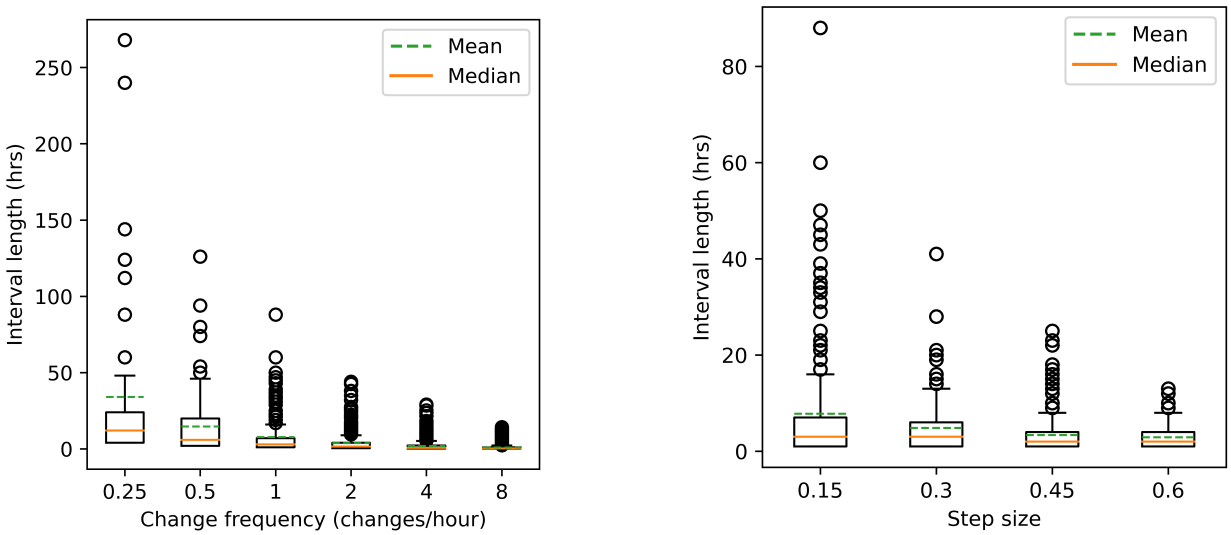


(b) Step size

Figure 5.1: Interval length statistics under increasing change frequency and step size

length decreases. Increasing capacity variation in the underlying system results in generally shorter interval lengths. As a result, the number of intervals in the variable capacity profile increases.

As shown previously in Figure 5.1, the variance of interval lengths decreases with increasing change frequency and step size. We examine the underlying cause of this characteristic in Figure 5.2. As a result of shorter mean interval lengths under increasing change frequency and step size, the range of values that an interval length can take narrows. This is shown by the narrowing spread of the interval length distribution in Figure 5.2. Therefore, as the change frequency and step size increases the uncertainty of an interval length decreases.



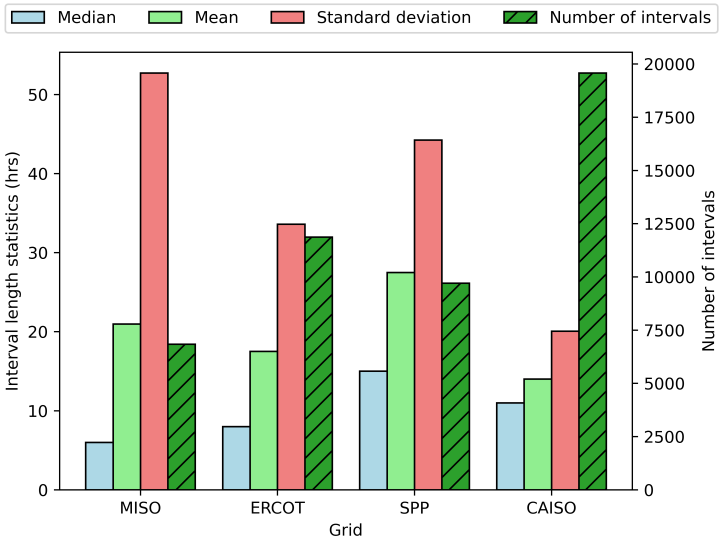
(a) Change frequency

(b) Step size

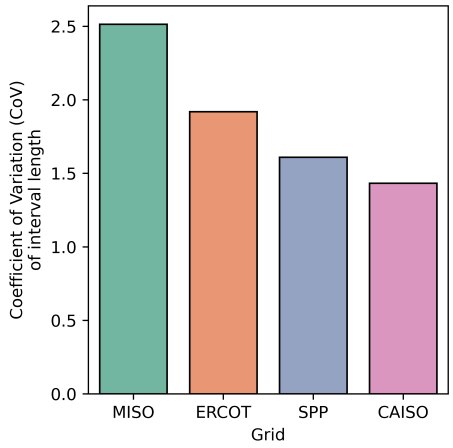
Figure 5.2: Distribution of interval lengths under increasing change frequency and step size

So far, we evaluated the multi-dimensional space of synthetic variable capacity and its impact on machine interval characteristics. Next, we highlight a subset of machine intervals under observed variable capacity. We examine how the properties of power grids influence machine interval characteristics. Our findings below tie with Section 4.2, which introduced observed variable capacity driven by variations in the power grids' carbon intensity.

As previously stated in Sections 2.1.2 and 4.2, we order the power grids, MISO, ERCOT,



(a) Statistics of machine interval length



(b) Coefficient of variation of machine interval length

Figure 5.3: Machine interval lengths under observed variable capacity

SPP and CAISO, with increasing coefficient of variation in hourly average carbon intensity. This results in increasing capacity variation across the four power grids. In Figure 5.3a, we evaluate machine interval lengths across the grids. We observe that machine intervals in CAISO, which has the highest coefficient of variation in hourly average carbon intensity, have the lowest mean interval length and the highest number of intervals. Further, in Figure 5.3b, we observe that the coefficient of variation in machine interval length decreases across the grids. This indicates that in power grids that experience higher variation, machine interval lengths are more consistent. This is in line with our findings in related to synthetic variable capacity above, as lower mean interval length leads to lower spread in the distribution of machine interval lengths.

It is further interesting to note differences in machine interval characteristics across ERCOT and SPP in Figure 5.3. SPP is a wind-dominated power grid, while ERCOT has a mix of both solar and wind generation. Due to the longer periodicity of wind generation, SPP has longer machine interval lengths. This has implications on workload performance as evidenced in Figure 4.10 and discussed in Section 4.2. However, SPP's higher coefficient of

variation in carbon intensity results in higher variance in machine interval lengths.

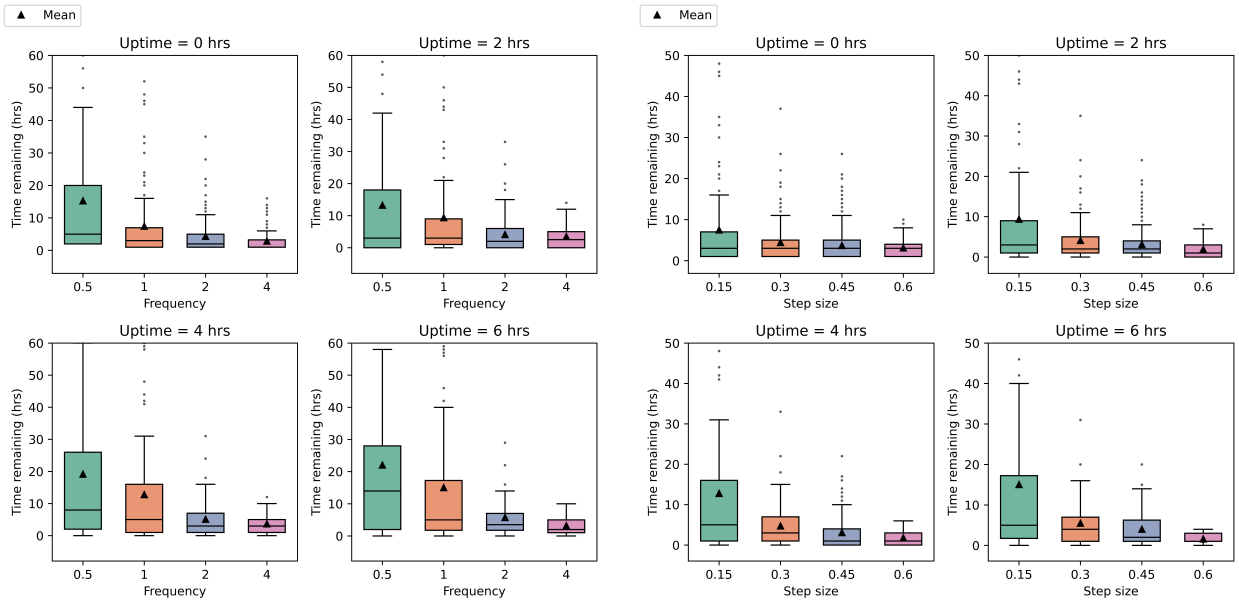
Insights:

- As the change frequency and step size increases, intervals become shorter and the number of intervals increase.
- As the change frequency and step size increases, the variance of interval length decreases, reducing uncertainty in machine interval lengths.
- Grids with higher variation in carbon intensity have shorter machine intervals.
- The periodicity of the underlying generation sources in a power grid influences machine interval characteristics.

5.2 The Impact of Variable Capacity on Time Remaining in an Interval

We formally defined the time remaining in an interval conditioned on uptime as $w_{j,i}.remaining(t)$ in Section 3.3. In this section, we evaluate the impact of the underlying capacity variation structure on the distribution of time remaining in an interval conditioned on uptime.

Figure 5.4 provides evidence that the distribution of time remaining in an interval conditioned on uptime, depends on the variable capacity structure and the uptime. For simplicity, we sampled uptime $w_{j,i}.uptime(t)$ values 0, 2, 4, 6 hours. However, our results hold over the full range of $w_{j,i}.uptime(t)$ values. As the change frequency and step size increases, the mean time remaining in an interval conditioned on uptime decreases. Further, the spread of time remaining values decrease with increasing capacity variation, resulting from shorter mean interval lengths as observed in Figure 5.1. This translates to lower variance in time remaining as capacity variation increases.



(a) Change frequency

(b) Step size

Figure 5.4: Distribution of time remaining in an interval conditioned on uptime, against change frequency and step size

Insights:

- The distribution of the time remaining in an interval conditioned on uptime depends on both uptime and the variable capacity structure.
- As the change frequency and step size increases, the mean time remaining in an interval conditioned on uptime decreases.

5.3 Machine Interval Characteristics for Predicting Interval Time Remaining

In prior sections, we explored a range of machine interval characteristics under variable capacity. In this section, our objective is to highlight machine interval characteristics that could potentially predict interval time remaining and allow better job-to-interval alignment for scheduling under variable capacity. These machine interval characteristics include,

- ***Distribution of the interval length***: The underlying capacity variation structure determines the distribution of interval lengths $w_{j,i}.len$. The distribution of $w_{j,i}.len$ of past intervals under the variable capacity profile could inform the lengths of future intervals for scheduling. Statistics such as the mean, P50, P90 and standard deviation of $w_{j,i}.len$ and its probability distribution could provide an indication of possible interval lengths under the variable capacity profile.

- ***Uptime of an interval***: The uptime of a machine interval $w_{j,i}.uptime(t) = t - w_{j,i}.start$, indicates how far an interval has progressed since its start. This is readily-available information within a cluster. Combined with other information, this could be used assess the current point in progression of an interval, relative to its end.

- ***Time remaining in an interval conditioned on uptime***: $w_{j,i}.remaining(t) = w_{j,i}.len - w_{j,i}.uptime(t)$ represents the time remaining in an interval, conditioned on its uptime. With knowledge of $w_{j,i}.uptime(t)$, we could develop statistical and probabilistic models to better understand the time remaining an interval and thereby, anticipate its end

- ***Time between capacity changes***: Under the variable capacity model, capacity changes occur every $\frac{1}{freq}$ time unit. Therefore, at every $\frac{1}{freq}$ time point, there is some probability that an interval could end. From the scheduler's point of view, being aware of when the next capacity change happens within the variable capacity system would increase the probability of being able to avoid an interval end before a job completes.

- ***Machine index***: Under the variable capacity model, machine indices are ordered based on their risk of termination. Differences in the level of risk experienced by the machines could be exploited by a scheduler to better align higher-value jobs to more stable machine intervals.

In Chapter 6, we propose and evaluate a class of scheduling heuristics that exploit above listed machine interval characteristics to better align jobs to machine intervals.

5.4 Summary

The underlying capacity variation structure determines the machine interval characteristics in a variable capacity profile. We observed that as the change frequency and step size increases, interval lengths becomes shorter. We studied the subset of machine intervals under observed variable capacity. Next, we observed that the distribution of time remaining in an interval conditioned on uptime, depends both on the interval's uptime and the underlying capacity variation structure. Finally, we highlighted machine interval characteristics that could potentially be exploited by a scheduler to predict interval time remaining and allow better job-to-interval alignment.

CHAPTER 6

INTERVAL-AWARE SCHEDULING FOR AVOIDING JOB TERMINATIONS

The results in Chapter 4 show that interval ends under variable capacity lead to job terminations, resulting in performance degradation. Chapter 5 highlights machine interval characteristics that could potentially predict interval time remaining. In this chapter, we propose and evaluate a class of scheduling heuristics that exploit these machine interval characteristics to better align jobs to intervals, while balancing the trade-off between other performance metrics. Section 6.1 introduces four scheduling heuristics that exploit various machine interval characteristics to reduce specific job termination populations and later combines these heuristics to develop the Interval-Aware Scheduler (IAS) that aims to lower job terminations across the entire workload. In Section 6.2, we analyze how IAS manages the trade-off between goodput and terminations. Section 6.3 evaluate IAS's performance across a range of real and synthetic cloud workloads.

6.1 Exploiting Machine Interval Characteristics for Scheduling

Building on the machine interval characteristics identified in Section 5.3 and the scheduling framework in Section 3.3, this section develops and evaluates four scheduling heuristics that exploit varying levels of machine interval information to predict interval time remaining and improve job-to-interval alignment, with the goal of reducing job terminations. Section 6.1.1 evaluates these scheduling heuristics individually, on their effectiveness in reducing terminations in the targeted job population and impact on other performance metrics. A more effective scheduler should combine multiple heuristics to handle the terminations across the entire workload. In Section 6.1.2, we evaluate the effects of combining heuristics and develop the Interval-Aware Scheduler (IAS) that effectively combines these heuristics, while

balancing the trade-off among performance metrics.

6.1.1 Heuristics exploiting various Machine Interval Characteristics

In this section we develop scheduling heuristics that exploit varying levels of machine interval information to avoid terminations. We consider the following scheduling heuristics, summarized in Table 6.1.

- ***Prioritize Big Jobs*** ($H1$): Schedule as many big jobs as possible on the fraction of resources that do not experience variable capacity (stable resources). This heuristic aims to reduce terminations of big jobs (longer duration / high resource consumption).

  The threshold for determining which jobs to schedule on stable resources could be determined based on various criteria, such as a user-defined job duration, a workload-based job duration percentile, or fitting jobs with high resource consumption to stable resources. In Algorithm 1, we consider prioritizing big jobs based on the workload's resource consumption, in a variable capacity platform under a LIFO machine termination policy.

- ***Capacity Change-aligned*** ($H2$): This heuristic delays scheduling jobs until after points in time that capacity changes occur. Consequently, avoiding jobs terminations resulting from interval ends by waiting until the next capacity change.

  Applying this heuristic indiscriminately across the entire workload would be sub-optimal, as it leads to high scheduling delays. Therefore, we focus on 'small' jobs, i.e., smaller job duration than the time between capacity changes, as they are more sensitive to capacity changes between consecutive time units. Algorithm 2 outlines this heuristic with additional consideration of the time elapsed since the last capacity change.

- ***Mean for Interval Time Remaining*** ($H3$): As the underlying capacity variation structure determines the machine interval characteristics, the statistical properties of time remaining conditioned on uptime (e.g., mean, median, standard deviation) could predict an interval's potential end. Under this heuristic, we schedule jobs on an interval only if its statistical properties indicate that the interval would last long enough to complete the job. As indicated by Algorithm 3, this heuristic considers an interval's uptime and the mean time remaining derived from the variable capacity platform's interval distribution. We constrain this heuristic to longer duration jobs as they would have a higher impact on goodput from termination.

- ***Conditional Probability for Interval Time Remaining*** ($H4$): This heuristic refines on the information exploited by $H3$, and considers the conditional probability distribution of time remaining (Probability Density Function (PDF) of time remaining conditioned on uptime) when determining whether an interval would last long enough to complete a job. This heuristic depends on a tunable aggressiveness measure to determine whether an interval would last longer than a job (see Algorithm 4). Higher aggressiveness indicates greater optimism, while a lower aggressiveness reflects a more cautious approach.

Heuristic	**Machine interval characteristic**	**Target job population**
Prioritize Big Jobs ($H1$)	Machine indices of stable resources	Big jobs (longer duration / high resource consumption)
Capacity Change-aligned ($H2$)	Time between capacity changes	'Small' jobs (smaller job duration than the time between capacity changes)
Mean for Interval Time Remaining $H3$)	Interval uptime + mean of time remaining conditioned on uptime	Long jobs ($>$ P90 job duration in the workload)
Conditional Probability for Interval Time Remaining ($H4$)	Interval uptime + PDF of time remaining conditioned on uptime	All jobs

Table 6.1: Summarization of scheduling heuristics

Algorithm 1: Prioritize Big Jobs ($H1$)

Data: job j_k, machine m_i,
resource consumption threshold for long jobs $longJobThresh$
Result: Schedule job j_k on machine m_i
$stableMachineIndex \leftarrow |M| * lb - 1$;
if $j_k.exec * j_k.res \geq longJobThresh$ **then**
  if $i \leq stableMachineIndex$ **then**
    Schedule j_k on m_i;
  end
end
else if $j_k.exec * j_k.res < longJobThresh$ **then**
  if $i > stableMachineIndex$ **then**
    Schedule j_k on m_i;
  end
end

Algorithm 2: Capacity change-aligned ($H2$)

Data: job j_k, machine m_i
Result: Schedule job j_k on machine m_i
$timeUntilNextChange \leftarrow (1 - (t \bmod \frac{1}{freq})) * \frac{1}{freq}$;
if $j_k.exec \leq \frac{1}{freq}$ **then**
  if $timeUntilNextChange > j_k.exec$ **then**
    Schedule j_k on m_i
  end
  else if $t \bmod \frac{1}{freq} == 0$ **then**
    Schedule j_k on m_i
  end
end
else if $j_k.exec > \frac{1}{freq}$ **then**
  Schedule j_k on m_i
end

Algorithm 3: Mean for interval time remaining ($H3$)

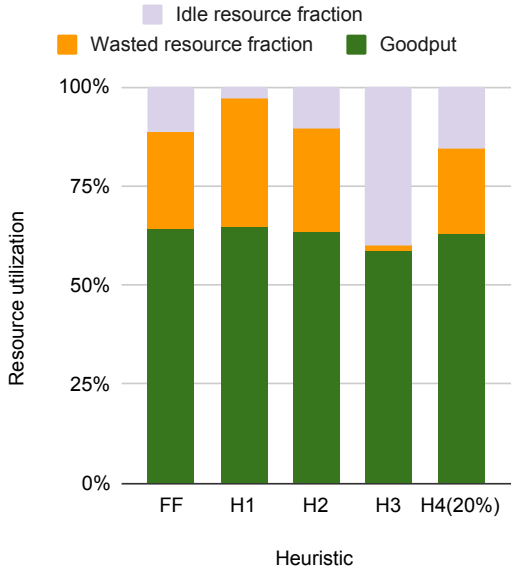
Data: job j_k, machine m_i, $P90$ job duration
  $w_{j,i}$, mean time remaining $\overline{(w_{j,i}.remaining(t) \mid w_{j,i}.uptime(t))}$
Result: Schedule job j_k on machine m_i
if $j_k.exec > P90$ **then**
  if $\overline{(w_{j,i}.remaining(t) \mid w_{j,i}.uptime(t))} \geq j_k.exec$ **then**
    Schedule j_k on m_i
  end
end
else
  Schedule j_k on m_i
end

Algorithm 4: Conditional probability of interval time remaining ($H4$)

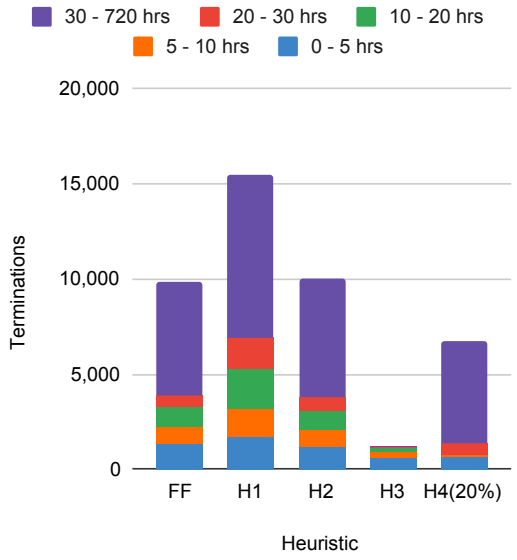
Data: job j_k, machine m_i, aggressiveness measure A,
  $w_{j,i}$, probability distribution $prob(w_{j,i}.remaining(t) \mid w_{j,i}.uptime(t))$
Result: Schedule job j_k on machine m_i
if $prob(w_{j,i}.remaining(t) >= j_k.exec | w_{j,i}.uptime(t)) < A$ **then**
  Schedule j_k on m_i
end

We evaluate the heuristics in Figure 6.1, on a the synthetic heavy-tailed workload with a Zipf distribution skew of 1.5 and mean job duration of 11.73 hrs (see Figure 6.3 for the jobs duration distribution of the workload). The underlying variable capacity platform has a 60% dynamic range, change frequency of 1 change/hour and step size of 0.15. We evaluate heuristics against the baseline cloud scheduling algorithm First-Fit (FF).

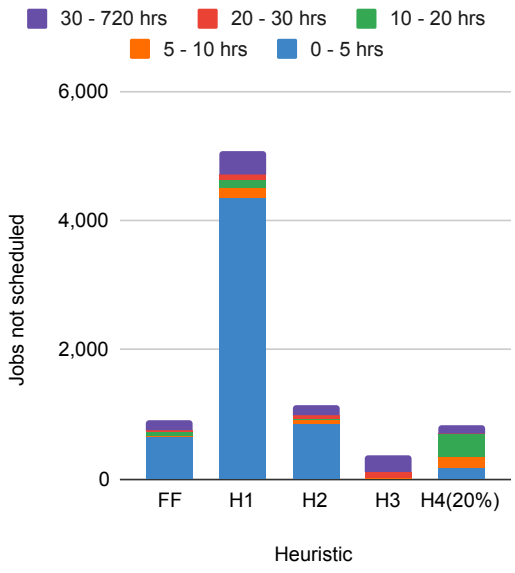
Prioritize big jobs ($H1$) increases the number of terminations and the number of jobs not scheduled. However, we see a 0.5% increase in goodput over FF, resulting from big jobs being safely scheduled on stable resources. The scheduling latency under this heuristic increases, as jobs await suitable stable/variable resources to become available for scheduling. By aligning capacity changes for small jobs, Yj***capacity change-aligned*** ($H2$) reduces the number of small job terminations (0-5 hrs job category) by 8.17% over FF. However, long job terminations increase, resulting in a 1.4% increase in the wasted resource fraction over FF. Further, $H2$ marginally increases scheduling latency, by delaying the scheduling of jobs to capacity changes. ***Mean for interval time remaining*** ($H3$) reduces terminations in the targeted long job population, resulting in 23.36% lower wasted resources than FF. However, a large fraction of long jobs (>20hrs job duration categories) are unable to find suitable resources, as the set of suitable intervals become narrower for long jobs, leading to 5.35% lower goodput. ***Conditional probability for interval time remaining*** ($H4$) targets an overall better alignment of jobs to intervals. This is evident in its lower number of job terminations and overall distribution, when compared with FF. However, a significant number of shorter jobs (<10 hrs) are unable to find suitable resources, as reflected in the distribution of jobs not scheduled. Therefore, a more optimal version of $H4$ would target longer duration jobs. $H3$ and $H4$ achieves lower P50, P90, P95 scheduling latency, as they achieve a better matching of jobs to intervals across jobs scheduled. Overall, $H4$ is the best performing heuristic across both goodput and terminations, achieving 32.13% less termination with a 0.9% goodput loss.



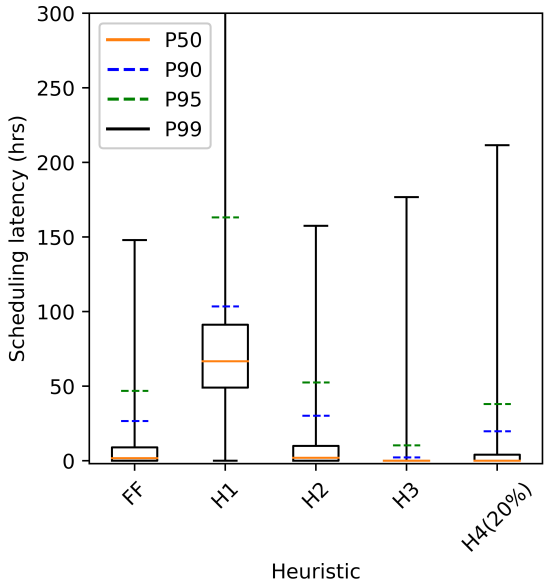
(a) Resource utilization



(b) Job terminations



(c) Jobs not scheduled



(d) Scheduling latency

Figure 6.1: Evaluating the heuristics individually, under variable capacity

We conclude that the scheduling heuristics proposed in this section are individually successful in reducing their respective targeted job populations, but have a negative impact on other job populations leading to suboptimal scheduling performance.

6.1.2 Improving Performance by Combining Heuristics

Based on our findings in Section 6.1.1, we deduce that a more efficient scheduler would consider a combination of the proposed heuristics, to reduce terminations across all job populations in the workload. Combining heuristics would not always result in optimal behavior, owing to potential conflicts and increasing complexity in the interactions among individual heuristics. However, we could improve performance by combining heuristics designed to target distinct job populations.

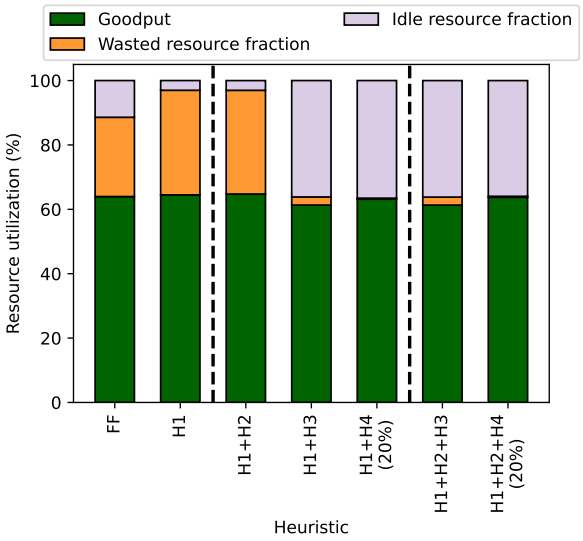
In this section, we explore various heuristic combinations, with the objective of developing a scheduler that effectively combines the proposed heuristics, while balancing the trade-off among performance metrics. While numerous combinations of heuristics is possible, we limit our evaluation in Figure 6.2 to five, in order to systematically understand their effects on scheduler performance. Starting from the FF baseline and prioritizing big jobs ($H1$), we successively examine the impact of combining individual heuristics to the scheduler, grouping our analysis based on combinations of two and then three heuristics.

First, we evaluate combinations of two heuristics. Combining the properties of capacity change-aligned ($H2$) to the scheduler lowers the number of small job terminations (0-5 hrs job category), which is the heuristic's target job population. This is observed in the combination $H1 + H2$ compared to $H1$ individually, within its distributions of job terminations and jobs not scheduled. As mean for interval time remaining ($H3$) targets long jobs, $H1 + H3$ offers incremental benefits in long job terminations over $H1$, by allowing better job-to-interval alignment in the variable resource pool. Therefore, $H1 + H3$ significantly lowers the number of long job terminations and the wasted resource fraction compared to $H1$. However, the

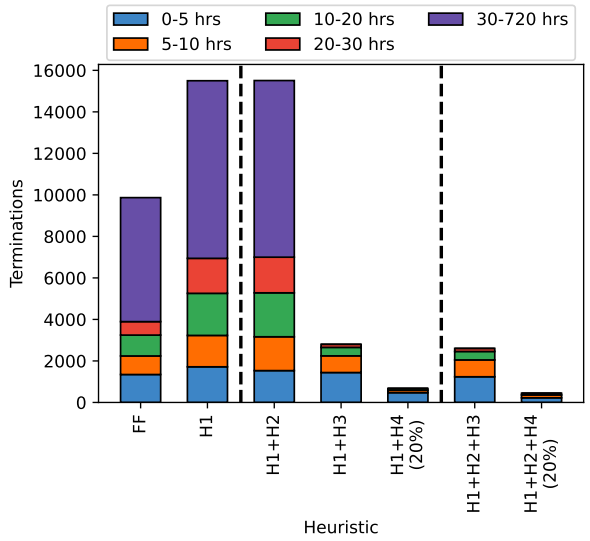
number of jobs not scheduled increase under $H3$, owing to the narrow set of suitable interval for long jobs allowed based on the heuristic's criteria. In $H1 + H4$, conditional probability for interval time remaining ($H4$) targets all jobs scheduled on the variable capacity pool and drastically lowers the number of terminations and jobs not scheduled, compared to the baseline.

Evaluating combinations of three heuristics, we observe that the combination $H1 + H2 + H4$ provides the best performance, in terms of both job terminations and goodput. This combination reduces terminations across all job populations. For the evaluated workload, we can achieve a 95% reduction in terminations over FF, with 0.2% reduction in goodput. The minimal goodput loss is a result of the scheduler being unable to effectively utilize the idle resource fraction as it awaits more suitable resources for jobs. Moreover, the scheduler could inform jobs that they were not scheduled rather than asynchronously terminating them later. As shown in Figure 6.2d, the P50 and P90 scheduling latency of the $H1 + H2 + H4$ combination is lower than FF.

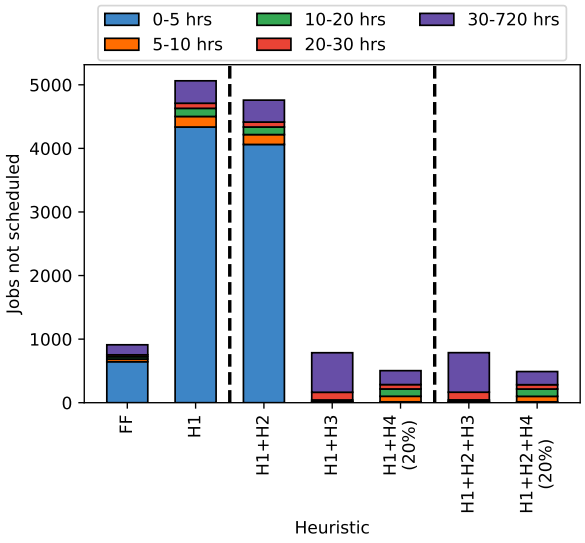
We build the Interval-Aware Scheduler (IAS) combining the heuristics prioritize big jobs ($H1$), capacity change-aligned ($H2$) and conditional probability for interval time remaining ($H4$) (see Algorithm 5). Going forward, our evaluation is based on the Interval-Aware Scheduler.



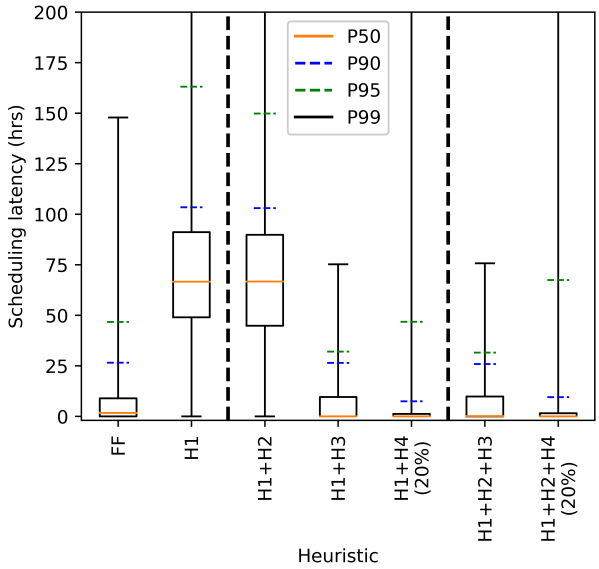
(a) Resource utilization



(b) Job terminations



(c) Jobs not scheduled



(d) Scheduling latency

Figure 6.2: Evaluating combinations of heuristics, under variable capacity

Algorithm 5: Interval-Aware Scheduler / IAS ($H1 + H2 + H4$)

Data: job j_k, machine m_i, aggressiveness measure A,
resource consumption threshold for long jobs $longJobThresh$,
$w_{j,i}$, probability distribution $prob(w_{j,i}.remaining(t) \mid w_{j,i}.uptime(t))$
Result: Schedule job j_k on machine m_i
$stableMachineIndex \leftarrow |M| * lb - 1$;
$timeUntilNextChange \leftarrow (1 - (t \bmod \frac{1}{freq})) * \frac{1}{freq}$;
if $j_k.exec * j_k.res \geq longJobThresh$ **then**
  if $i \leq stableMachineIndex$ **then**
    Schedule j_k on m_i;
  end
end
else if $j_k.exec * j_k.res < longJobThresh$ **then**
  if $i > stableMachineIndex$ **then**
    if $j_k.exec \leq \frac{1}{freq}$ **then**
      if $timeUntilNextChange > j_k.exec$ **then**
        Schedule j_k on m_i
      end
      else if $t \bmod \frac{1}{freq} == 0$ **then**
        Schedule j_k on m_i
      end
    end
    else if $j_k.exec > \frac{1}{freq}$ **then**
      if $prob(w_{j,i}.remaining(t) >= j_k.exec | w_{j,i}.uptime(t)) < A$ **then**
        Schedule j_k on m_i
      end
    end
  end
end

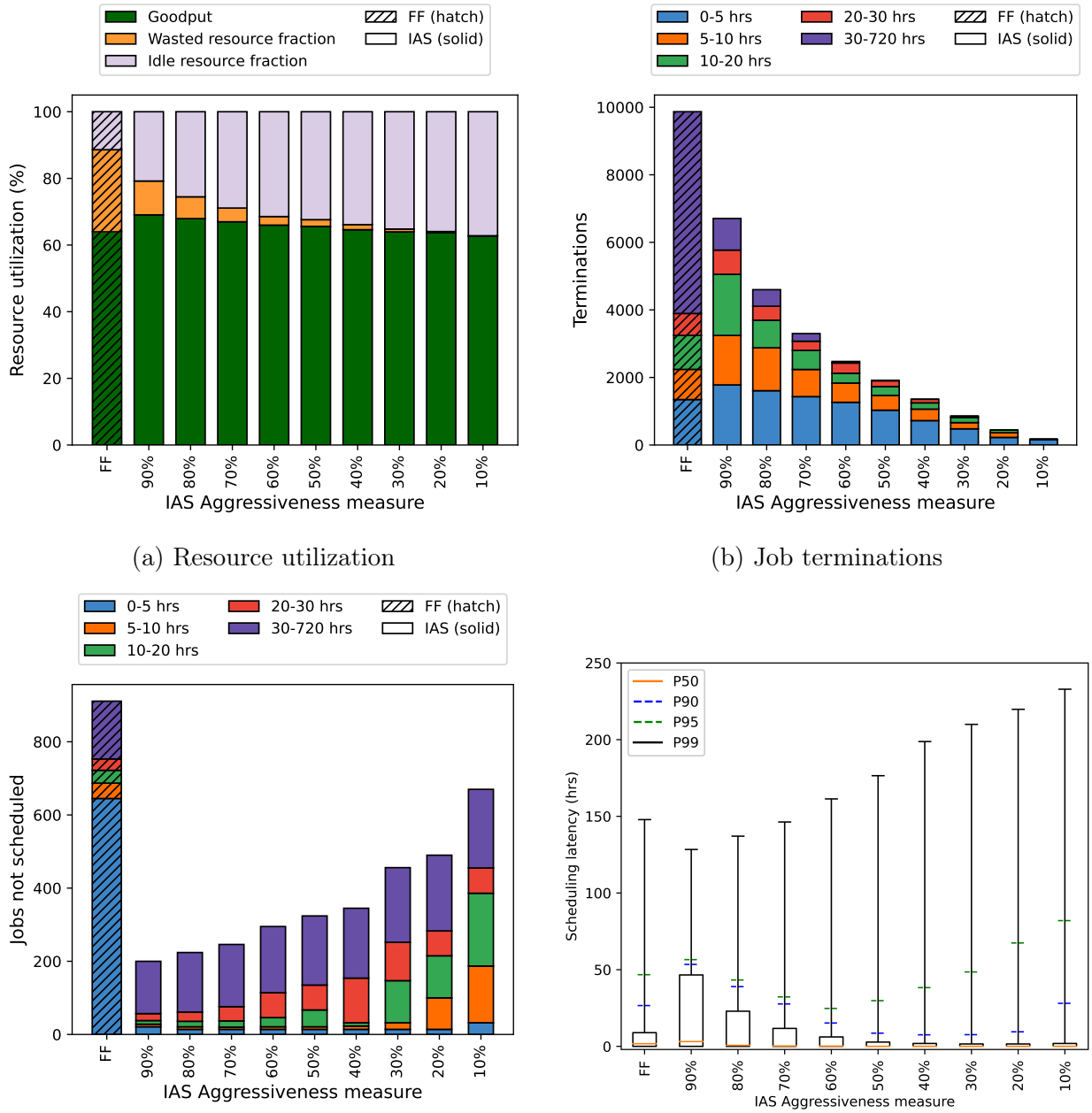
Insights:

- The proposed scheduling heuristics are individually successful in reducing their respective targeted job populations, but have a negative impact on other job populations.
- A more efficient interval-aligned scheduler would explore a combination of heuristics for reducing terminations across the entire workload.
- The Interval-Aware Scheduler (IAS) drastically reduces job terminations by 95% over First-Fit, with a 0.2% reduction in goodput and lower scheduling latency (P50, P90) than First-Fit.

6.2 Balancing Terminations and Goodput in the Interval-Aware Scheduler

In the prior section, we built the IAS (Algorithm 5) that achieves the best performance, in terms of both terminations and goodput, on cloud workloads under variable capacity. In this section, we evaluate how the selection of this aggressiveness measure A affects IAS's trade-off between goodput and terminations.

IAS assesses intervals based on their likelihood of lasting long enough to complete a job, avoiding its termination. Thus, IAS depends on a tunable aggressiveness measure that determines whether an interval would last longer than a particular job's duration. Higher aggressiveness indicates that the scheduler is more optimistic about intervals lasting longer, consequently making riskier scheduling decisions. Lower aggressiveness of the scheduler reflects a more cautious approach, delaying the scheduling of jobs until a better suited interval becomes available.



(a) Resource utilization

(b) Job terminations

(c) Jobs not scheduled

(d) Scheduling latency

Figure 6.3: IAS's performance across the range of aggressiveness measures, under variable capacity

As the scheduler becomes less aggressive (more pessimistic about time remaining), it assumes a lower risk of the interval ending sooner than the job duration. Consequently, the number of job terminations lower as the scheduler's aggressiveness decreases (Figure 6.3b). Additionally, a high fraction of jobs await more suitable resources to become available and might not get scheduled (Figure 6.3c). If the platform is willing to accept higher terminations (aggressiveness measure $> 30\%$), IAS could achieve greater goodput than FF, by trading-off job terminations for goodput.

IAS consistently outperforms the baseline in reducing job terminations, regardless of the scheduler's aggressiveness. Under 90% aggressiveness, IAS achieves 5% greater goodput, with a 32% decrease in the number of terminations and 14.5% lower wasted resources over FF. The number of terminations can be further reduced by up to 98% under lower aggressiveness, if a 1.25% reduction in goodput over FF is tolerable.

The appropriate tuning of the aggressiveness measure depends on the workload's requirements and cost of performance loss. For a cloud workload, where stringent SLOs must be met, lower aggressiveness is preferred as it yields the lowest terminations. With 10% aggressiveness, IAS achieves a failure rate of 0.6%, which translates to 99.4% availability. This is comparable to typical cloud VM availability SLOs of 99.5% [1] and 99.9% [33], and not much larger than the expected unavailability of cloud VMs.

Insights:

- The Interval-Aware Scheduler consistently outperforms First-Fit with fewer terminations, regardless of the scheduler's aggressiveness.
- IAS achieves up to 5% greater goodput than First-Fit, with a 32% reduction in job terminations, by accepting higher terminations.
- IAS could further reduce the number of terminations by up to 98% over First-Fit, if a tolerate a 1.25% goodput loss is tolerable.
- IAS can achieve 99.4% availability under variable capacity, which is comparable to typical cloud VM SLOs and not much larger than expected unavailability.

6.3 The Impact of Workload Job Duration Distribution on IAS Performance

Effective scheduling under variable capacity involves accurately aligning job with intervals. Therefore, the workload's distribution of job duration relative to the variable capacity profile's interval length distribution ($w_{j,i}.len$) becomes an important consideration. We evaluate IAS with the Azure workload (detailed in Section 2.2.2), which is extremely heavy-tailed. In order to characterize the impact of the workload's job duration distribution on the IAS's performance, we further examine a range of synthetic heavy-tailed workloads with varying Zipf skew parameters (previously discussed in Section 4.3), ordered from more to less heavy-tailed. The resulting job duration distributions are as shown in Figure 6.4.

Figure 6.5 shows the resource utilization under IAS across the set of workloads. In the extremely heavy-tailed Azure workload, IAS decreases the wasted resource fraction by 22.63%, while maintaining similar goodput to FF (-0.05%). Across all synthetic workloads, IAS achieves 0.25–3.81% greater goodput and lower wasted resources, compared to FF. IAS

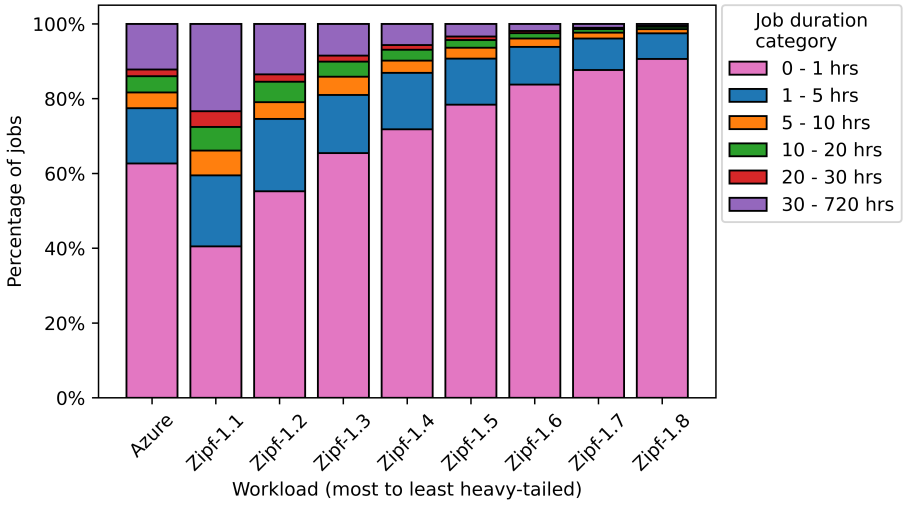


Figure 6.4: Distribution of job durations across evaluated workloads

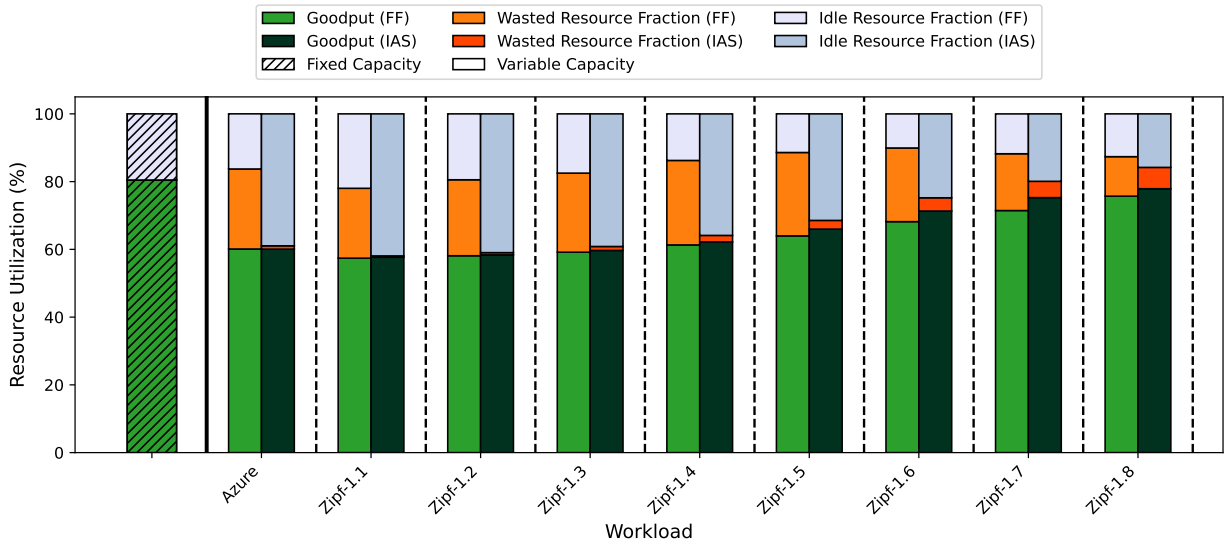


Figure 6.5: Resource utilization under IAS(A=60%) across workloads, compared to First-Fit scheduling

achieves greater goodput in less heavy-tailed workloads. As the workload becomes heavier, there is less opportunity for the scheduler to increase goodput, owing to a narrowing set of suitable intervals for the growing fraction of long jobs.

It is worth noting that while it is possible to achieve greater goodput in less heavy tailed workloads, the wasted resource fraction increases. From a goodput maximization point of view, IAS significantly outperforms FF. However, from the terminations perspective, we might have to compromise goodput in order to avoid service disruptions. This creates a non-trivial trade-off, between the likelihood of a job being scheduled and the likelihood of termination.

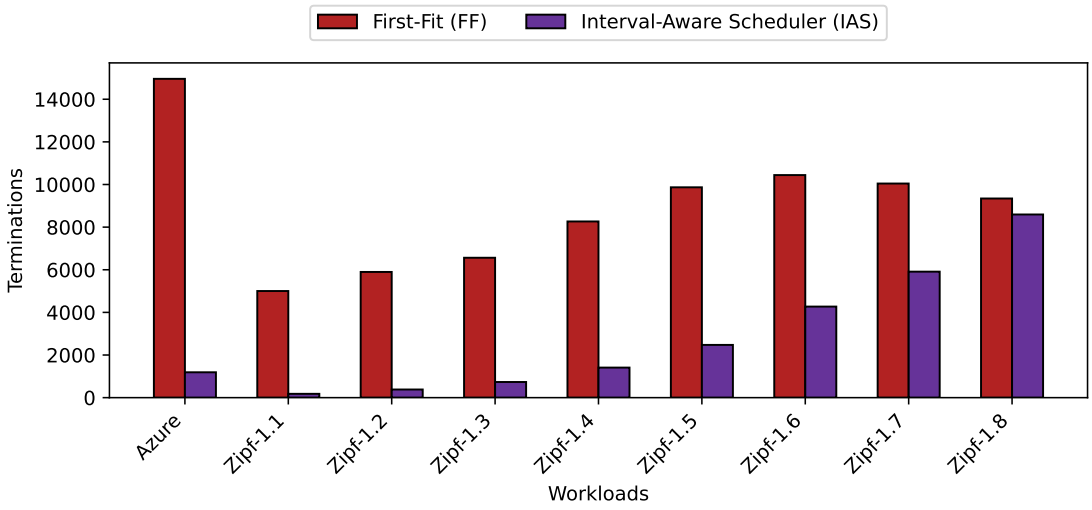


Figure 6.6: Job terminations under IAS(A=60%) across workloads, compared to traditional First-Fit scheduling

IAS is successful in drastically reducing terminations over FF across a broad range of heavy-tailed workloads. In Figure 6.6, we evaluate the number of job terminations under IAS, against the baseline FF. In the Azure workload, we can achieve 92% lower terminations under variable capacity. Figure 6.7 and Figure 6.8 drills down on the number of jobs terminations and jobs not scheduled under IAS, across the spectrum of synthetic workloads. The number of job terminations and jobs not scheduled increase as the workload becomes less heavy-

tailed, as a result of an increasing number of jobs in the workload. However, both the number of terminations and number of jobs not scheduled relative to the number of jobs in the workload decrease as the workload becomes less heavy-tailed. We observe a higher number of long jobs not scheduled in heavier-tailed workloads, which contributes to the difficulty in achieving greater goodput than FF in heavier-tailed workloads.

The relative reduction in terminations compared to the baseline increases as the workload becomes more heavy-tailed. When drilling down on the distribution of job duration in terminations (Figure 6.7), we observe that the capacity change-aligned heuristic within IAS is successful in reducing all terminations within the targeted small job duration population (0-1 hrs). The prioritize big jobs and conditional probability for time remaining heuristics are collectively successful in reducing long job terminations (30-720 hrs). Table 6.2 summarized the performance benefits of IAS over FF, across the set of workloads.

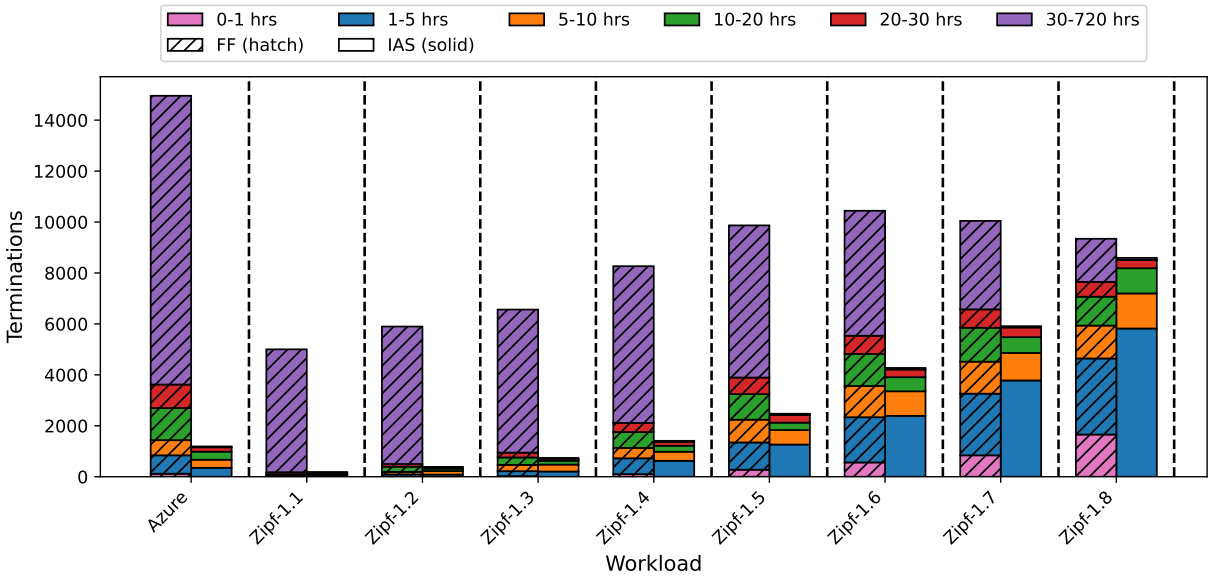


Figure 6.7: Drilldown of terminations under IAS(A=60%), across synthetic workloads

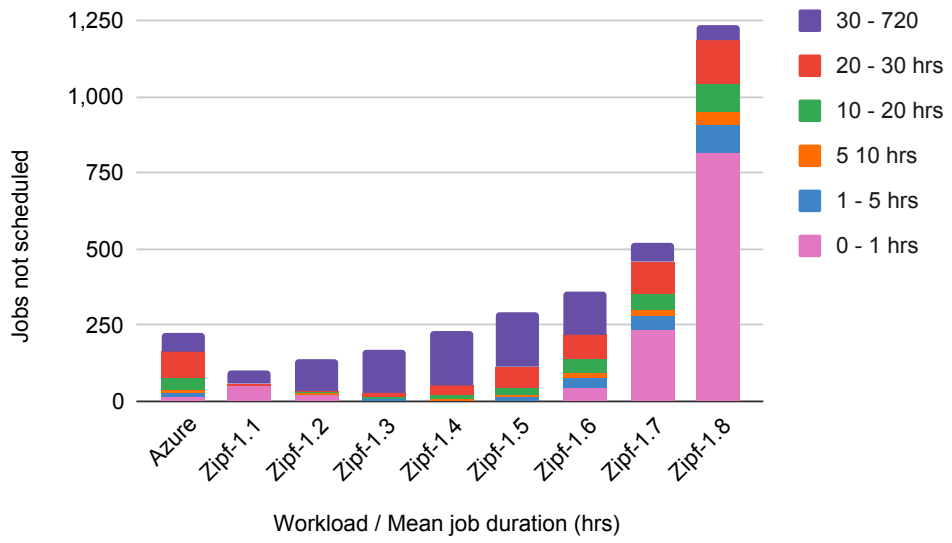


Figure 6.8: Drilldown of jobs not scheduled under IAS(A=60%), across synthetic workloads

Workload	**Mean job duration (hrs)**	**Reduction in terminations over FF**	**Gain in goodput over FF**
Azure	60.12	92.04%	-0.05%
Zipf-1.1	288.09	96.42%	0.27%
Zipf-1.2	123.24	93.52%	0.25%
Zipf-1.3	58.86	88.82%	0.52%
Zipf-1.4	26.07	82.94%	0.89%
Zipf-1.5	11.73	74.93%	2.02%
Zipf-1.6	5.43	59.09%	3.13%
Zipf-1.7	2.90	41.15%	3.81%
Zipf-1.8	1.51	8.03%	2.21%

Table 6.2: Performance benefits of IAS(A=60%), over the baseline First-Fit

Insights:

- IAS drastically reduces job terminations across a broad range of heavy-tailed job duration distributions.
- For the extreme heavy-tailed Azure workload, IAS achieves 92% lower terminations under variable capacity, while maintaining similar goodput to First-Fit.
- IAS achieves greater goodput than First-Fit in less heavy-tailed workloads.
- As the workload becomes heavier, there is less opportunity for IAS to increase goodput, but the relative reduction in terminations over First-Fit increases.

6.4 Summary

We propose and evaluate a class of scheduling heuristics that exploit machine interval characteristics to better align jobs to machine intervals. These heuristics are individually successful in reducing terminations in their respective target job population, but have a negative impact on other job populations. By combining them, we construct the Interval-Aware Scheduler (IAS) that drastically reduces terminations across a broad range of heavy-tailed distributions. IAS outperforms FF in reducing terminations, regardless of the scheduler's aggressiveness. IAS achieves 5% greater goodput than FF, with 32% lower terminations. IAS can further reduce terminations by up to 98% over FF, if the system can tolerate 1.25% goodput loss. Balancing the trade-off between goodput and terminations depends on the workload's requirements. In the extremely heavy-tailed Azure workload, IAS achieves 92% lower terminations under variable capacity, with 0.05% goodput loss. IAS can achieve greater goodput in less heavy-tailed workloads. As the workload becomes heavier-tailed, there is less opportunity for IAS to increase goodput, but the relative reduction in terminations over FF increases.

CHAPTER 7

RELATED WORK

We are motivated by prior work in dynamic data centers whose capacity is driven by external constraints, workload scheduling for reducing carbon emissions, and scheduling on variable capacity resources. In this section, we summarize key findings and techniques from these domains. However, no prior work has focused on exploiting machine interval characteristics to inform scheduling under variable capacity.

7.1 Dynamic Data Centers

The concept of data center flexibility driven by external constraints has been explored in various contexts in the literature. Renewable generation, demand response, electricity prices and carbon intensity signals are several noteworthy driving forces for variable capacity data centers.

Several studies have investigated the potential for data centers to modulate their resource capacity based on renewable energy availability. Goiri et al. proposed strategies to manage data centers powered by renewable energy. Parasol and GreenSwitch [30] evaluates the possibility of adapting the workload to match the renewable energy supply. Li et al. investigates the opportunistic use of batteries to store renewable energy surplus and discharge when renewable energy is not available [43]. Most work in this domain, including Greenpar [37], GreenSlot [31], and GreenHadoop [29], relies on supplemental power–such as electricity grid connections, backup generators, or batteries–to manage the integration of renewables to data center power supply, whereas our study does not assume external sources to maintain a stable capacity.

A body of research in data center demand response explores the coupling resource management with the power grid. Demand response programs are essential for maintaining grid

reliability. In [11], Chen et al. studies the greening of data center demand response that typically relies on diesel backup generators and workload shedding. Further research [44] investigates combining workload scheduling and local power generation to avoid the coincident peak and reduce the energy expenditure.

The Zero Carbon Cloud project has extensively explored harnessing stranded renewable generation for high-performance computing. This considers the case of volatile computing resources that are turned fully on or off to best utilize intermittently available, stranded renewable generation [13, 14]. Variable capacity driven by electricity prices could improve operational cost efficiency in data centers [15], and past literature has examined the feasibility of data center dynamicity based on electricity price signals [40].

Data center load adaptation based on the carbon intensity of power is assessed in Carbon Responder [69], which further investigates efficiently and fairly allocating power curtailment across different workloads. Radovanović et al. [56] analyzes risk-aware optimization techniques for generating carbon-aware Virtual Capacity Curves (VCCs) for Google's data center clusters. These VCCs impose hourly resource limits for workloads with temporal flexibility while preserving total daily capacity and allowing delayed workloads to complete within 24 hours. Extensions to this work in [35] include leveraging both temporal and spatial flexibility of compute jobs in constructing VCCs.

The flexibility of data center components is necessary for variable capacity and have been examined in prior studies. Flexing data center compute capacity has been proposed through turbo mode [15] and overclocking [62]. FlexCoolDC [28] evaluates expanding the data center cooling equipment's operational range and actively exploiting the cooling design redundancy. Flex [74] considers oversubscribing to the data center power infrastructure in order to expand available power for compute.

To summarize, some prior work in the domain of data center flexibility driven by external phenomena rely on electricity grid connections, backup generators, or batteries to manage

capacity variations. Other work considers load adaptation, including shifting load temporally and spatially. In our work, we rely on a purely variable capacity platform, with no assumption of external generation sources to stabilize capacity. Furthermore, our scheduling approach does not rely on load adaptations but instead focuses on scheduling the workload online using resources available under externally imposed variable capacity.

7.2 Scheduling Workloads for Sustainability

An extensive body of research investigates workload scheduling for reducing carbon emissions and maximizing green use. In this section, we delve into various techniques employed by previous literature.

The use of temporal and spatial workload shifting for data center sustainability could provide significant carbon benefits, owing to fluctuations in carbon intensity over time and space. Wiesner et al. [67] identifies characteristics of delay-tolerant workloads and analyzes the potential for temporal workload shifting. Google's studies on VCCs [56] also rely on temporal workload shifting with a maximum delay of 24 hours, as a load adaptation mechanism. [75] discusses the potential of workload migration between data centers in reducing curtailment and carbon emissions. Shifting compute workloads to geographic regions with lower carbon intensity, based on time zone differences and regional fuel mix have been studied in [70]. Studies of spatial workload shifting has been extended to other data center workloads, considering AI inference workloads in [12] and Content Delivery Networks in CDN-Shifter [50]. A combination of both temporal and spatial shifting techniques for maximizing green use is studied in CarbonClipper [42] and [35].

Resource scaling and job speedup are common techniques designed to consume greater energy when abundant renewable generation is available. Greenpar [37] increases the resource allocations of jobs and uses job speedup profiles to reduce runtimes, allowing them to maximize green energy consumption while meeting SLAs. CarbonScaler [36] exploits the

application's ability to change compute demand by scaling the workload based on carbon intensity signals.

Prediction of renewable availability could inform scheduling on renewable-powered resources. GreenSlot [31], GreenHadoop [29] and [37] predicts the amount of solar energy that will be available in the near future in order to maximize the green energy consumption while meeting job deadlines.

The literature employs temporal workload shifting, which is suitable for the subset of delay-tolerant cloud workloads and HPC workload. In contrast, our work primarily focuses on the cloud service model, with stringent latency requirements. Prior work in predicting renewable availability mainly rely grid/source-level predictions, whereas our work considers future availability at individual machine-level, capturing the effects of the platform's machine termination policy.

7.3 Scheduling on Variable Capacity Resources

The broad literature on workload schedulers deal with fixed capacity platforms [66, 39, 8]. However, a body of research that deals with the addition and removal of resources with time has continued to grow.

The most established studies focus on improving energy efficiency by flexing resource capacity through scheduling. Shut-down models, where a system is put into a sleep state when idle, are shown to reduce energy consumption in [3]. Switching off idle physical compute nodes has been further evaluated in [5], where the scheduler would continuously consolidate VMs and migrate VMs between hosts, rendering a fraction of nodes idle. Speed scaling mechanisms, such as Dynamic Voltage and Frequency Scaling (DVFS), allow processors and servers to run at lower speed at the cost of increased execution times. Speed scaling mechanisms allow processors and servers to run at lower speed at the cost of increased execution times. [68] proposes a scheduler that uses DVFS to allocate jobs in cloud data

centers, reducing energy consumption.

Past studies have explored scheduling policies that assure secure job execution in the presence of unpredictable resource failures, which is an extended form of variable capacity scheduling. [59] extends existing scheduling heuristics, preemptive, replication and delay-tolerant, to provide security assurances. [60] constructs statistical models to assess the reliability of resources based on prior performance and behavior, considering this reputation-based reliability rating in the job allocation algorithm. Reputation-based scheduling on unreliable resources is further analyzed in [2].

Opportunities to mitigate performance degradation from variable capacity through intelligent machine selection for termination is explored in [73]. Two policies that terminate machines such that wasted work is minimized or terminated jobs with the least fraction completed are evaluated. [54] proposes an online scheduler that determines the fraction of resources that can be safely used, relying on estimations of the associated risk of machine under variable capacity.

In summary, prior work in scheduling on variable capacity resources rely on VM migration, speed scaling, reputation-based scheduling and intelligent machine termination. In contrast to existing methods, our approach exploits the characteristics of the underlying platform to better align jobs to machine intervals.

7.4 Summary

In prior chapters, we demonstrated the possibility of exploiting machine interval characteristics to drastically reduce the number of job terminations when scheduling under variable capacity. Although data center flexibility and scheduling for variable capacity have been studied in various contexts, none of the existing work exploits machine intervals to inform scheduling decisions.

CHAPTER 8

CONCLUSION AND FUTURE DIRECTIONS

8.1 Conclusion

In this thesis, we study opportunities to avoid terminations under variable capacity by exploiting machine interval characteristics for scheduling.

We observed that variable capacity in cloud datacenters leads to significant degradation in workload performance, with 12–24% and 5–19% goodput loss in Azure and Borg workloads, respectively, under increasing change frequency. We determined that machine interval ends in variable capacity lead to job terminations, resulting in goodput loss. This is especially evident in the proportionality between the number of job terminations and the number of machine intervals under the variable capacity profile. Therefore, job terminations are a crucial performance metric for scheduling under variable capacity. Cloud workloads are especially susceptible to job terminations, owing to various downstream effects on cloud services. We advocate for the community to consider job terminations as the primary scheduling objective for cloud workloads under variable capacity. We further characterized the impact of workload characteristics on performance under variable capacity and found greater performance degradation in heavier-tailed workloads. Analyzing observed variable capacity driven by variations in power grids' carbon intensity, we established that grids with higher carbon variation suffer greater performance degradation under variable capacity.

The underlying capacity variation structure determines the distribution of machine intervals. We characterized machine intervals under synthetic and observed variable capacity, and identified machine interval characteristics that could predict a machine interval's potential end. We proposed a class of scheduling heuristics that exploits machine interval characteristics to better align jobs to machine interval under variable capacity. We evaluated these heuristics individually, on their effectiveness in reducing terminations in the targeted job

population. Considering a combination of heuristics, we propose the Interval-Aware Scheduler (IAS) that drastically reduces terminations across a broad range of workloads. IAS achieves 5% greater goodput than First-Fit, with a 32% reduction in terminations. IAS can further reduce terminations by up to 98% over First-Fit, if a 1.25% goodput loss is tolerable. Balancing the trade-off between terminations and goodput depends on the workload's requirements and cost function. For the extremely heavy-tailed Azure workload, IAS achieves 92% lower terminations under variable capacity, while maintaining similar goodput to First-Fit (-0.05%). IAS achieves greater goodput in less heavy-tailed workloads, with goodput gains of up to 3.8% over First-Fit. As the workload becomes heavier-tailed, there is less opportunity to increase goodput, but the relative reduction in terminations over First-Fit increases.

8.2 Future Directions

Building on the insights and findings in this work, we outline a few promising research directions for future exploration in the space of variable capacity data centers.

Workload flexibility can help mitigate the performance implications of variable capacity, offering means to adapt to capacity changes more effectively. However, promoting workload flexibility and motivating developers to build flexible applications remains a challenge. Future work could explore incentive structures for developing flexible applications by analyzing the financial impact of performance loss costs associated with various workloads, such as AI training and inference, streaming services, scientific discovery, and digital twin applications, with the objective of developing pricing models for workload flexibility. Other studies could investigate the effectiveness and range of tolerable capacity variation provided by specific methods for enabling workload flexibility, including workload grouping for cost-effective migration, running at initially lower precision levels and expanding precision based on subsequent results, and affordable interruptions through checkpointing.

Coordination is integral for effectively managing variable capacity, particularly between power grids and data centers. Our work could be expanded by identifying power grid information (e.g., day-ahead planning) that could allow accurate forecasting of future capacity in variable capacity data centers. Coordinating networks of cloud data centers under variable capacity could exploit geographical carbon intensity and time zone differences. However, additional studies are required in understanding the challenges and opportunities, especially in preserving cloud providers' competitive advantage and fairness. Defining and operating flexible platforms imposes significant design and operational complexities, requiring further research in flexibility across the full ecosystem of the data center, including cooling, networking, and storage.

Scheduling models and metrics under traditional fixed capacity scheduling are no longer sufficient to quantify the implications of variable capacity. There is a research gap in performance metrics tailored towards cloud workload performance under variable capacity, with considerations for SLO violations, impact to user experience, and measuring application-specific performance loss. Moreover, information exchange between the scheduler, workload and platform, is crucial for preserving performance under variable capacity. Future research could explore methods for enabling effective real-time information exchange, minimizing the impact to scheduling decision time.

REFERENCES

[1] Amazon Web Services, Inc. AWS Compute Service Level Agreement, 2024. Accessed: January 30, 2025.

[2] Farag Azzedin and Muthucumaru Maheswaran. Integrating trust into grid resource management systems. In *Proceedings international conference on parallel processing*, pages 47–54. IEEE, 2002.

[3] Philippe Baptiste. Scheduling unit tasks to minimize the number of idle periods: a polynomial time algorithm for offline dynamic power management. In *SODA*, volume 6, pages 364–367, 2006.

[4] Salman A Baset. Cloud slas: present and future. *ACM SIGOPS Operating Systems Review*, 46(2):57–66, 2012.

[5] Anton Beloglazov and Rajkumar Buyya. Energy efficient resource management in virtualized cloud data centers. In *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 826–831. IEEE, 2010.

[6] Marin Bougeret, Henri Casanova, Mikael Rabie, Yves Robert, and Frédéric Vivien. Checkpointing strategies for parallel jobs. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–11, 2011.

[7] Wesley Brewer, Matthias Maiterth, Vineet Kumar, Rafal Wojda, Sedrick Bouknight, Jesse Hines, Woong Shin, Scott Greenwood, David Grant, Wesley Williams, et al. A digital twin framework for liquid-cooled supercomputers as demonstrated at exascale. *arXiv preprint arXiv:2410.05133*, 2024.

[8] Brendan Burns, Brian Grant, David Oppenheimer, Eric Brewer, and John Wilkes. Borg, omega, and kubernetes. *Commun. ACM*, 59(5):50–57, April 2016.

[9] California Independent System Operator (CAISO). California independent system operator. `https://www.caiso.com`. Accessed: 2024-12-01.

[10] Department of Market Monitoring California ISO. Demand response issues and performance 2023. `https://www.caiso.com/Documents/Demand-Response-Report-2023-Mar-6-2024.pdf`, 2024. [Accessed 29-10-2024].

[11] Niangjun Chen, Xiaoqi Ren, Shaolei Ren, and Adam Wierman. Greening multi-tenant data center demand response. *ACM SIGMETRICS Performance Evaluation Review*, 43(2):36–38, 2015.

[12] Andrew A Chien, Liuzixuan Lin, Hai Nguyen, Varsha Rao, Tristan Sharma, and Rajini Wijayawardana. Reducing the carbon impact of generative ai inference (today and in 2035). In *Proceedings of the 2nd workshop on sustainable computer systems*, pages 1–7, 2023.

[13] Andrew A Chien, Rich Wolski, and Fan Yang. Zero-carbon cloud: A volatile resource for high-performance computing. In *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, pages 1997–2001. IEEE, 2015.

[14] Andrew A Chien, Richard Wolski, and Fan Yang. The zero-carbon cloud: High-value, dispatchable demand for renewable power generators. *The Electricity Journal*, 28(8):110–118, 2015.

[15] Andrew A Chien, Chaojie Zhang, and Liuzixuan Lin. Beyond pue: Flexible datacenters empowering the cloud to decarbonize. *USENIX Hot Carbon*, 2022.

[16] Google Cloud. Google Cloud Platform Service Level Agreements. `https://cloud.google.com/terms/sla`. [Accessed 13-11-2024].

[17] California Energy Commission. 2023 Total System Electric Generation. `https://www.energy.ca.gov/data-reports/energy-almanac/california-electricity-data/2023-total-system-electric-generation`. [Accessed 11-11-2024].

[18] Microsoft Corporation. 2024 environmental sustainability report, 2024. Accessed: 2024-12-01.

[19] Eli Cortez, Anand Bonde, Alexandre Muzio, Mark Russinovich, Marcus Fontoura, and Ricardo Bianchini. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In *Proceedings of the 26th Symposium on Operating Systems Principles*, pages 153–167, 2017.

[20] Sheng Di, Yves Robert, Frédéric Vivien, and Franck Cappello. Toward an optimal online checkpoint solution under a two-level hpc checkpoint model. *IEEE Transactions on parallel and distributed systems*, 28(1):244–259, 2016.

[21] Jack Dongarra, Thomas Herault, and Yves Robert. *Fault tolerance techniques for high-performance computing*. Springer, 2015.

[22] Electric Power Research Institute. Epri launches initiative to enhance data center flexibility and grid reliability, October 2024. Accessed: 2024-12-01.

[23] Electric Reliability Council of Texas. Fuel mix report: 2023, 2024. Accessed: 2024-11-11.

[24] Electric Reliability Council of Texas (ERCOT). Electric reliability council of texas. `https://www.ercot.com`. Accessed: 2024-12-01.

[25] Electric Reliability Council of Texas (ERCOT). Large flexible load task force (lfltf) — ercot.com. `https://www.ercot.com/committees/tac/lfltf`, 2024. [Accessed 29-10-2024].

[26] Vincent C Emeakaroha, Marco AS Netto, Rodrigo N Calheiros, Ivona Brandic, Rajkumar Buyya, and César AF De Rose. Towards autonomic detection of sla violations in cloud infrastructures. *Future Generation Computer Systems*, 28(7):1017–1029, 2012.

[27] NRG Energy. Illinois miso fuel mix chart, 2024. Accessed: 2024-11-11.

[28] Wedan Emmanuel Gnibga, Andrew A Chien, Anne Blavette, and Anne Cécile Orgerie. Flexcooldc: Datacenter cooling flexibility for harmonizing water, energy, carbon, and cost trade-offs. In *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems*, pages 108–122, 2024.

[29] Íñigo Goiri, Kien Le, Thu D. Nguyen, Jordi Guitart, Jordi Torres, and Ricardo Bianchini. Greenhadoop: leveraging green energy in data-processing frameworks. In *Proceedings of the 7th ACM European Conference on Computer Systems*, EuroSys '12, page 57–70, New York, NY, USA, 2012. Association for Computing Machinery.

[30] Íñigo Goiri, William Katsak, Kien Le, Thu D Nguyen, and Ricardo Bianchini. Parasol and greenswitch: Managing datacenters powered by renewable energy. *ACM SIGPLAN Notices*, 48(4):51–64, 2013.

[31] Íñigo Goiri, Kien Le, Md E Haque, Ryan Beauchea, Thu D Nguyen, Jordi Guitart, Jordi Torres, and Ricardo Bianchini. Greenslot: scheduling energy consumption in green datacenters. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–11, 2011.

[32] Google. Borg cluster traces from google, 2019. Accessed: 2024-11-30.

[33] Google Cloud. Google Cloud Compute Engine Service Level Agreement, 2024. Accessed: January 30, 2025.

[34] Jing Guo, Zihao Chang, Sa Wang, Haiyang Ding, Yihui Feng, Liang Mao, and Yungang Bao. Who limits the resource efficiency of my datacenter: an analysis of alibaba datacenter traces. In *Proceedings of the international symposium on quality of service*, pages 1–10, 2019.

[35] Sophie Hall, Francesco Micheli, Giuseppe Belgioioso, Ana Radovanović, and Florian Dörfler. Carbon-aware computing for data centers with probabilistic performance guarantees. *arXiv preprint arXiv:2410.21510*, 2024.

[36] Walid A. Hanafy, Qianlin Liang, Noman Bashir, David Irwin, and Prashant Shenoy. Carbonscaler: Leveraging cloud workload elasticity for optimizing carbon-efficiency. In *Abstracts of the 2024 ACM SIGMETRICS/IFIP PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS/PERFORMANCE '24, page 49–50, New York, NY, USA, 2024. Association for Computing Machinery.

[37] Md E Haque, IŽigo Goiri, Ricardo Bianchini, and Thu D Nguyen. Greenpar: Scheduling parallel high performance applications in green datacenters. In *Proceedings of the 29th ACM on International Conference on Supercomputing*, pages 217–227, 2015.

[38] Thomas Herault, Yves Robert, Aurelien Bouteiller, Arnold Arnold, Kurt B Ferreira, George George, and Jack Dongarra. Checkpointing strategies for shared high-performance computing platforms. *International Journal of Networking and Computing*, 9(1):28–52, 2019.

[39] Benjamin Hindman, Andy Konwinski, Matei Zaharia, Ali Ghodsi, Anthony D Joseph, Randy Katz, Scott Shenker, and Ion Stoica. Mesos: A platform for {Fine-Grained} resource sharing in the data center. In *8th USENIX Symposium on Networked Systems Design and Implementation (NSDI 11)*, 2011.

[40] Nathaniel Horner, Inês Azevedo, Doug Sicker, and Yuvraj Agarwal. Dynamic data center load response to variability in private and public electricity costs. In *2016 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 80–85. IEEE, 2016.

[41] Peter Judge. Dominion energy admits it can't meet data center power demands in virginia. *Data Center Dynamics*, July 2022.

[42] Adam Lechowicz, Nicolas Christianson, Bo Sun, Noman Bashir, Mohammad Hajiesmaili, Adam Wierman, and Prashant Shenoy. Carbonclipper: Optimal algorithms for carbon-aware spatiotemporal workload management. *arXiv preprint arXiv:2408.07831*, 2024.

[43] Yunbo Li, Anne-Cécile Orgerie, and Jean-Marc Menaud. Balancing the use of batteries and opportunistic scheduling policies for maximizing renewable energy consumption in a cloud data center. In *2017 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*, pages 408–415. IEEE, 2017.

[44] Zhenhua Liu, Adam Wierman, Yuan Chen, Benjamin Razon, and Niangjun Chen. Data center demand response: avoiding the coincident peak via workload shifting and local generation. *SIGMETRICS Perform. Eval. Rev.*, 41(1):341–342, June 2013.

[45] Google LLC. Google 2024 environmental report, 2024. Accessed: 2024-12-01.

[46] Varun Mehra and Raiden Hasegawa. Supporting power grids with demand response at google data centers. *Google Cloud Blog*, October 2023. Accessed: 2024-12-01.

[47] Microsoft. Service level agreements (sla) for online services. `https://www.microsoft.com/licensing/docs/view/Service-Level-Agreements-SLA-for-Online-Services?lang=1`. [Accessed 13-11-2024].

[48] Microsoft. Azure public dataset v2, 2019. Accessed: 2024-11-30.

[49] Midcontinent Independent System Operator (MISO). Midcontinent independent system operator. https://www.misoenergy.org. Accessed: 2024-12-01.

[50] Jorge Murillo, Walid A. Hanafy, David Irwin, Ramesh Sitaraman, and Prashant Shenoy. Cdn-shifter: Leveraging spatial workload shifting to decarbonize content delivery networks. In *Proceedings of the 2024 ACM Symposium on Cloud Computing*, SoCC '24, page 505–521, New York, NY, USA, 2024. Association for Computing Machinery.

[51] Emma Newburger. Utilities face looming crunch as electricity demand from ai surges. *CNBC*, August 2024. Accessed: 2024-12-01.

[52] Mark EJ Newman. Power laws, pareto distributions and zipf's law. *Contemporary physics*, 46(5):323–351, 2005.

[53] Bloomberg News. Ai data centers strain power grids. *Bloomberg*, November 2024. Accessed: 2024-12-01.

[54] Lucas Perotin, Chaojie Zhang, Rajini Wijayawardana, Anne Benoit, Yves Robert, and Andrew Chien. Risk-aware scheduling algorithms for variable capacity resources. In *Proceedings of the SC '23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis*, SC-W '23, page 1306–1315, New York, NY, USA, 2023. Association for Computing Machinery.

[55] Southwest Power Pool. Fast Facts — spp.org. https://www.spp.org/about-us/fast-facts/. [Accessed 11-11-2024].

[56] Ana Radovanović, Ross Koningstein, Ian Schneider, Bokan Chen, Alexandre Duarte, Binz Roy, Diyue Xiao, Maya Haridasan, Patrick Hung, Nick Care, et al. Carbon-aware computing for datacenters. *IEEE Transactions on Power Systems*, 38(2):1270–1280, 2022.

[57] Amazon Web Services. AWS Service Level Agreements. https://aws.amazon.com/legal/service-level-agreements/?aws-sla-cards.sort-by=item.additionalFields.serviceNameLower&aws-sla-cards.sort-order=asc&awsf.tech-category-filter=*all. [Accessed 13-11-2024].

[58] Arman Shehabi, Sarah J. Smith, Alex Hubbard, Alexander Newkirk, Nuoa Lei, Md Abu Bakar Siddik, Billie Holecek, Jonathan Koomey, Eric Masanet, and Dale Sartor. 2024 united states data center energy usage report. Technical Report LBNL-2001637, Lawrence Berkeley National Laboratory, 2024.

[59] Shanshan Song, Kai Hwang, and Yu-Kwong Kwok. Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling. *IEEE Transactions on Computers*, 55(6):703–719, 2006.

[60] Jason Sonnek, Abhishek Chandra, and Jon Weissman. Adaptive reputation-based scheduling on unreliable distributed infrastructures. *IEEE Transactions on Parallel and Distributed Systems*, 18(11):1551–1564, 2007.

[61] Southwest Power Pool (SPP). Southwest power pool. `https://www.spp.org`. Accessed: 2024-12-01.

[62] Jovan Stojkovic, Pulkit A Misra, Íñigo Goiri, Sam Whitlock, Esha Choukse, Mayukh Das, Chetan Bansal, Jason Lee, Zoey Sun, Haoran Qiu, et al. Smartoclock: Workload- and risk-aware overclocking in the cloud. In *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)*, pages 437–451. IEEE, 2024.

[63] Dan Swinhoe. Eirgrid pulls plug on 30 irish data center projects. *Data Center Dynamics*, May 2022.

[64] Muhammad Tirmazi, Adam Barker, Nan Deng, Md Ehtesam Haque, Zhijing Gene Qin, Steven Hand, Mor Harchol-Balter, and John Wilkes. Borg: the next generation. In *EuroSys'20*, Heraklion, Crete, 2020.

[65] T. Bruce Tsuchida, Long Lam, Peter Fox-Penner, Akhilesh Ramakrishnan, Sylvia Tang, Adam Bigelow, and Ethan Snyder. Electricity demand growth and forecasting in a time of change. Technical report, The Brattle Group, May 2024. Accessed: 2024-12-01.

[66] Abhishek Verma, Luis Pedrosa, Madhukar Korupolu, David Oppenheimer, Eric Tune, and John Wilkes. Large-scale cluster management at google with borg. In *Proceedings of the tenth european conference on computer systems*, pages 1–17, 2015.

[67] Philipp Wiesner, Ilja Behnke, Dominik Scheinert, Kordian Gontarska, and Lauritz Thamsen. Let's wait awhile: How temporal workload shifting can reduce carbon emissions in the cloud. In *Proceedings of the 22nd International Middleware Conference*, pages 260–272, 2021.

[68] Chia-Ming Wu, Ruay-Shiung Chang, and Hsin-Yu Chan. A green energy-efficient scheduling algorithm using the dvfs technique for cloud datacenters. *Future Generation Computer Systems*, 37:141–147, 2014.

[69] Jiali Xing, Bilge Acun, Aditya Sundarrajan, David Brooks, Manoj Chakkaravarthy, Nikky Avila, Carole-Jean Wu, and Benjamin C Lee. Carbon responder: Coordinating demand response for the datacenter fleet. *arXiv preprint arXiv:2311.08589*, 2023.

[70] Minxian Xu and Rajkumar Buyya. Managing renewable energy and carbon footprint in multi-cloud computing environments. *Journal of Parallel and Distributed Computing*, 135:191–202, 2020.

[71] Niva Yadav. Taiwan to stop large data centers in the north, cites insufficient power. *DataCenter Dynamics*, August 2024. Accessed: 2024-12-01.

[72] Chaojie Zhang. *Eliminating the Capacity Variation Penalty for Cloud Resource Management*. PhD thesis, University of Chicago, 2023. Accessed: 2024-12-01.

[73] Chaojie Zhang and Andrew A Chien. Scheduling challenges for variable capacity resources. In *Job Scheduling Strategies for Parallel Processing: 24th International Workshop, JSSPP 2021, Virtual Event, May 21, 2021, Revised Selected Papers 24*, pages 190–209. Springer, 2021.

[74] Chaojie Zhang, Alok Gautam Kumbhare, Ioannis Manousakis, Deli Zhang, Pulkit A. Misra, Rod Assis, Kyle Woolcock, Nithish Mahalingam, Brijesh Warrier, David Gauthier, Lalu Kunnath, Steve Solomon, Osvaldo Morales, Marcus Fontoura, and Ricardo Bianchini. Flex: High-availability datacenters with zero reserved power. In *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, pages 319–332, 2021.

[75] Jiajia Zheng, Andrew A Chien, and Sangwon Suh. Mitigating curtailment and carbon emissions through load migration between data centers. *Joule*, 4(10):2208–2222, 2020.